



Journal of Language Modelling

VOLUME 8 ISSUE 1
JUNE 2020



*Institute of Computer Science
Polish Academy of Sciences
Warsaw*

Journal of Language Modelling

VOLUME 8 ISSUE 1
JUNE 2020

Articles

- Minimal phrase structure:
a new formalized theory of phrase structure 1
John J. Lowe, Joseph Lovestrand
- Distinguishing between paradigmatic semantic relations
across word classes: human ratings and distributional similarity 53
Sabine Schulte im Walde
- Neural network models for phonology and phonetics 103
Paul Boersma, Titia Benders, Klaas Seinhorst
- Computing and classifying reduplication
with 2-way finite-state transducers 179
Hossep Dolatian, Jeffrey Heinz



JOURNAL OF
LANGUAGE MODELLING

ISSN 2299-8470 (electronic version)

ISSN 2299-856X (printed version)

<http://jlm.ipipan.waw.pl/>

MANAGING EDITOR

Adam Przepiórkowski IPI PAN

SECTION EDITORS

Elżbieta Hajnicz IPI PAN

Agnieszka Mykowiecka IPI PAN

Marcin Woliński IPI PAN

STATISTICS EDITOR

Łukasz Dębowski IPI PAN



Published by IPI PAN


Institute of Computer Science, Polish Academy of Sciences
ul. Jana Kazimierza 5, 01-248 Warszawa, Poland

Circulation: 100 + print on demand

Layout designed by Adam Twardoch.

Typeset in X_YL^AT_EX using the typefaces: *Playfair*
by Claus Eggers Sørensen, *Charis SIL* by SIL International,
JLM monogram by Łukasz Dziedzic.

*All content is licensed under
the Creative Commons Attribution 4.0 International License.*

 <http://creativecommons.org/licenses/by/4.0/>

EDITORIAL BOARD

Steven Abney University of Michigan, USA

Ash Asudeh Carleton University, CANADA;
University of Oxford, UNITED KINGDOM

Chris Biemann Technische Universität Darmstadt, GERMANY

Igor Boguslavsky Technical University of Madrid, SPAIN;
Institute for Information Transmission Problems,
Russian Academy of Sciences, Moscow, RUSSIA

António Branco University of Lisbon, PORTUGAL

David Chiang University of Southern California, Los Angeles, USA

Greville Corbett University of Surrey, UNITED KINGDOM

Dan Cristea University of Iași, ROMANIA

Jan Daciuk Gdańsk University of Technology, POLAND

Mary Dalrymple University of Oxford, UNITED KINGDOM

Darja Fišer University of Ljubljana, SLOVENIA

Anette Frank Universität Heidelberg, GERMANY

Claire Gardent CNRS/LORIA, Nancy, FRANCE

Jonathan Ginzburg Université Paris-Diderot, FRANCE

Stefan Th. Gries University of California, Santa Barbara, USA

Heiki-Jaan Kaalep University of Tartu, ESTONIA

Laura Kallmeyer Heinrich-Heine-Universität Düsseldorf, GERMANY

Jong-Bok Kim Kyung Hee University, Seoul, KOREA

Kimmo Koskenniemi University of Helsinki, FINLAND

Jonas Kuhn Universität Stuttgart, GERMANY

Alessandro Lenci University of Pisa, ITALY

Ján Mačutek Comenius University in Bratislava, SLOVAKIA

Igor Mel'čuk University of Montreal, CANADA

Glyn Morrill Technical University of Catalonia, Barcelona, SPAIN

Stefan Müller Freie Universität Berlin, GERMANY

Mark-Jan Nederhof University of St Andrews, UNITED KINGDOM

Petya Osenova Sofia University, BULGARIA

David Pesetsky Massachusetts Institute of Technology, USA

Maciej Piasecki Wrocław University of Technology, POLAND

Christopher Potts Stanford University, USA

Louisa Sadler University of Essex, UNITED KINGDOM

Agata Savary Université François Rabelais Tours, FRANCE

Sabine Schulte im Walde Universität Stuttgart, GERMANY

Stuart M. Shieber Harvard University, USA

Mark Steedman University of Edinburgh, UNITED KINGDOM

Stan Szpakowicz School of Electrical Engineering
and Computer Science, University of Ottawa, CANADA

Shravan Vasishth Universität Potsdam, GERMANY

Zygmunt Vetulani Adam Mickiewicz University, Poznań, POLAND

Aline Villavicencio Federal University of Rio Grande do Sul,
Porto Alegre, BRAZIL

Veronika Vincze University of Szeged, HUNGARY

Yorick Wilks Florida Institute of Human and Machine Cognition, USA

Shuly Wintner University of Haifa, ISRAEL

Zdeněk Žabokrtský Charles University in Prague, CZECH REPUBLIC

Minimal phrase structure: a new formalized theory of phrase structure

John J. Lowe¹ and Joseph Lovestrand²

¹ University of Oxford

² SOAS, University of London

ABSTRACT

X' theory was a major milestone in the history of the development of generative grammar.¹ It enabled important insights to be made into the phrase structure of human language, but it had a number of weaknesses, and has been essentially replaced in Chomskyan generativism by Bare Phrase Structure (BPS), which assumes fewer theoretical primitives than X' theory, and also avoids several of the latter's weaknesses. However, Bare Phrase Structure has not been widely adopted outside the Minimalist Program (MP), rather, X' theory remains widespread. In this paper, we develop a new, fully formalized approach to phrase structure which incorporates insights and advances from BPS, but does not require the Minimalist-specific assumptions that come with BPS. We formulate our proposal within Lexical-Functional Grammar (LFG), providing an empirically and theoretically superior model for phrase structure compared with standard versions of X' theory current in LFG.

Keywords:
phrase structure,
X' theory, Bare
Phrase Structure,
Lexical-Functional
Grammar

¹ We are grateful to the audiences at the University of Oxford Syntax Working Group (8 June 2016), at SE-LFG23 (13 May 2017), and at LFG17 (25 July 2017), where earlier versions of these proposals were presented. In particular we are grateful to Adam Przepiórkowski for insightful criticisms and helpful suggestions. We also thank the editors and anonymous reviewers. All remaining errors are our own.

INTRODUCTION

X' theory, first introduced in Chomsky (1970) and elaborated in Jackendoff (1977) among other works, was a major milestone in the history of the development of generative grammar. It provided, for the first time, a mechanism for capturing generalizations and constraints on possible phrase structures in language. X' theory originated as a means of generalizing over sets of phrase structure rules (PSRs), but in the early 1980s, within the Principles & Parameters model, it led to the abandonment of PSRs as a part of the grammar of individual languages. X' theory encapsulated important insights into the phrase structure of human language, but it had a number of weaknesses, and has been essentially replaced in Chomskyan generativism by Bare Phrase Structure (Chomsky 1995). Bare Phrase Structure (BPS) assumes fewer theoretical primitives than X' theory, and is therefore preferable from a minimalist perspective; it also avoids several of the empirical and theoretical weaknesses of X' theory. However, Bare Phrase Structure is unavoidably associated with a number of assumptions which are theory-specific to the Minimalist Program (MP) – most obviously perhaps, its derivational nature – and for this reason has not been widely adopted outside the MP.

Where Bare Phrase Structure is not adopted, X' theory remains the most widespread approach to phrase structure, and it remains the standard means of approaching phrase structure in most introductory text books. The grammatical framework of Lexical-Functional Grammar (LFG: Kaplan and Bresnan 1982) retains X' theory in largely its original form (i.e. as a set of cross-linguistic generalizations over PSRs in the grammars of individual languages), and thus retains both the benefits and weaknesses of this approach to phrase structure. We take the version of X' theory currently utilized in LFG to be the most elaborate and precisely formalized version of X' theory currently in use.

In this paper, we develop a new, fully formalized approach to phrase structure within LFG which avoids the major weaknesses of X' theory and incorporates many of the advantages of BPS.² While for-

²An early version of our proposal was made in Lovstrand and Lowe (2017). The present version differs in significant ways, most importantly in its use of distributive features (Section 3.3) to eliminate redundancy in labelling.

malized within LFG, our proposal is easily extensible to other theories. Our model has been tested within the computational implementation of LFG, the Xerox Linguistic Environment (XLE: Crouch *et al.* 2011).³

CONSTRAINING PHRASE STRUCTURE

2

Since the introduction of PSRs by Chomsky (1957) as a central component of the theory of formal syntax, there has been significant progress constraining this formal mechanism to approximate the actual types of phrase structures that are attested in languages, and to prevent the theory from being able to produce unattested phrase structures. The most significant milestone in the development of the theory of phrase structure was the development of X' theory. However, X' theory had a number of inadequacies which ultimately led to its replacement in the mainstream Chomskyan tradition. In this paper, we focus on seven features lacking from X' theory which should form a part of an adequate theory of phrase structure; most but not all of these are found in BPS. An adequate theory of phrase structure should (in contrast with existing formalized versions of X' theory):

- (1) a. Utilize only as much structure as required to model constituency, avoiding nonbranching dominance chains.
- b. Avoid the assumption of massive/default optionality in PSRs.
- c. Avoid redundancy in category labelling, ensuring that endocentric phrases necessarily share the category of their head without stipulation.
- d. Lack a distinct notion of X' .
- e. Incorporate a notion of X^{max} distinct from 'XP', and a notion of the highest projection distinct from X^{max} .

³Being formulated within LFG, our model functions as a set of constraints on language-specific PSRs, but it is important to note that our proposal could without difficulty be reinterpreted within different frameworks purely as a set of constraints on phrase structure more generally, with no language-specific PSRs as such.

- f. Incorporate a principled account of exocentricity.
- g. Incorporate a principled account of nonprojecting categories.

Most of the desiderata in (1) address specific issues that have arisen in the development of X' theory. BPS has addressed many of these issues, though not all. The last two desiderata expand the coverage of the theory of phrase structure to include two types of non- X' -theoretic structures: nonprojecting words and exocentric structures are adopted in X' -theoretic approaches to phrase structure in LFG, but have not been formally incorporated in the theory. Our proposal below is the first fully formalized theory of phrase structure that satisfies all of the desiderata in (1).

In the following sections, we discuss two contemporary approaches to phrase structure: the version of X' theory current within LFG, which we take to be the most fully developed version of X' theory currently in use; and BPS, the standard approach to phrase structure within the Chomskyan generative tradition.

2.1

Current X' theory in LFG

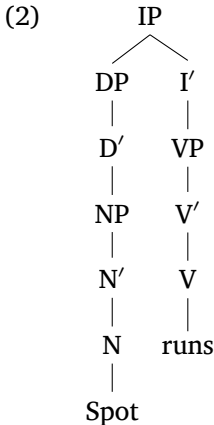
X' theory began as a means of stating generalizations over sets of PSRs.⁴ Following Stowell (1981), X' theory was reconceived within the Principles & Parameters framework as a set of universal constraints on phrase structure, and subsequently language-specific PSRs themselves were eliminated; language-specific characteristics of phrase structure were instead constrained by syntactic processes, such as the assignment of Case. This final step was not taken in LFG. In LFG, X' theory remains a means of generalizing over and stating constraints on sets of PSRs. PSRs themselves cannot be eliminated, because they constitute the main body of non-lexical constraints in a grammar. A minimal Lexical-Functional Grammar consists of a set of lexical entries and a set of PSRs; grammatical structure is, and can only be built by the application of specific PSRs (which ultimately license insertion of lexical information).

⁴A detailed introduction to X' theory and its development is provided by Carnie (2010, chapter 7). See also Carnie (2000) and Kornai and Pullum (1990).

The advantage of LFG's phrase-structure based approach to structure building is its computational efficiency: despite being a unification-based system, which therefore in principle has the power of an unrestricted rewriting system, the structure-building component of an LFG is a context-free phrase structure grammar; as shown by Maxwell and Kaplan (1996), appropriate interleaving of context-free parsing and f-structure unification can be computed in cubic time.

Despite the obvious strengths which led to its great success, and which were largely adopted into BPS, X' theory suffers from a number of weaknesses; see Kornai and Pullum (1990) for a detailed examination of the theoretical weaknesses of X' theory. We focus here on X' theory as it is currently conceived and used within LFG, which admits a number of extensions to and alterations of the strict principles of X' theory in its original formulation.

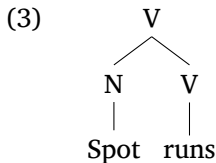
We focus on four main weakness of X' theory as utilized within LFG, all of which are evident in (2), a standard LFG constituent structure for the sentence *Spot runs*: nonbranching dominance chains, optionality of daughters (related to the existence of nonbranching dominance chains, of course, but including heads), redundancy in category labelling, and the need to assume intermediate (X') nodes as an independent theoretical construct (cf. (1a–d)). We discuss these issues in turn.



As discussed in Section 3.1, the LFG representation of phrase structure, c(onstituent)-structure, models only surface constituency relations, while functional syntactic relations are modelled at a separate

level of structure, f(unctional)-structure. Thus phrases which consist of only one word, like the DP *Spot* and the VP *runs*, can only be modelled within LFG's approach to X' theory by assuming nonbranching dominance chains, such as the DP chain in (2), where we have four nonbranching nodes dominating the N. There can be no silent specifier, head or complement positions hosting functional features to fill out the tree, because such features are represented at f-structure and, as stated, the tree models only the surface constituency relations of the overt elements of the sentence.⁵ Even within syntactic theories which admit empty nodes, adherence to X' theory would still involve some nonbranching dominance chains (though perhaps not as long as in (2)).

Although nonbranching chains as in (2) do model relevant properties of the structure, such as the dual maximality (phrasality) and minimality of the individual words, the resulting structure, involving ten nonterminal nodes, seems inordinately complex as a representation of the surface constituency of a two word sentence. This constituency could be equally well captured by the tree in (3), which is considerably more in the spirit of BPS. Our proposal below licenses structures equivalent to (3).



Related to this problem is the issue of optionality of phrase structure nodes (cf. (1b)). Clearly, dominance chains like XP-X'-X require that specifier and complement positions be optional. But as can be seen in (2), heads can also be optional. This must be possi-

⁵There is some debate within LFG over the existence of traces, i.e. whether there may be some terminal nodes in a c-structure which do not correspond to any overt element. Arguments against traces were made by Kaplan and Zaenen (1989), and widely accepted within the LFG community; traces are accepted by Bresnan (1995, 1998, 2001) and Bresnan *et al.* (2016) only in order to account for weak crossover, but analyses of weak crossover which do not involve traces are offered by Dalrymple *et al.* (2001, 2007), Nadathur (2013) and Dalrymple and King (2013).

ble for functional categories like I and D, on the assumption, standard in LFG, that V and N are necessarily dominated by these categories (as in (2)). But many analyses also require heads of lexical phrases to be optional. Most work in LFG, therefore, including the standard textbooks of Bresnan (2001) and Dalrymple (2001), assume that all phrase structure positions are in principle optional, heads and nonheads alike. However, there are certain structures in some languages in which optionality must be suppressed; see Snijders (2012) and Dalrymple *et al.* (2015, 386–388) for detailed discussion of such cases.⁶

Optionality as the default situation, ruled out in certain circumstances, is widely assumed in existing LFG analyses, but has never been properly formalized: in LFG, the right-hand side of a PSR must be a regular expression; in regular expressions it is optionality (defined as disjunction with the empty set), not obligatoriness, which has to be specified. In contrast, it would be more intuitive, and PSRs would be considerably less ambiguous, if optionality were the exception, rather than the rule. The model we present below avoids the need for mass optionality, treating optionality as an occasional necessity, rather than a default.

A further weakness of X' theory involves another type of redundancy in representation: each node is independently specified with a category label, but given the inherent constraints on X' -theoretic structures, each node in a projection chain necessarily has the same category label, meaning that it ought not to be necessary to specify this information more than once for each projection chain. That is, the notion that a phrasal node necessarily has the same category label as its head ought to fall out naturally, rather than by stipulation, which is essentially the way it has to be done in X' theory. Our proposal makes use of the concept of *distributive features* to ensure that only a single instance of category labelling applies for each projection chain.

The fourth major weakness of X' theory is that it entails the existence of the intermediate node type X' as an independent theoretical construct (cf. (1d)). However, a wealth of research has demonstrated that there is no clear evidence of syntactic processes which

⁶ See further Lovstrand and Lowe (2017, 289–290).

make reference to the X' level, suggesting that it is not an independent concept in human language.⁷

2.2

Further problems: augmenting X' theory

In attempting to provide a sufficiently flexible model of phrase structure to adequately capture the wide range of crosslinguistic variation in surface configurational syntactic structure, LFG has been forced to admit certain augmentations to the basic X'-theoretic structures it inherited. These augmentations are not problematic in themselves, but they have never been properly integrated into existing formal analyses of X' theory.

In addition to endocentric phrase structures, LFG also admits *exocentric* structures, most commonly the exocentric clausal category S (Bresnan 1982; Kroeger 1993; Bresnan 2001). S is not subject to ordinary X'-theoretic constraints: it is a non-headed category that may contain a predicate along with any or all of its arguments. S is most commonly utilized in the analysis of non-configurational languages (Austin and Bresnan 1996; Nordlinger 1998), but it is also utilized in some analyses of languages with relatively fixed word order, such as Welsh (Sadler 1997) and Barayin (Lovestrand 2018).⁸ While S, and sometimes other exocentric categories, are widely admitted in LFG, recent formalizations of X' theory find no place for exocentricity, leaving it outside the formal system while nevertheless remaining crucial to actual grammars and analyses.

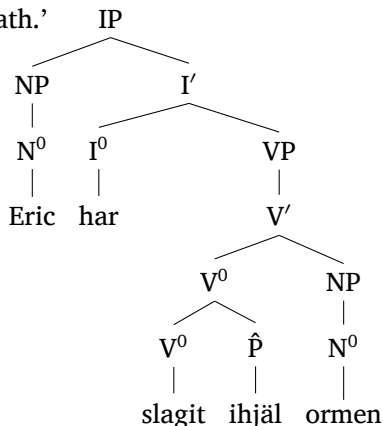
A further concept widely adopted within LFG is that of nonprojecting categories. Toivonen (2003) argues that alongside the traditional projecting lexical categories, there exist also *nonprojecting* categories, represented as \hat{X} , which adjoin to X^0 (projecting) heads. Nonprojecting words do not head phrases, and so it is not possible for another phrase to stand in a specifier, complement or adjunct relation to such a word. Nonprojecting words are often particles and/or clitics. Toivonen argues in detail that verb particles in Swedish are nonpro-

⁷ Early arguments in Travis (1984), see also Carnie (2000, 2010).

⁸ See example (38) below.

jecting \hat{P} s, giving the example in (4), and proposing the augmentation to X' theory shown in (5).⁹

- (4) Eric har slagit ihjäl ormen
 Eric has beaten to.death snake.DEF
 ‘Eric has beaten the snake to death.’



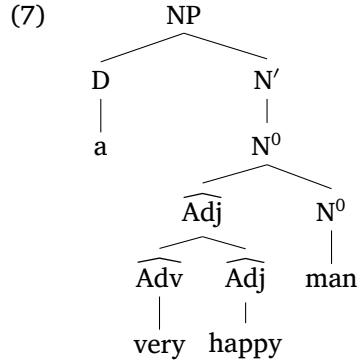
- (5) $X^0 \rightarrow X^0 , \hat{Y}$

The possibilities for nonprojecting words have been further broadened by other authors, relaxing Toivonen’s (2003) assumption that nonprojecting words adjoin only to X^0 heads. Spencer (2005) argues for adjunction of nonprojecting words to phrasal categories, as well as to X^0 heads, in order to capture the properties of case clitics in Hindi. Duncan (2007) and, more recently, Arnold and Sadler (2013), propose that nonprojecting categories may also adjoin to nonprojecting categories. Arnold and Sadler (2013) base their proposals on the relatively familiar features of prenominal modification in English. Building on work by Poser (1992) and Sadler and Arnold (1994), they argue that prenominal modification in English should be analysed in terms of nonprojecting categories; this accounts for the fact that prenominal adjectives cannot take postpositioned complements or modifiers, unlike adjectives in other positions. But since prenominal modification is recursive, this requires that nonprojecting categories

⁹The comma in the templatic PSR in (5) is the ‘shuffle’ operator, indicating variable order of the sequences on either side. For its use in LFG see Dalrymple *et al.* (2019, 204–205).

can be adjoined not only to X^0 , but also to nonprojecting \hat{X} s. That is, we require a rule of the kind in (6); the analysis proposed by Arnold and Sadler (2013) for prenominal modification in English is shown in (7).

(6) $\hat{X} \rightarrow \hat{Y} \hat{X}$



Here the nonprojecting category $\widehat{\text{Adj}}$ adjoins to N^0 , while nonprojecting $\widehat{\text{Adv}}$ adjoins to $\widehat{\text{Adj}}$.

Once again, existing formalizations of X' theory within LFG do not adequately account for nonprojecting categories. Our proposal does so, and we model our approach to nonprojecting categories with respect to English prenominal modification, adopting the proposals of Arnold and Sadler (2013) illustrated here. Our model also allows in principle for adjunction of a nonprojecting node (or any kind of node) to a phrasal category, XP , as proposed by Spencer (2005).¹⁰

2.3

BPS

The origins of BPS have been discussed in detail by a number of authors, including Carnie (2010, 135–167), and here we will focus only on the major innovations and insights which distinguish BPS from X' theory.¹¹ In general, and in line with the Minimalist Program, BPS

¹⁰This possibility is not modelled below, but it could be achieved by modifying the adjunction rule in (36b) so that the template @LOM is replaced by @LPM. For English prenominal modification, this would be necessary to capture the constituency of phrases like [*small [book of poems]*].

¹¹Formalizations of the principles of BPS are given by e.g. Stabler (1997), Gärtner (2002) and Collins and Stabler (2016). We discuss the latter work below.

aims to incorporate the major insights of X' theory not as stipulations but as the natural consequences of deeper principles. In doing this, certain problematic aspects of X' theory have been discarded.

One early identification of a major weakness in X' theory was by Fukui (1986), who shows that the amount of structure found with particular types of projection may vary crosslinguistically; in particular, in some languages functional categories lack specifiers. Fukui draws the conclusion that there is a difference between XP (understood as X'') and X^{max} , a maximal projection: some maximal projections are equivalent to X' . Thus, if there is cross- or even intra-language variation in the amount of structure admitted in different projections, X' theory provides no coherent notion of a maximal projection. As noted by Lovestrland and Lowe (2017, 288–289), this weakness persists in X' theory as utilized within LFG; for example, Bresnan *et al.* (2016, 130) permit phrases to lack specifiers “as a parametric choice”, without addressing the formal problems this raises.

Similar problems with distinguishing X^{max} from the top projection, in cases of adjunction, are discussed by Hornstein and Nunes (2008): if the properties of mother and head daughter are identical in adjunction structures, then adjunction to X^{max} results in multiple X^{max} projections; only one X^{max} is the top projection, but this cannot be formally distinguished from the others.¹² Our proposal below can capture both the distinction between XP and X^{max} , and between X^{max} and the top projection.

The consequence of Fukui’s separation of XP from X^{max} is a relativization of the notion of maximal category, and a concurrent weakening of the status of bar levels as absolute notions. A similarly relativized approach to projection levels was taken by Speas (1986). The underlying intuition is that the amount of structure in a phrase is only as much as needed to account for the constituency; maximal projections may correspond to X'' , X' , or even X , depending on the phrase in question. Thus, a node may be both maximal and minimal at the

¹² An alternative and more standard way of approaching adjunction within BPS involves the notion of ‘pair-merge’ (Chomsky 2001). We do not see how ‘pair-merge’ could be treated coherently within the framework adopted in this paper, and note that it has been criticized within the Chomskian tradition, e.g. by Hornstein and Nunes (2008).

same time; it is primarily this intuition which motivates X' theoretic structures like (2) to be simplified into structures more like (3). The relativized approach to X' theoretic notions proposed by Speas (1986) provides a coherent definition of X^{max} , which is lacking in X' theory.¹³ But at the same time, this approach eliminates a coherent notion of X' . Speas (1986) shows that this is a valid elimination, since there are no syntactic phenomena which necessarily make reference to the X' level (see also footnote 7).

The insights of Fukui (1986) and Speas (1986) fed into the theory of BPS as developed by Chomsky (1995). One of the fundamental features of BPS is the notion that all structure building can be attributed to a single basic syntactic operation, Merge. Merge takes two elements and forms them into a set, which is labelled with one of the two elements. The element which provides the label is the head.

The labelling mechanism is a further aspect of BPS relevant to the present discussion. For Chomsky (1995), the label of a merged structure is automatically derived from one of the merged elements. Thus labelling is a part of the definition of Merge, and as such the notion that a phrase necessarily has the same category label as its head falls out without further stipulation, given the definition of Merge. In contrast, as noted above, in X' theory the fact that a head X necessarily heads a phrase XP (rather than YP) falls out only by stipulation: PSRs, or constraints on PSRs, are stated in such a way that this intuition is not violated, but in principle different rules or constraints might have been stated which did violate the intuition. Following Collins (2002), some approaches to BPS go further, attempting to eliminate labelling altogether. While this is not universally accepted, it reflects the deeper aims of the MP to eliminate as far as possible all redundant elements of analysis.

Another central element of BPS is the concern with accounting for linearization patterns, building on the work of Kayne (1994). In the PSR-based approach we use as the basis for our proposals in this paper, linear order is a given, stipulated in the PSRs wherever determinate, with variable ordering a marked possibility. We therefore do not consider this aspect of BPS further here.

¹³Speas' definition of maximal projection, as emended by Carnie (2010, 139), runs: " $X = XP$ if $\exists G$, immediately dominating X , the head of $G \neq$ the head of X ."

Conclusion

2.4

In the foregoing discussion, we have identified seven main ways in which a theory of phrase structure should improve upon existing formalizations of X' theory and/or should incorporate insights from BPS. A formal model of phrase structure should avoid non-branching chains, and the default optional nodes associated with them. It should not stipulate a mid-level X' node, and should include a mechanism to distinguish a maximal node, in the sense of the mother of a structure including all specifiers and complements, from a higher node including adjunction structures. The theory should naturally produce endo-centric structures in which heads and mothers share category information, while at the same time successfully modeling nonprojecting and exocentric structures.

A NEW MODEL:
MINIMAL PHRASE STRUCTURE

3

Underlying architecture

3.1

As stated, our proposal is formalized within LFG. LFG is a constraint-based, non-derivational framework for grammatical analysis; handbooks include Dalrymple (2001), Falk (2001), Bresnan *et al.* (2016) and Dalrymple *et al.* (2019). A central aspect of the LFG framework is that it distinguishes different types of grammatical information and models them as distinct levels of grammatical representation. These levels are related to one another by means of *projection functions*.

One level of grammatical representation, central to the present topic, is the c(onstituent)-structure, which represents the phrasal structure of a clause. C-structure is represented as a phrase-structure tree, and constraints on possible c-structures are stated as PSRs. As discussed above, c-structure represents only the surface constituency of a clause or phrase, while more abstract functional syntactic properties and relations, such as grammatical functions, long-distance dependencies and agreement features, are dealt with at the level of f(unctional)-structure. F-structure is represented as an attribute-value matrix, and

understood in set-theoretic terms as a set of attribute-value pairs (Dalrymple 2001, 30).

So, for the English sentence *Spot runs*, the c-structure can be represented as in (2), assuming for the moment standard X' theoretic structures; the f-structure for the same sentence, representing the abstract grammatical structure of the clause, can be represented as in (8).¹⁴

$$(8) \quad \left[\begin{array}{l} \text{PRED} \quad \text{'run(SUBJ)'} \\ \text{SUBJ} \quad \left[\text{PRED} \quad \text{'Spot'} \right] \end{array} \right]$$

These two levels of grammatical representation are related via the projection function ϕ , which maps c-structure nodes to corresponding f-structures. Functional descriptions (f-descriptions) constrain the possible relations between c-structures and f-structures. The relations between c- and f-structure are stated by reference to c-structure nodes, their mothers, and the f-structures projected from those nodes and their mothers. So, any c-structure node can be referred to by the variable $*$, and its mother by the variable $\hat{*}$. The f-structure projected from any c-structure node is therefore obtained by the application of the function ϕ to the variable $*$, that is $\phi(*)$, and likewise the f-structure projected from a c-structure node's mother is obtained by the application of ϕ to $\hat{*}$, that is $\phi(\hat{*})$. Reference to these f-structures is abbreviated using the metavariables \downarrow and \uparrow :

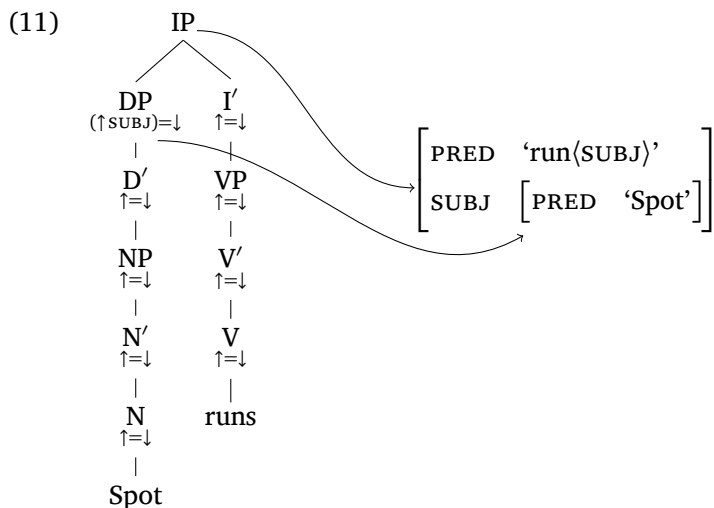
$$(9) \quad \begin{array}{l} \text{a. } \downarrow \equiv \phi(*) \\ \text{b. } \uparrow \equiv \phi(\hat{*}) \end{array}$$

Using these metavariables it is possible to concisely state constraints on the relation between c-structure and f-structure. For example, in English the specifier of IP is associated with the grammatical role of subject. The following PSR captures this constraint:

$$(10) \quad \text{IP} \rightarrow \quad \text{DP} \quad \text{I}' \\ \quad \quad \quad (\uparrow\text{SUBJ}) = \downarrow \quad \uparrow = \downarrow$$

¹⁴Following standard LFG conventions, we represent only those features of f-structure that are relevant for the discussion at hand, omitting features encoding information about person, number, gender, tense, aspect, and other grammatical information. More complex f-structures containing more features appear below, e.g. (23) and (24).

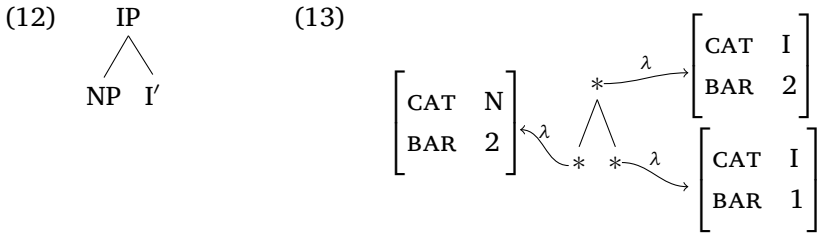
The annotation $(\uparrow\text{SUBJ})=\downarrow$ on the specifier of IP states that the f-structure projected from the DP (\downarrow) is the value of the attribute SUBJ in the f-structure projected from the DP's mother (\uparrow). The annotation $\uparrow=\downarrow$ on the I' states that the f-structure projected from the I' (\downarrow) is the same f-structure as that projected from the IP (\uparrow). Ex. (11) repeats the c-structure in (2), but augmented with the functional descriptions specified for each node in the PSRs, and shows the projection function ϕ relating the c-structure to the f-structure (from (8)) by means of arrows between the two structures.



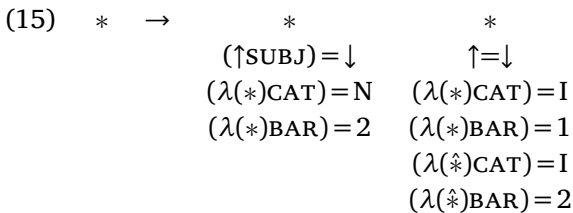
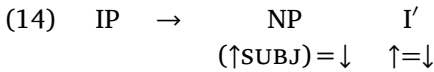
Importantly, c-structure and f-structure are not the only two levels of grammatical representation, and ϕ is not the only projection function. For example, the function σ maps f-structures to s(ematic)-structures. Kaplan (1989) generalized the concept of projection functions between levels of grammatical representation, resulting in a 'projection architecture' of different levels of linguistic structure. Much recent work has debated the full inventory of projections and projection functions, including e.g. Bögel *et al.* (2009), Dalrymple and Mycock (2011), Dalrymple and Nikolaeva (2011), Giorgolo and Asudeh (2011), Asudeh (2012, 53), and Mycock and Lowe (2013).

For our purposes, the details of the projection architecture are not important. But one additional projection is vital to the present discussion. While c-structure representations standardly incorporate

information on category labels and projection level in representing nodes as IP, N', V, etc., this is to be understood as a shorthand. Following Kaplan (1989), category information and projection level are not directly encoded in c-structure, but are projected from c-structure nodes via a projection λ . That is, the representation in (12) must be understood as a shorthand for something like (13). We refer to the structure projected by λ as the l-structure. Note that the l-structures in (13) are for illustrative purposes only; the feature BAR is not an element of the analysis we propose below.¹⁵



Since projection level and category information are not actually a part of c-structure, but are projected from it just like f-structure features, it follows that projection level and category information must be constrained in PSRs by means of functional descriptions on nodes, rather than as inherent properties of nodes. For example, just as (12) is an abbreviation for (13), so the PSR in (14) can be understood as an abbreviation for something like (15); recall that * represents a phrase structure node.



¹⁵On BAR see Section 4.1 below.

Clearly, the functional descriptions specifying category and projection level in (15) are highly inadequate, and fail to capture most or all of the desiderata for a formal model of phrase structure as set out above. In particular, the feature BAR with values 0, 1, 2, does no more than model the X'-theoretic distinction between X, X' and XP, retaining all the problems with these notions discussed above.

Our proposal goes beyond the basic assumptions in (13) in two major ways; the first of these will be discussed in this section, the second in Section 3.3. Firstly, we propose that a relatively minor alteration of the feature set seen in (13) is sufficient to license a model of phrase structure which incorporates most of the desiderata set out above. We propose three 1-structure features instead of two: CAT, which represents category labelling just as in (13); L, which intuitively represents the 'level' of any node, roughly corresponding in traditional terms to whether the node is a zero, one or two bar level node; and P, which intuitively represents the maximum projection level of the word/projection concerned.

$$(16) \quad * \xrightarrow{\lambda} \begin{bmatrix} \text{CAT} & \text{V} \\ \text{L} & 0/1/2 \\ \text{P} & 0/1/2 \end{bmatrix}$$

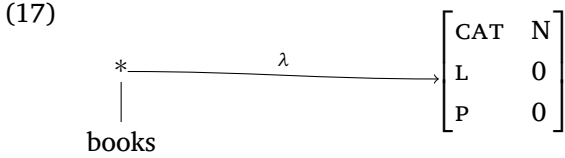
The values of L and P are integers, e.g. 0, 1, 2.¹⁶ We assume that the value 2 is a sufficient maximum for English, but our formalization below does not enforce either a maximum or minimum value, meaning that if higher values are justified for some phrase types in some languages, or if some phrase types require only two values, 0 and 1 (for example because they lack specifiers), this will fall out unproblematically without further stipulation.

In order to make our proposal as clear as possible, we illustrate the 1-structures we assume for the phrases *books*, *the book*, and *Bill's books*. However, the 1-structure relations indicated here are not yet final, because we have not yet discussed our second innovation over (13); in

¹⁶ But see footnote 21.

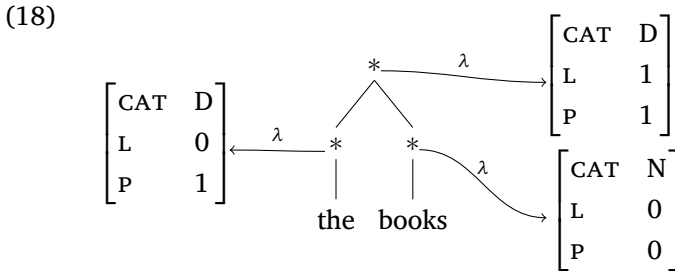
order to simplify the presentation, we integrate that into our model separately, in Section 3.3.

The phrase *books* in the sentence *I read books* will have the following structure:



As a phrase consisting of a single word, *books* is both maximal and minimal. In our system, the definition of a minimal projection is any node with the feature $\langle L,0 \rangle$, while the definition of a maximal projection is any node with the feature set $\{\langle L,n \rangle, \langle P,n \rangle\}$, that is any node whose L and P features have identical values. A node which is both maximal and minimal therefore has the feature set $\{\langle L,0 \rangle, \langle P,0 \rangle\}$.

The phrase *the books* in the sentence *I read the books* will have the following (preliminary) structure:

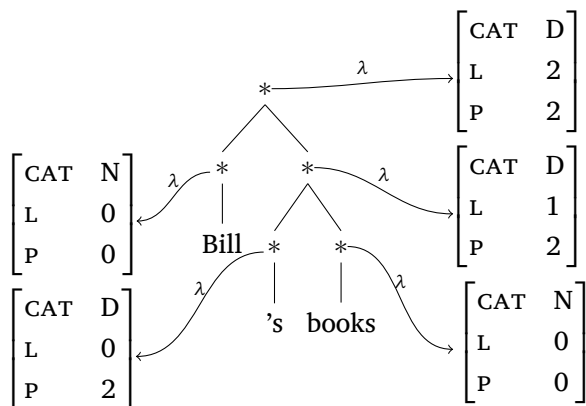


Once again, the noun *books* is both maximal and minimal as the noun phrase complement of D. The head D is a minimal projection, so has the feature $\langle L,0 \rangle$, but it is not maximal. The maximal projection of the determiner phrase is the node that directly dominates the D head and the N complement. Since there are only two words in the phrase, we require only a single projection up from the preterminal nodes, just as in a BPS analysis. The maximal projection is one projection level up from the head; it therefore has the feature $\langle L,1 \rangle$. As a maximal projection, its L and P values must be identical; it therefore also has the feature $\langle P,1 \rangle$. The feature P represents the maximal projection level for the entire projection, and is shared by all nodes in the projection chain. Thus as the head of the determiner phrase, the head D must

have the same P value as the maximal projection, meaning that it also has the feature $\langle P,1 \rangle$.

Now, consider the phrase *Bill's books*. Let us assume (purely for the sake of argument) that the possessive marker 's is a separate word which fills the head of the determiner phrase, and that *Bill* appears in the specifier of the determiner phrase.

(19)



Once again the noun *books* is simultaneously maximal and minimal, and the same is true of the other noun in the phrase, *Bill*. But now the DP consists of three words, and thus necessarily has more structure. Since there is both a specifier and complement to D, the maximal projection is two projection levels higher than the head, and therefore has the feature set $\{\langle L,2 \rangle, \langle P,2 \rangle\}$. The head, as a minimal projection, has the feature $\langle L,0 \rangle$, and since the maximal projection from the head has the feature $\langle P,2 \rangle$, the head also has this feature. The intermediate node is one projection up from the head, and is part of a projection chain which extends two levels of projection above the head (i.e. which has the feature $\langle P,2 \rangle$); the intermediate node therefore has the feature set $\{\langle L,1 \rangle, \langle P,2 \rangle\}$.

Sets and distributive features

3.3

Although the system illustrated in the previous section enables us to formalize an approach to phrase structure which eliminates non-branching dominance chains, and achieves several of the other desiderata set out above, it nevertheless incorporates a degree of

redundancy, particularly as regards the CAT and P features. Essentially, in any projection chain the values for CAT and P for every node are identical, as e.g. with the three l-structures projected from the head, intermediate and maximal D projections in (19). It is possible to stipulate this identity, by means of constraints which require the head daughter of any node to have the same CAT and P values as its mother. But as discussed above, it would be preferable if the necessarily shared properties of such nodes were shared as a natural consequence of the model (as in BPS), rather than by stipulation (as in X' theory).

Happily, the LFG framework provides the mechanism we seek. L-structures are represented as attribute-value matrices, and just like f-structures, as discussed above, are understood in set-theoretic terms as sets of attribute-value pairs. It is also possible, and sometimes necessary, to assume sets of f-structures, that is sets of sets of attribute-value pairs. By extension, sets of l-structures are formally unproblematic.

Features (or attributes) interact with sets of f-structures in interesting ways, such that it becomes necessary to distinguish two types of features, *distributive* and *non-distributive* features. The need for this distinction has been most clearly demonstrated in relation to coordination and agreement; we therefore take a small detour to justify the difference between distributive and nondistributive features, before demonstrating their use for the present topic.

3.3.1 Agreement and (non)distributive features

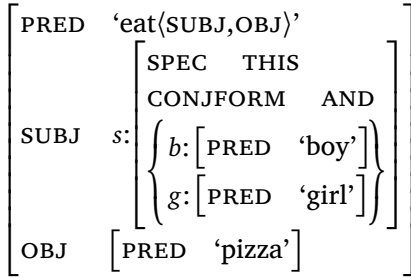
Consider the following data, based on King and Dalrymple (2004):

- (20) a. This boy and girl eat/*eats pizza.
b. *These boy and girl eat/eats pizza.
c. A boy and girl eat/*eats pizza.
d. *This boy and girls eat/eats pizza.

In English, a single determiner can occur with two conjoined singular nouns, and in this case the determiner must be singular. Yet the verb agreement with such a subject phrase must be plural. In LFG, coordinated phrases are analysed at f-structure as a set, whose members are the f-structures of the individual coordinated phrases. It is also possible for sets to have their own features, independent of the f-structures they contain; for example, a conjunction provides a feature

such as $\langle \text{CONJFORM,AND} \rangle$, but this feature is a feature of the whole conjoined phrase, not of either (or both) of the embedded phrases. So for the sentence *this boy and girl eat pizza* the f-structure will look something like this:

(21) This boy and girl eat pizza.



The structure labeled *s* is a *hybrid set*: it is a set containing both individual attribute-value pairs (features) and f-structures. The representation of *s*, with square brackets enclosing the features and braces enclosing the f-structures, is potentially misleading. It is not the case that the set of f-structures $\{b, g\}$ is contained within and distinct from *s*, but the square brackets and braces together identify the hybrid set *s*, which contains four elements: two features ($\langle \text{SPEC,THIS} \rangle$ and $\langle \text{CONJFORM,AND} \rangle$), and two f-structures (*b* and *g*).

In order to deal with the simultaneously singular and plural agreement of the conjoined noun phrase, King and Dalrymple (2004) adopt the proposal of Wechsler and Zlatić (2003) that there are actually two types of agreement feature for nouns: *CONCORD* and *INDEX* features. Informally, *CONCORD* is more morphological, and is generally relevant for agreement between nouns and their immediate specifiers and modifiers (e.g. determiners and adjectives). On the other hand, *INDEX* is more semantic, and is relevant for agreement outside the noun phrase, e.g. verb agreement.

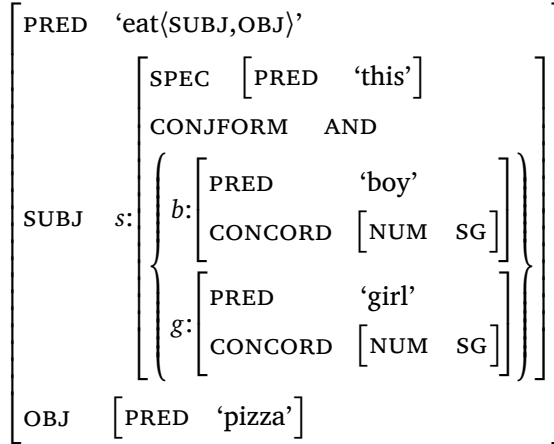
Singular *this*, *boy* and *girl* specify both their *CONCORD NUM* and *INDEX NUM* as *SG*, while plural *these*, *boys* and *girls* specify their *CONCORD NUM* and *INDEX NUM* as *PL*. This is sufficient to account for the grammaticality/ungrammaticality of *this boy/these boys/*this boys/*these boy* etc. But to account for the grammaticality of *this boy and girl*, and the ungrammaticality of **these boy and girl*, we now

require the distinction between distributive and nondistributive features. Distributive features are defined as follows (Dalrymple and Kaplan 2000):

- (22) If a is a distributive feature and s is a set of f-structures, then $(s a = v)$ holds iff $(f a) = v$ for all f-structures f which are members of s .

Informally, a nondistributive feature may hold of a set of f-structures (making the set a hybrid set) independently of whether it holds of each or any of the members of that set. In contrast distributive features cannot hold of a set independently, but must hold for every member of the set. If CONCORD agreement features are *distributive*, then any CONCORD feature specified of a set must hold of all f-structures within that set. So when *this* modifies two conjoined nouns, and hence maps to a set of f-structures, its specification $(\uparrow \text{CONCORD NUM}) = \text{SG}$ holds only if all f-structures within the set have the feature $\langle \text{CONCORD NUM,SG} \rangle$.

- (23) This boy and girl eat pizza.

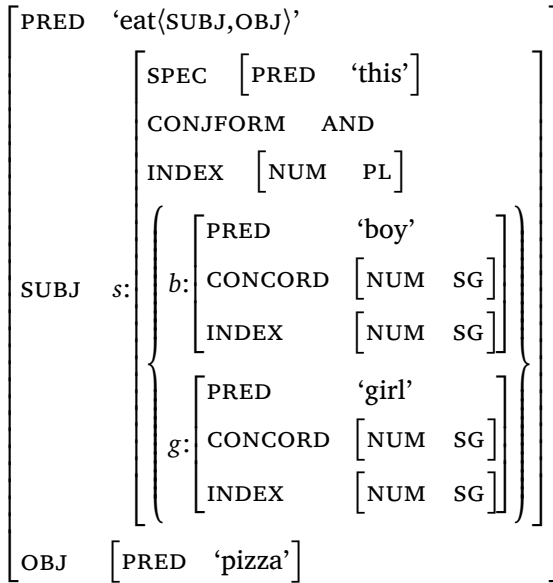


Correspondingly, **these boy and girl* is ruled out because *these* will require every member of its set to have the feature $\langle \text{CONCORD NUM,PL} \rangle$, which will not be compatible with the singular concord of the nouns. Singular or plural determiners with nouns of mismatched number, e.g. **this boy and girls*, are also ruled out, since the definition of distributivity requires that when a distributive feature is applied to

a set, *every* member of that set must necessarily have the same value for that feature.

As for verb agreement, this depends on INDEX. INDEX is a *non-distributive* feature. Any non-3SG present tense verb specifies that the value of its SUBJ INDEX NUM is PL, or else that the value of its SUBJ PERS is not 3; only the first disjunction is relevant here. If the subject is an ordinary, non-conjoined noun phrase, then the noun must be plural (since plural nouns specify their INDEX NUM as PL, while singular nouns specify it as SG, as discussed above). If the subject is a set, then the feature ⟨INDEX NUM,PL⟩ must hold of the set, but *need not* hold of any of the members of the set. Thus *s* has the feature ⟨INDEX NUM,PL⟩, which is different from the INDEX NUM feature of the members of *s*. This is exactly what we require to account for sentences like (20a):

(24) This boy and girl eat pizza.



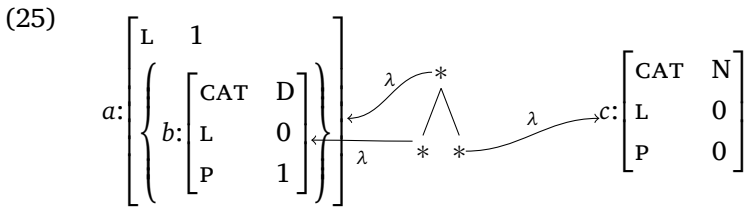
Back to phrase structure

3.3.2

How does the difference between distributive and nondistributive features help with modelling projection chains? Although, in coordination, sets of f-structures are necessarily sets of more than one f-structure, it is of course also possible to have singleton sets, i.e. sets

containing a single member.¹⁷ Now, if a distributive feature applies to an f-structure, or l-structure, which is a singleton member of a set, that feature necessarily holds of the set as well. Likewise, if a distributive feature is specified of a singleton set, it necessarily holds of the member of that set.¹⁸

Now let us revisit the projection structure for the phrase *the books*. In (18) we treated the three l-structures projected from the three nodes as structurally independent of each other. But now let us assume that in any projection chain the l-structure of the head daughter is contained within the l-structure of the mother, the mother's l-structure therefore being a hybrid set. The intuition we are trying to model is that CAT and P values are necessarily identical for any node in a projection chain.¹⁹ If projection chains are modelled using set inclusion, then we can achieve the desired outcome simply by defining the relevant features as distributive. So instead of (18), we now propose:



That is, if CAT and P are distributive features, and if the l-structure for any head daughter is a member of the (hybrid, singleton) set that constitutes the mother l-structure, then CAT and P features are *necessarily* shared between any mother and head daughter. This means we require no stipulation to ensure that, say, a head of category D projects a phrase of category D: the distributive nature of the CAT

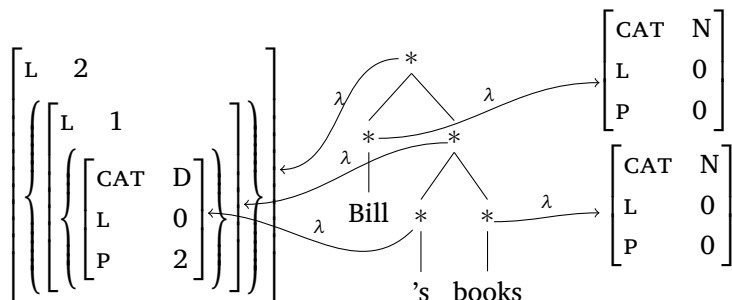
¹⁷ This is a regular outcome in LFG analyses of adjunction.

¹⁸ Recently, Andrews (2018) has explored the potential of singleton hybrid sets at f-structure for dealing with long-standing problems of scope in LFG, and our proposal is inspired by his work.

¹⁹ We do not address coordination in this paper, but note that coordination of unlike categories is unproblematic, as we do not need to assume that set inclusion holds between coordinated nodes and their mother. To deal with unlike categories will require a more complex representation of categories, such as that proposed by Dalrymple (2017), which is entirely compatible with the model proposed here.

feature and the nature of l-structure inclusion enforces this. The feature L, of course, must be defined as nondistributive, since mothers and daughters in a projection chain may have different values for this feature. Set inclusion can be recursive, so the principles illustrated in (25) will equally well account for a phrase which projects two levels (or more) above the head, as in *Bill's books*:

(26)



Phrase structure rules and templates

3.4

In the previous section, we showed the desired outcome of our model. Now the question is how to state the relevant constraints which will realise that model. The constraints which derive l-structure values are realised as functional descriptions on PSRs and in lexical entries, i.e. the standard locus of constraints in LFG.

We require a fixed number of f-descriptions to model l-structure, which occur in different combinations in different contexts; in order to generalize over multiple instances of these f-descriptions, we define them as *templates* (Dalrymple *et al.* 2004; Asudeh *et al.* 2013); templates function like macros, allowing the same combinations of f-descriptions to be applied together wherever appropriate. For example, some projections require that the L and P values for a particular node are identical (i.e. a maximal projection), others require that the L value for a particular node is identical to the mother node's L value. We assume the following basic templates:²⁰

²⁰These templates use an alternative representation for projection functions from that introduced above: *_λ is the same as λ(*).

(27) Basic templates:

a. L-structure inclusion	$LSTRIN \equiv *_{\lambda} \in \hat{*}_{\lambda}$
b. Maximal phrase	$LP \equiv (*_{\lambda} L) = (*_{\lambda} P)$
c. Mother node is a maximal phrase	$LPM \equiv (\hat{*}_{\lambda} L) = (\hat{*}_{\lambda} P)$
d. L of node = L of its mother	$LUD \equiv (\hat{*}_{\lambda} L) = (*_{\lambda} L)$
e. L of mother node = 1	$LIM \equiv (\hat{*}_{\lambda} L) = 1$
f. L is one less than L of mother	$LDOWN \equiv (*_{\lambda} L)$ $= (\hat{*}_{\lambda} L) - 1$
g. L = 0	$LO \equiv (*_{\lambda} L) = 0$
h. L of mother node = 0	$LOM \equiv (\hat{*}_{\lambda} L) = 0$
i. Mother node has a P value	$PXM \equiv (\hat{*}_{\lambda} P)$
j. Node does not have a P value	$PNX \equiv \neg(*_{\lambda} P)$
k. Mother does not have P value	$PNXM \equiv \neg(\hat{*}_{\lambda} P)$

The first template here, LSTRIN, defines the l-structure inclusion relation: the l-structure of the current node is a member of the l-structure of the mother of the current node (the latter l-structure by consequence therefore being a set). Other templates refer directly to L and P values: they either specify that two features have the same value, or specify an absolute or relative value for a particular feature, or state existential constraints on the feature P.

The template LDOWN specifies a relative value for L: the value of L of the current node is one less than the value of L of the mother node. This crucial template is what drives the increase/decrease of L values up/down a projection chain. Note that technically natural numbers play no role in the LFG formalism; feature values like 0, 1, 2, are symbols, not natural numbers, so mathematical statements like $L - 1$ are not strictly possible. It is, however, unproblematic to formalize addition/subtraction using the successor function, and we retain the mathematical statement as in (27f) for readability.²¹

²¹ In Lovstrand (2018, 153) the @LDOWN template is defined as: @LDOWN $\equiv (\hat{*}_{\lambda} L PLUS) = (*_{\lambda} L)$. In this approach, the value of L is either 0 or an attribute-value matrix with the attribute PLUS. In the l-structure, what is informally rep-

The constraint in (27i) requires the feature P to exist in the l-structure of the mother node; PNX requires that P does not exist as a feature of the l-structure of the current node, and PNXM requires the same of the mother's l-structure. These existential constraints are required to account for nonprojecting categories, as discussed in Section 3.6.

The constraints in (27) are the only constraints needed to model the phrase structure of natural language. Given these, and only these, constraints, certain features of the system fall out unproblematically. For example, in our system, intuitively, for any l-structure the value of L is never greater than the value of P: $\forall *_{\lambda}, P \geq L$. Given only the templates in (27), an l-structure that violates this intuitive general constraint cannot be generated, so the constraint need not be independently stated.

Common phrase structure positions require particular combinations of the constraints in (27). We therefore define further templates for convenience, which call combinations of the templates in (27).

(28) Complex templates:

a. Head of an endocentric projection:

$$\text{HEADX} \equiv @L\text{DOWN} \wedge @L\text{STRIN}$$

b. Head of an adjunction structure:

$$\text{HEADA} \equiv @L\text{UD} \wedge @L\text{STRIN}$$

c. Specifier or adjunct: EXT $\equiv @L\text{PM} \wedge @L\text{P}$

d. Complement: INT $\equiv @L\text{IM} \wedge @L\text{P}$

e. Nonprojecting node: NONPRJ $\equiv @L\text{O} \wedge @P\text{NX}$

f. Nonprojecting mother: NONPRJM $\equiv @L\text{OM} \wedge @P\text{NXM}$

g. Projecting mother: PRJM $\equiv @L\text{OM} \wedge @P\text{XM}$

HEADX applies to heads in specifier and complement structures, HEADA applies to heads in adjunction structures. EXT and INT apply to specifier/adjunct phrases and complement phrases respectively. We can now rewrite the standard schematic PSRs of X' theory in our system:

resented as the number 1 is formally represented as [L [PLUS 0]], the informal number 2 is formally [L [PLUS [PLUS 0]]], and so on.

(29) Schematic phrase structure rules:

- a. Specifier rule: * → * , *
 @EXT @HEADX
- b. Complement rule: * → * , *
 @HEADX @INT
- c. Adjunction rule: * → * , *
 @HEADA @EXT

Notice the generality of these rules with respect to category sharing. There is no need for category label to be specified on the left-hand side of a rule (or indeed on the right-hand side), because the category of the mother automatically follows from the category of the head daughter (by the constraint LSTRIN called by the templates HEADX and HEADA). In other words, once the head of an endocentric structure is identified by its template, there is no further need to stipulate what the category of the mother node is. However, this differs from exocentric structures, where the category of the mother node may need to be specified as an additional constraint on one of the daughters. Given this explicit formal restriction on the category of the mother node in our approach, the left-hand side of traditional PSRs, and the arrow, are redundant; we could equally well rewrite (29) as:²²

(30) Schematic phrase structure constraints:

- a. Specifier structure: [* , *]
 @EXT @HEADX
- b. Complement structure: [* , *]
 @HEADX @INT
- c. Adjunction structure: [* , *]
 @HEADA @EXT

Such a representation accords more closely with the constraint-based conception of LFG, which interprets PSRs not as procedural rules, but as constraints on possible structures.

²²The square brackets in (30) serve to indicate the left and right edges of the relevant constituents.

As an illustration of our model, we give the necessary phrase structure constraints and lexical entries to derive the sentence *Bill read a book of poems*. In these constraints, we specify category labels on the right-hand side in the traditional way, but this is to be understood as a shorthand for an f-description defining the CAT value of the relevant node's l-structure.

(31) Phrase structure constraints:

- a. [{N|D} { I | V }]
 (↑SUBJ)=↓ ↑=↓ ↑=↓
 @EXT @HEADX @INT
 (*_λ CAT) = I
- b. [V {N|D}]
 ↑=↓ (↑OBJ)=↓
 @HEADX @INT
- c. [D N]
 ↑=↓ ↑=↓
 @HEADX @INT
- d. [N P]
 ↑=↓ (↑OBL)=↓
 @HEADX @INT
- e. [P {N|D}]
 ↑=↓ (↑OBJ)=↓
 @HEADX @INT

The constraint in (31a) equates to a traditional specifier rule for IP; it is formulated so as to license optionality of the functional head I (notice that optionality is not a default, but has to be specifically licensed in this way). The constraint in (31b) equates to the complement rule for VP; that in (31c) equates to the complement rule for DP;

(31d) is the complement rule for NP, and (31e) is the complement rule for PP.²³

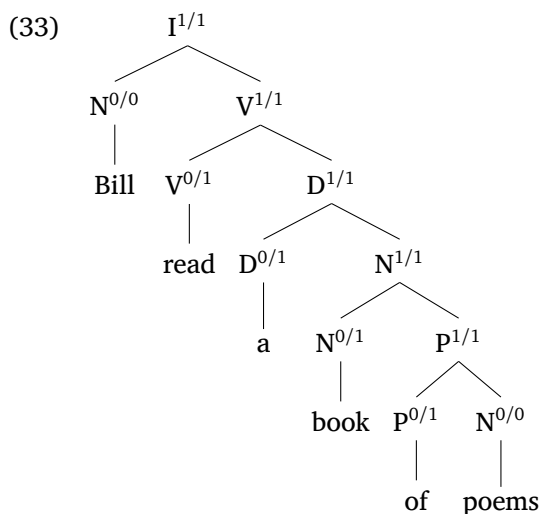
The PSRs in (31), together with the lexical entries in (32), produce the phrase structure in (33). Although we understand the features CAT, L and P as features within the ‘l-structure’ projected from a node, for ease of representation in trees such as (33), we propose an abbreviatory notation whereby L and P values are shown as superscripts on category node labels. Each node, represented by its category label, appears with superscript numbers separated by a slash. The first number represents the L value, the second the P value for that node. So, a node which has the features $\langle \text{CAT}, V \rangle$, $\langle L, 0 \rangle$ and $\langle P, 1 \rangle$, will be represented as $V^{0/1}$.

(32) Lexical entries:

- a. Bill: N
(\uparrow PRED) = ‘Bill’
@PRJM
- b. book: N
(\uparrow PRED) = ‘book<OBL>’
@PRJM
- c. poem: N
(\uparrow PRED) = ‘poem’
@PRJM

²³Note that we adopt a simplified approach to category labels in this paper, treating N and D as fully distinct labels, but the rules provided here imply a more sophisticated approach, following e.g. Grimshaw (1991) and Bresnan (2001). We assume that in fact N and D share the same category label N, but are distinguished in terms of another feature $\pm F$. The value of F may be specified in a given rule or not; so in (31a), $\{N|D\}$ is really to be understood as N with underspecified value for F; but in (31c), which constrains the structure within a determiner phrase, the $+F$ value of the head, and the $-F$ value of the non-head, are crucial elements of the rule. The underspecification of certain nodes improves the resulting analyses by eliminating the need for certain nonbranching nodes. For example, with the subject position in (31a) underspecified, both *This* and *Bill* can serve as single word subject phrases requiring only a single c-structure node, $D^{0/0}$ in the former case, $N^{0/0}$ in the latter. For the present purposes, so as not to further complicate our presentation, we abstract away from the details of this, and present our analysis as though N and D are fully distinct categories, modelling the underspecification via optionality.

- d. read: V
(↑PRED) = 'read{SUBJ,OBJ}'
@PRJM
- e. a: D
(↑SPEC) = 'a'
@PRJM
- f. of: P
(↑PRED) = 'of{OBJ}'
@PRJM



L and P values are determined 'bottom up'. So *poems* attaches to a node $N^{0/0}$, since there are no higher levels of projection in this phrase. In contrast, *book* attaches to a node $N^{0/1}$, since there is one level of projection above the head; the word *read* attaches to a node $V^{0/1}$, since there is one level of projection within the verb phrase. The L value is determined from the bottom up, with all words specifying $L=0$ of their preterminal node. The head of an X' -theoretic projection is associated with the template LDOWN (via the template HEADX), meaning that every mother node in a headed projection chain has L value one greater than that of its head daughter.

The P value is determined by the number of projection levels in the phrase. All maximal projections are associated with the template LP, meaning that the P value for every maximal node is iden-

tical to the value of L for that node. So in a two-level projection, where the preterminal head daughter has the feature $\langle L,0 \rangle$ and the mother therefore has the feature $\langle L,1 \rangle$ (by L_{DOWN}), the value of P for the mother node will be the same as its L feature, i.e. 1. The inclusion relation specified for the L -structures of heads in a projection chain ensures that all nodes in any projection chain automatically and necessarily share the same value for CAT and P , as discussed above.

Regarding the top node, the constraint in (31a) licenses an I node with specifier and complement, but no head, daughters. This models the headless²⁴ IP structure standardly assumed in LFG for clauses without auxiliaries, but without requiring nonbranching projections. Only maximal projections ($L = P$) can have specifier daughters (as constrained by the template EXT); only nodes with the feature $L = 1$ can have complement daughters (as constrained by the template INT); the top node must therefore be $I^{1/1}$, satisfying both constraints simultaneously.²⁵

3.6

Dealing with nonprojecting categories

As discussed in Section 2.2, no existing formalization of phrase structure adequately accounts for the existence of nonprojecting categories. Following Arnold and Sadler (2013), we model the difference between prenominal and nonprenominal adjectives in English in these terms: prenominal adjectives, which cannot take complements or other postmodifiers, and hence appear not to be able to head full phrases, are treated as nonprojecting adjectives, while adjectives in other positions (predicative or predicated) can head full phrases and so are projecting.

²⁴Headless, but not exocentric, as the IP serves as the extended projection of the V .

²⁵There is a partial parallel here between our approach and the exocentric treatment of CP by Jayaseelan (2008) and Putnam and Stroik (2009, 2010); our treatment of headless CP structures, which we do not have space to discuss here, would fully parallel the approach to headless IP s set out here, and would thus be very close to these exocentric treatments of CP . An alternative to the headless IP assumed here would be to adopt the older analysis of an exocentric clausal node S , as assumed e.g. in HPSG (Pollard and Sag 1994) and in LFG by Bresnan *et al.* (2016).

Many adjectives in English can appear in both prenominal and other positions, e.g. *small*, and for such cases we assume that the grammar licenses both variants; some adjectives are restricted to one or the other position, however, and we analyse this by assuming that such adjectives have only nonprojecting (e.g. *former*), or only projecting (e.g. *asleep*), variants.

- (34) a. The small dog eats biscuits.
b. The dog is small.
c. The former president eats biscuits.
d. *The president is former.
e. *The asleep dog eats biscuits.
f. The dog is asleep.

Our model fully captures the grammaticality judgments in (34). The l-structure feature set of a nonprojecting category must be fully distinguishable from the possible feature sets available to projecting categories. For example, one might think that as a necessarily minimal and maximal projection, a nonprojecting category should necessarily have the features $\{\langle L,0 \rangle, \langle P,0 \rangle\}$ (as is assumed for clitics within BPS). However, this is a possible feature set for projecting words, whenever they happen to appear alone constituting a phrase. We must therefore allow adjectives in predicate position to have the features $\{\langle L,0 \rangle, \langle P,0 \rangle\}$, so this feature set cannot be attributed to nonprojecting adjectives, otherwise we would not be able to prevent e.g. *former* from appearing in predicate position.

We propose that as necessarily minimal projections, nonprojecting adjectives have the feature $\langle L,0 \rangle$, but that as categories which necessarily do not project, they have no value for the feature P. This is the purpose of the templates PNX and PNXM in (27j–k). The lexical specification for a nonprojecting word includes the template PNXM (called by NONPRJM), which ensures that the preterminal c-structure node dominating the word lacks the feature P. The template PNX appears in PSRs (called by NONPRJ) on nodes which are restricted to nonprojecting categories.

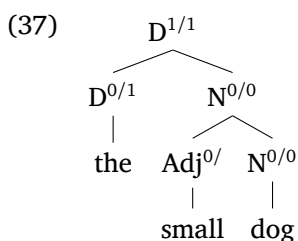
We thus assume the following lexical entries for *small*, *former*, and *asleep*:

(35) Lexical entries:

- a. small Adj
 (↑PRED) = ‘small’
 { @PRJM | @NONPRJM }
- b. former Adj
 (↑PRED) = ‘former’
 @NONPRJM
- c. asleep Adj
 (↑PRED) = ‘asleep’
 @PRJM

The constraints in (36) license predicate adjectives and prenominal adjectives. (36a) defines a standard complement structure, and therefore the Adj complement has the specification LP (called by @INT), meaning that nonprojecting adjectives cannot stand in predicate position. (36b) requires that a prenominal adjective lack a feature P, thereby restricting the prenominal position to nonprojecting adjectives. A tree illustrating a noun phrase with nonprojecting adjective is given in (37).

- (36) a. Predicate adjective: [I Adj]
 @HEADX @INT
- b. Non-proj. adjunction: [Adj N]
 @LOM @HEADA
 @NONPRJ

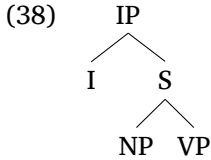


3.7

Exocentric categories

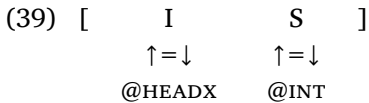
As discussed in Section 2.2, exocentric projections are another widely accepted possibility in LFG which have nevertheless never been adequately formalized within a theory of phrase structure.

Our proposal enables a neat and insightful analysis of exocentricity. For the purposes of illustration, we adopt the analysis of Welsh proposed by Sadler (1997), which involves the following basic clause structure (stated in traditional, X'-theoretic, terms):

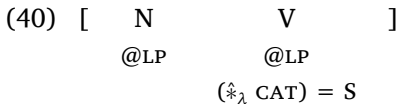


The clause-initial finite verb, often an auxiliary, appears in I, and the complement of I is an exocentric phrase which includes both the subject phrase (the NP in (38)) and the VP (often containing the lexical verb, and any object, etc.).

In our model, S will be licensed as a complement daughter of I; the functional constraints placed on S in the PSR will be fully parallel to those placed on any other complement, so the template INT will apply to the S node:

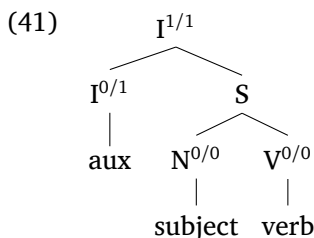


The template INT calls the templates LIM and LP. The first specifies the value of L for the mother node, while the second requires that the values of L and P for the S node be identical. Now consider the rule that introduces the daughters of S. Since S is exocentric, no daughter of S is the head, nor is any daughter a specifier, a complement, or even an adjunct; therefore none of the standard endocentric templates above apply to any of the daughter nodes. The daughters of S may themselves be specified as necessarily projecting, but no daughter need make any specification about the L/P values of S.



When we try to construct a tree based on these rules parallel to (38), it is impossible to assign values to S for its L and P features. As a complement of I, S must satisfy the requirement L = P, and in the

absence of specific values, this can only be satisfied if neither value exists. That is, we get the following:



Since S is an exocentric category, its daughters lack the typical endocentric specifications introduced above. The result is that S lacks L/P values. Since these features are used to define and constrain endocentric projections, we take this to be an intuitive definition of exocentricity: exocentric categories lack L/P features.²⁶

3.8

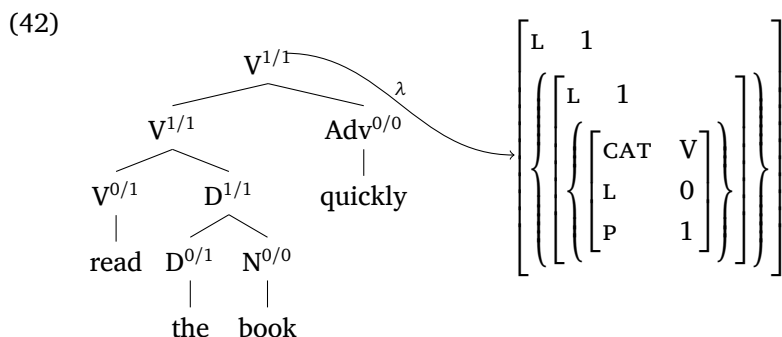
Comparison with traditional X' theory

In (1) we gave seven desiderata for a formal model of phrase structure. All seven are achieved by our model. The use of two features with numerical values, L and P, enable us to define PSRs in such a way that nonbranching chains are eliminated: a node can be both maximal and minimal at the same time, and more complex phrases have only the nodes required to model constituency. Our proposal also eliminates the need for default optionality in PSRs, as standardly assumed in LFG. Standard LFG takes optionality to be a default, because in any projection heads (particularly, but not only, functional heads), specifiers and complements may be absent. In our model, however, optionality is an exception rather than the rule: if a phrase lacks a complement and/or

²⁶This conclusion is not without complication. According to the formulation of Kaplan and Bresnan (1982), a constraint such as $L = P$ is violated if neither L nor P are assigned a value, and thus a derivation based on (39) and (40) would fail. On the other hand, in the computational implementation of LFG, XLE (Crouch *et al.* 2011), the constraint $L = P$ is satisfied if $\neg L \wedge \neg P$. The rationale for Kaplan and Bresnan's approach is not clear to us; it has been suggested to us (Adam Przepiórkowski, p.c.) that the theory could be unproblematically emended to fall in line with XLE, and we adopt that emendation here.

specifier, the PSRs introducing those positions are simply not utilized, and a simpler structure results.

Our definition of a maximal phrase, $L=P$, avoids the problem raised by Fukui (1986) regarding the ambiguity of the label ‘XP’, since a maximal phrase may be e.g. $X^{0/0}$, $X^{1/1}$ or $X^{2/2}$. At the same time, our approach to projection chains, involving inclusion of l-structures, avoids the ambiguity between X^{max} and the top node of a projection chain noted by Hornstein and Nunes (2008). Consider the following VP:



Both the top node of the VP and its head daughter are maximal nodes in the sense defined above (being $V^{1/1}$), but the top node is distinct (and therefore distinguishable), because its l-structure alone is not included within another l-structure. Thus the top node in any projection satisfies the equations $(*_\lambda L) = (*_\lambda P)$ and the negative existential constraint $\neg(\in *_\lambda)$ (in words: there is no set of which my l-structure is a member), whereas other maximal nodes in a projection satisfy only the first.

Our proposal also lacks any distinct notion comparable to X' . Suppose we wanted to define adjunction to X' , i.e. adjunction of phrases closer to the head than any specifiers, but further from the head than any complements. A head which has a complement is, in our system, either 0/1 or 0/2 (depending on whether there is also a specifier). So nodes with the L/P values 0/1 and 0/2 must be excluded from the set of nodes to which X' adjuncts could adjoin. But a head which has a specifier, but no complement, and to which we might therefore wish to permit X' adjunction, will in our system be 0/1. Thus 0/1 nodes sometimes correspond to the size of an X' and sometimes do not. Therefore,

a notion equivalent to X' adjunction is unformalizable in our system, because there is no coherent set of L/P values which correspond to the traditional notion of X' .

Given our set-inclusion approach to projection chains, our model also reduces redundancy in category labelling and in specification of P values; that a phrase necessarily has the same values for CAT and P as its head is a necessary consequence of the model, requiring no additional stipulation. As shown in the previous sections, our model also affords principled accounts of nonprojecting categories and exocentric categories, which are lacking in existing formalizations of phrase structure.

4 OTHER PROPOSALS

In this section, we discuss three alternative formalizations of phrase structure, two within LFG and one within Minimalism. These approaches are simpler alternatives to the model presented above, in the sense that they use fewer formal features. However, their relative simplicity comes at the cost of failing to meet the theoretical desiderata laid out in (1), and incomplete coverage of attested phrase structure types.

4.1 *Bresnan (2001)*

We take Bresnan (2001; unmodified in Bresnan *et al.* 2016) as representative of standard assumptions regarding the formal properties of phrase structure in LFG.²⁷ Bresnan (2001, 100) describes the formal properties of c-structure nodes thus: “Formally, X' categories can be analyzed as triples consisting of a categorical feature matrix, a level of structure, and a third, privative feature F, which flags a category as ‘function’ (F) or unspecified as to function (lexical).” The “level of structure” feature, which we call BAR following Andrews and Manning (1999), has three values: 0, 1, 2. These digits each correspond

²⁷ Bresnan’s decomposition of syntactic categories builds on earlier work, e.g. King (1995).

to a level of structure which is represented notationally using the traditional X-bar symbols: X^0 for [BAR 0], X' for [BAR 1], and XP for [BAR 2]. The use of integers in this context implies that, in an endocentric projection, a mother must have a BAR value higher than its daughter. The question of dominance is not discussed formally by Bresnan, but the familiar templatic description of X-bar principles (43) makes it clear that some undefined, and presumably stipulatory, mechanism is intended to enforce the dominance sequence.

- (43) a. Specifier phrase structure rule
 $XP \rightarrow X', YP$
b. Complement phrase structure rule
 $X' \rightarrow X^0, ZP$

This model is distinctly simpler than the model proposed in this paper, but it has a number of shortcomings. Most obviously, the use of a single numerically valued feature to model projection levels means that Bresnan's proposal is essentially a formalization of X' theory, and thus it inherits all the failings of X' theory. There is no principled distinction between XP and X^{max} (see Lovestrand and Lowe 2017, 288–289), nor between X^{max} and the top node of a projection; there is an independent notion corresponding to X' (a node with $\langle \text{BAR}, 1 \rangle$); optionality is necessarily the default in PSRs, and by consequence non-branching dominance chains are widespread.

Bresnan (2001, 91) realises that nonbranching dominance chains are unsatisfactory, and proposes a derivational process which Dalrymple *et al.* (2015) call “ X' Elision”, to ‘prune’ unnecessary nodes from a well-formed c-structure so that it is as small as possible. As a derivational process this ‘ X' elision’ is not well integrated into the constraint-based assumptions of LFG, and although it does for the most part give the right results, it is preferable to avoid generating unnecessary nodes in the first place, as in our model, rather than generating them and then eliding them.

Bresnan's model also does not avoid redundancy in category labelling, and provides no formal account of exocentric or nonprojecting categories; the latter are not admitted in Bresnan (2001). They are admitted in Bresnan *et al.* (2016), but with no formal integration into the theory of phrase structure, which is unchanged from Bresnan (2001). There is no value of BAR which would both capture the minimality

of nonprojecting categories and would not also render them indistinguishable from zero level categories. These issues are discussed further in Lovestrand and Lowe (2017).

4.2 *Marcotte (2014)*

Marcotte (2014) simplifies Bresnan’s (2001) model by removing reference to bar levels, and thus requiring fewer theoretical primitives. Marcotte (2014, 417) explicitly likens his proposal to Chomsky’s (1995) BPS; we therefore provide a detailed comparison of this proposal with our own.

While Marcotte’s proposal can be said to reduce the number of formal devices needed to account for c-structure nodes, there are several syntactic structures that it cannot account for, and it does not meet all the desiderata set out in (1).

4.2.1 Marcotte’s proposal

Marcotte’s proposal is to remove the BAR feature, and to instead define the relationships between nodes in terms of dominance relationships and shared category features. Marcotte proposes to label what we (following Kaplan 1989) have called l-structure as “x-structure”, and assumes a function χ from nodes to x-structures, equivalent to our (and Kaplan’s) λ . The function \mathcal{M} is the function that relates the daughter node to its mother; as usual $*$ represents the node in question, and n represents any other node. There are three basic definitions of types of nodes.²⁸

(44) Marcotte’s (2014) “Bare phrase structure for Lexical-Functional Grammar”

- a. PROJECTING NODE: A node projects iff its x-structure is identical with its mother’s x-structure.

$$\text{Proj}(*) \iff \chi(*) = \chi(\mathcal{M}(*))$$

²⁸Marcotte’s model is very similar to that of Speas (1986), cf. the definition of maximal projection in footnote 13.

- b. MAXIMAL PROJECTION: A node is a maximal projection iff it is not a projecting node.

$$Max(*) \iff \neg Proj(*)$$

- c. TERMINAL: A node is a terminal iff no node has it as a mother.

$$Term(*) \iff \neg \exists n. \mathcal{M}(n) = *$$

In this system, there are four types of nodes, roughly equivalent to X^0 , X' , XP and \hat{X} . A projecting head (roughly equivalent to an X^0) is a node that meets the definitions of PROJECTING NODE (it has the same category as its mother) and the definition of TERMINAL (it is not the mother of any node).²⁹ A maximal projection (roughly equivalent to XP) is any node that meets the definition of MAXIMAL PROJECTION (it does not have a mother with identical features), and is not a TERMINAL. Intermediate nodes (roughly X') meet the definition of PROJECTING NODE, but do not meet the definition of TERMINAL (it is the mother of another node). A nonprojecting node (roughly \hat{X}) is a node that is both a MAXIMAL PROJECTION and a TERMINAL.

Marcotte applies his approach to c-structure to the structure-function principles. He provides definitions for where it should be expected to find nodes that are functional co-heads with their sister (annotated as $\uparrow = \downarrow$), and definitions for where we should expect to find subjects, objects, obliques and possessors. Notably absent is a definition of adjuncts.³⁰

(45) Marcotte (2014) “Endocentric c- to f-structure mappings”

- a. A projecting node shares the f-structure of its mother:

$$Proj(*) \implies \uparrow = \downarrow$$

- b. A SUBJ is a DP daughter of IP:

$$\begin{array}{l} Max(*) \quad Max(\mathcal{M}(*)) \\ \chi(*) = D \quad \chi(\mathcal{M}(*)) = I \end{array} \implies (\uparrow_{SUBJ}) = \downarrow$$

²⁹For Marcotte, the lexical information at the bottom of the tree is not considered a node, so his ‘terminal’ nodes are equivalent to what we (and Bresnan 2001) call preterminal nodes.

³⁰Marcotte decomposes lexical and functional categories using his own system of privative features, Pr , Tr and f such that: $V: \{Pr, Tr\}$, $A: \{Pr\}$, $P: \{Tr\}$, $N: \{\}$, $I: \{Pr, Tr, f\}$, and $D: \{f\}$. This part of his system has been simplified for readability.

- c. An OBJ is a DP with a V(P) or P(P) mother:

$$\begin{array}{l} \text{Max}(\ast) \\ \chi(\ast) = \text{D} \end{array} \quad \{V \mid P\} \sqsubset \chi(\mathcal{M}(\ast)) \implies (\uparrow\text{OBJ}) = \downarrow$$

- d. An OBL is a non-verbal/adjectival XP with a non-functional mother:

$$\text{Max}(\ast) \quad \{V \mid A \mid I\} \not\sqsubset \chi(\ast) \quad \{D \mid I\} \not\sqsubset \chi(\mathcal{M}(\ast)) \implies (\uparrow\text{OBL}) = \downarrow$$

- e. A POSS is a DP daughter of a DP:

$$\begin{array}{l} \text{Max}(\ast) \quad \text{Max}(\mathcal{M}(\ast)) \\ \chi(\ast) = \text{D} \quad \chi(\mathcal{M}(\ast)) = \text{D} \end{array} \implies (\uparrow\text{POSS}) = \downarrow$$

The first definition is the simplest. A PROJECTING NODE (any node that has the same category features as its mother) can be annotated as a functional co-head ($\uparrow = \downarrow$). A SUBJ annotation can be added to a MAXIMAL PROJECTION that has the category feature D. That node must also have a mother that is a MAXIMAL PROJECTION with the features of the category I. Likewise, the annotation for OBJ can be added to a MAXIMAL PROJECTION with the features of the category D. The mother of this node must have the feature V or P. An OBL annotation again requires a MAXIMAL PROJECTION. The node cannot be V, A or I, and its mother node must not be a functional node. Finally, for a POSS annotation, the MAXIMAL PROJECTION must have the features for the category D, and its mother must be a MAXIMAL PROJECTION with the feature for a D as well.

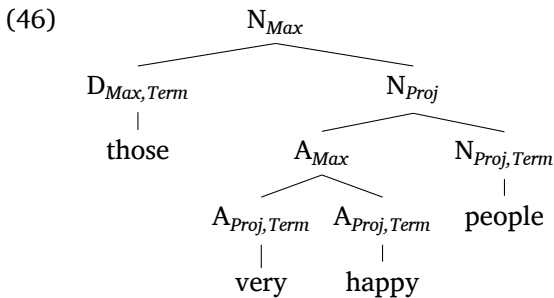
Marcotte ingeniously creates a set of structure-function association principles very similar to those proposed by Bresnan (2001), but without referring to any bar levels directly. He restricts himself to referring only to whether a node has identical category features to its mother (PROJECTING NODE) or not (MAXIMAL PROJECTION), and to what type of category features can be associated with which grammatical functions. However, there are problems with the proposal which would require significant modifications to the system in order to solve, modifications which would severely compromise the elegance of the system.

4.2.2

Issues

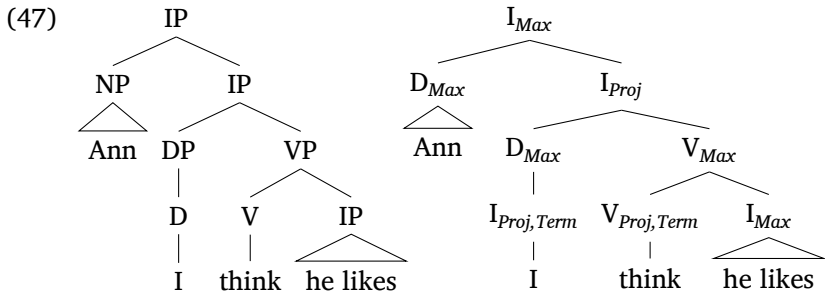
While it does have a notion of nonprojecting categories, Marcotte's (2014) model cannot deal with recursive adjunction of nonproject-

ing words to nonprojecting words, as we have assumed for nonprojecting adjectives in English, following Arnold and Sadler (2013). Ex. (46) shows a prenominal adjective structure in English translated into Marcotte’s proposed system. The point of Arnold and Sadler’s analysis is that there are a number of generalizations that can be made about prenominal modifiers in English which can be captured by analyzing those nodes as nonprojecting words. This generalization is lost in translation to Marcotte’s system. The modifiers *very* and *happy* do not meet Marcotte’s definition of nonprojecting words. According to Marcotte, a nonprojecting word is both a **TERMINAL** and a **MAXIMAL PROJECTION**. A **MAXIMAL PROJECTION** is defined as a node that does not have the same category features as its mother. In this particular structure, both modifiers have the same category as their mother, so, by definition, they are **PROJECTING NODES**, not **MAXIMAL NODES**.³¹ Another difference between the two analyses is that in Arnold and Sadler’s analysis, the node adjoined to the head noun is a nonprojecting node. This preserves their generalization that only nonprojecting modifiers can occur in this position. This generalization is lost in Marcotte’s system. The mother of the two modifiers is, by definition, a **MAXIMAL PROJECTION**, but it is not a **TERMINAL** since it has a daughter node that shares the same features. In Marcotte’s system, the mother of two nonprojecting nodes can never be a nonprojecting node itself.



³¹ This example would not be a problem if, following e.g. Payne *et al.* (2010), adjectives and adverbs were treated as part of separate c-structure categories. But the point remains valid, e.g. in the phrase *those really very happy people* the same problem would apply to the relationship between *really* and *very*.

Marcotte (2014) also fails to make the correct distinction between X^{max} and the top node of a projection, as found in XP adjunction structures. In XP adjunction, a maximal phrase (XP) is the mother of another maximal phrase of the same category features. Such a structure is not possible in Marcotte’s proposed system because, by definition, if a node’s mother has the same category features, that node cannot be a MAXIMAL PROJECTION: it is a PROJECTING NODE. Ex. (47) shows the analysis of “topicalization” from Bresnan *et al.* (2016, 196) on the left, with a translation into Marcotte’s proposed system on the right. The structure on the left has an IP in the topmost position dominating an identical IP node. On the right, in Marcotte’s system, only the topmost I node is a MAXIMAL PROJECTION. The I node below the topmost node is, by definition, a PROJECTING NODE since it shares its category features with its mother. Thus, although Marcotte (2014) does distinguish the top node of a projection from lower nodes, no node below the top node may be classified as X^{max} . This poses very real practical problems because, for example, in Marcotte’s system the position annotated for SUBJ must be dominated by a MAXIMAL PROJECTION. In the tree on the left, the DP subject is dominated by IP, so it meets the structural requirement for a subject. On the right, the D node that dominates what should be the subject is *not* the daughter of a MAXIMAL PROJECTION so it does not meet the structural requirement for a subject position; this tree, therefore, cannot be generated in Marcotte’s system.



4.2.3

Summary

Marcotte’s model is clearly an improvement on that of Bresnan (2001), capturing more of the desiderata we have been considering for a theory of phrase structure. His model eliminates nonbranching nodes, with trees that have only the structure required to model constituency.

Marcotte also avoids the need for default optionality in phrase structure rules, and lacks a distinct notion equivalent to X' . However, Marcotte's model does not correctly model the distinction between X^{max} , XP, and the top node of a projection, nor does it offer an adequate account of nonprojecting categories, nor any account of exocentric categories. It also does not avoid redundancy in category labelling.

Marcotte's proposal is in some respects very similar to that of Muysken (1982), who proposed to reformulate X' theory in terms of two binary valued features, [\pm projection] and [\pm maximal]. The weaknesses of Marcotte's model apply equally to the proposals of Muysken (1982).

Collins and Stabler (2016)

4.3

As part of their mathematically precise formalization of Minimalism, Collins and Stabler (2016, 62–66) define a labelling algorithm which they state allows natural definitions of all the X' theory concepts. Despite the fact that Collins and Stabler (2016) is one of the most complete and precise formalizations of mainstream Minimalism, their formalization of phrase structure is limited in certain respects. Given the much less flexible approach to phrase structure adopted in minimalism, some of the desiderata given in (1), notably the requirement for principled accounts of exocentricity and nonprojecting categories, are not relevant for Collins and Stabler (2016), and hence have no place in their system. As a formalization of BPS, it naturally captures most of the other desiderata. However, their system also lacks any account of adjunction, which would be crucial for a complete account of minimalist phrase structure, and does not provide any way to distinguish the highest projection from X^{max} .

Collins and Stabler (2016, 65) define a labelling function from syntactic objects to lexical item tokens, such that a. for all lexical item tokens LI, $\text{Label}(\text{LI}) = \text{LI}$, and b. for all complex syntactic objects, the label of the object is the label of its head.³² As a labelling function this is similar to LFG's λ projection, but differs in a number of ways. The

³²In some sense b. is similar to our definition of CAT above as a distributive feature in order to propagate category information automatically from head daughter to mother.

most important difference for the present purposes is that there are no distinct labels such as N and C, but it is lexical item tokens themselves which function as labels.

Given this notion of labelling, Collins and Stabler (2016) define maximal, minimal and intermediate projections:

- (48) a. For all C a syntactic object and LI a lexical item token both contained in a derivable workspace W, C is a *maximal projection* of LI iff $\text{Label}(C) = \text{LI}$ and there is no D contained in W which immediately contains C such that $\text{Label}(D) = \text{Label}(C)$.
- b. For all C, C is a *minimal projection* iff C is a lexical item token.
- c. For all syntactic objects C contained in workspace W, LI a lexical item token, C is an *intermediate projection* of LI iff $\text{Label}(C) = \text{LI}$, and C is neither a minimal nor a maximal projection in W.

They further define the complement as the first element merged with a head, and a specifier as any further element merged with a projection of the head.

The definition of maximal projection defines what we have called the highest projection, but does not allow any distinction between this and X^{max} . This distinction is relevant where adjunction to the maximal projection is admitted; since Collins and Stabler do not formalize adjunction, the failure to distinguish these notions is understandable. The definition of a minimal projection is unproblematic, and differs from the analysis presented above most significantly in that in BPS lexical item tokens are themselves the terminal nodes of the phrase structure, whereas in our model lexical item tokens are distinct from the terminal nodes of the c-structure.

The definition of complement as the first element merged with a head is not too dissimilar from our own definition, which effectively defines complement as the sister of a head with $\langle L, 0 \rangle$. In defining specifier as any further element merged with a projection of the head, Collins and Stabler license multiple specifiers, but leave little room for a notion of adjunction.

Overall, Collins and Stabler's formalization of BPS captures the most important notions of BPS discussed above and integrated into our

model but, partly due to the less enriched notion of phrase structure which they are modelling, does not appear immediately extensible to cover adjunction and all the other phrase structure phenomena which we have attempted to model in this paper.

CONCLUSION

5

Hitherto, LFG has continued to utilize a model of phrase structure which is largely unchanged from the 1970s, and does not incorporate the insights and advances made within BPS and other theories. Our proposal offers a new model of phrase structure within LFG which captures the central insights of the last forty years of work on phrase structure in a fully formalized, and potentially theoretically broad, way.

REFERENCES

- Avery D. ANDREWS (2018), Sets, heads and spreading in LFG, *Journal of Language Modelling*, 6(1):131–174.
- Avery D. ANDREWS and Christopher D. MANNING (1999), *Complex predicates and information spreading in LFG*, CSLI Publications, Stanford, CA.
- Doug ARNOLD and Louisa SADLER (2013), Displaced dependent constructions, in Miriam BUTT and Tracy Holloway KING, editors, *Proceedings of the LFG13 Conference*, pp. 48–68, CSLI Publications, Stanford, CA.
- Ash ASUDEH (2012), *The logic of pronominal resumption*, Oxford University Press, Oxford.
- Ash ASUDEH, Mary DALRYMPLE, and Ida TOIVONEN (2013), Constructions with lexical integrity, *Journal of Language Modelling*, 1(1):1–54.
- Peter AUSTIN and Joan BRESNAN (1996), Non-configurationality in Australian Aboriginal languages, *Natural Language & Linguistic Theory*, 14(2):215–268.
- Tina BÖGEL, Miriam BUTT, Ronald M. KAPLAN, Tracy Holloway KING, and John T. MAXWELL, III (2009), Prosodic phonology in LFG: A new proposal, in Miriam BUTT and Tracy Holloway KING, editors, *Proceedings of the LFG09 Conference*, pp. 146–166, CSLI Publications, Stanford, CA.

Joan BRESNAN (1982), Control and complementation, *Linguistic Inquiry*, 13(3):343–434, reprinted in Joan Bresnan (ed.), *The mental representation of grammatical relations*, MIT Press, pp. 282–390.

Joan BRESNAN (1995), Linear order, syntactic rank, and empty categories: On weak crossover, in Mary DALRYMPLE, Ronald M. KAPLAN, John T. MAXWELL, III, and Annie ZAENEN, editors, *Formal issues in Lexical-Functional Grammar*, pp. 241–274, CSLI Publications, Stanford, CA.

Joan BRESNAN (1998), Morphology competes with syntax: Explaining typological variation in weak crossover effects, in Pilar BARBOSA, D. FOX, P. HAGSTROM, M. MCGINNIS, and D. PESETSKY, editors, *Is the best good enough? Proceedings from the Workshop on Optimality in Syntax*, MIT Press, Cambridge, MA.

Joan BRESNAN (2001), *Lexical-functional syntax*, Blackwell, Oxford.

Joan BRESNAN, Ash ASUDEH, Ida TOIVONEN, and Stephen WECHSLER (2016), *Lexical-functional syntax*, Wiley-Blackwell, Oxford, second edition. First edition by Joan Bresnan, 2001, Blackwell.

Andrew CARNIE (2000), On the definition of X^0 and XP, *Syntax*, 3(2):59–106.

Andrew CARNIE (2010), *Constituent structure*, Oxford University Press, Oxford, second edition.

Noam CHOMSKY (1957), *Syntactic structures*, Mouton, The Hague.

Noam CHOMSKY (1970), Remarks on nominalization, in Roderick A. JACOBS and Peter S. ROSENBAUM, editors, *Readings in English transformational grammar*, pp. 184–221, Ginn, Waltham, MA, also in: Chomsky, Noam (1972), *Studies on semantics in generative grammar*, The Hague: Mouton, pp. 11–61.

Noam CHOMSKY (1995), Bare phrase structure, in Héctor CAMPOS and Paula KEMPCHINSKY, editors, *Evolution and revolution in linguistic theory*, pp. 51–109, Georgetown University Press, Washington, D.C.

Noam CHOMSKY (2001), *Beyond explanatory adequacy* (MIT Occasional Papers in Linguistics 20), Department of Linguistics and Philosophy, MIT, Cambridge, MA.

Chris COLLINS (2002), Eliminating labels, in Samuel EPSTEIN and David SEELY, editors, *Derivation and explanation in the Minimalist Program*, pp. 42–64, Blackwell, Malden, MA.

Chris COLLINS and Edward STABLER (2016), A formalization of Minimalist syntax, *Syntax*, 19(1):43–78.

Dick CROUCH, Mary DALRYMPLE, Ronald M. KAPLAN, Tracy Holloway KING, John T. MAXWELL, III, and Paula NEWMAN (2011), *XLE Documentation*, Palo Alto Research Center, Palo Alto, CA,
http://www2.parc.com/is1/groups/nl1tt/xle/doc/xle_toc.html.

- Mary DALRYMPLE (2001), *Lexical functional grammar*, Academic Press, San Diego, CA.
- Mary DALRYMPLE (2017), Unlike phrase structure category coordination, in Victoria ROSÉN and Koenraad DE SMEDT, editors, *The very model of a modern linguist: in honor of Helge Dyvik* (Bergen Language and Linguistics Studies (BeLLS), volume 8), pp. 33–55, University of Bergen, Bergen.
- Mary DALRYMPLE and Ronald M. KAPLAN (2000), Feature indeterminacy and feature resolution, *Language*, 76(4):759–798.
- Mary DALRYMPLE, Ronald M. KAPLAN, and Tracy Holloway KING (2001), Weak crossover and the absence of traces, in Miriam BUTT and Tracy Holloway KING, editors, *Proceedings of the LFG01 Conference*, CSLI Publications, Stanford, CA.
- Mary DALRYMPLE, Ronald M. KAPLAN, and Tracy Holloway KING (2004), Linguistic generalizations over descriptions, in Miriam BUTT and Tracy Holloway KING, editors, *Proceedings of the LFG04 Conference*, pp. 199–208, CSLI Publications, Stanford, CA.
- Mary DALRYMPLE, Ronald M. KAPLAN, and Tracy Holloway KING (2007), The absence of traces: evidence from weak crossover, in Annie ZAENEN, Jane SIMPSON, Tracy Holloway KING, Jane GRIMSHAW, Joan MALING, and Christopher MANNING, editors, *Architectures, rules, and preferences: Variations on themes by Joan W. Bresnan*, CSLI Publications, Stanford, CA.
- Mary DALRYMPLE, Ronald M. KAPLAN, and Tracy Holloway KING (2015), Economy of expression as a principle of syntax, *Journal of Language Modelling*, 3(2):377–412.
- Mary DALRYMPLE and Tracy Holloway KING (2013), Nested and crossed dependencies and the existence of traces, in Tracy Holloway KING and Valeria DE PAIVA, editors, *From quirky case to representing space: papers in honor of Annie Zaenen*, pp. 139–151, CSLI Publications, Stanford, CA.
- Mary DALRYMPLE, John J. LOWE, and Louise MYCOCK (2019), *The Oxford reference guide to Lexical Functional Grammar*, Oxford University Press, Oxford.
- Mary DALRYMPLE and Louise MYCOCK (2011), The Prosody-semantics interface, in Miriam BUTT and Tracy Holloway KING, editors, *Proceedings of the LFG11 Conference*, pp. 173–193, CSLI Publications, Stanford, CA.
- Mary DALRYMPLE and Irina NIKOLAEVA (2011), *Objects and information structure*, Cambridge University Press, Cambridge.
- Lachlan DUNCAN (2007), Analytic noun incorporation in Chuj and K'ichee' Mayan, in Miriam BUTT and Tracy Holloway KING, editors, *Proceedings of the LFG07 Conference*, pp. 163–183, CSLI Publications, Stanford, CA.
- Yehuda N. FALK (2001), *Lexical-functional grammar: An introduction to parallel constraint-based syntax*, CSLI Publications, Stanford, CA.

- Naoki FUKUI (1986), *A theory of category projection and its applications*, Ph.D. thesis, MIT.
- Hans-Martin GÄRTNER (2002), *Generalized transformations and beyond: Reflections on minimalist syntax*, de Gruyter, Berlin.
- Gianluca GIORGOLO and Ash ASUDEH (2011), Multimodal communication in LFG: Gestures and the correspondence architecture, in Miriam BUTT and Tracy Holloway KING, editors, *Proceedings of the LFG11 Conference*, pp. 257–277, CSLI Publications, Stanford, CA.
- Jane GRIMSHAW (1991), *Extended projections*, MS, Brandeis University.
- Norbert HORNSTEIN and Jairo NUNES (2008), Adjunction, labeling and bare phrase structure, *Biolinguistics*, 2(1):57–86.
- Ray JACKENDOFF (1977), *\bar{X} syntax: A study of phrase structure*, MIT Press, Cambridge, MA.
- K. A. JAYASEELAN (2008), Bare phrase structure and specifier-less syntax, *Biolinguistics*, 2(1):87–106.
- Ronald M. KAPLAN (1989), The formal architecture of lexical-functional grammar, in Chu-Ren HUANG and Keh-Jiann CHEN, editors, *ROCLING II: Proceedings of the computational linguistics conference*, pp. 3–18, The Association for Computational Linguistics and Chinese Language Processing (ACLCLP), Taipei, also published in *Journal of Information Science and Engineering* 5 (1989), pp. 305–322, and in *Formal issues in Lexical-Functional Grammar*, ed. Mary Dalrymple, Ronald M. Kaplan, John T. Maxwell III and Annie Zaenen, CSLI Publications, 1995, pp. 7–27.
- Ronald M. KAPLAN and Joan BRESNAN (1982), Lexical-functional grammar: A formal system for grammatical representation, in Joan BRESNAN, editor, *The mental representation of grammatical relations*, pp. 173–281, MIT Press, Cambridge, MA.
- Ronald M. KAPLAN and Annie ZAENEN (1989), Long-distance dependencies, constituent structure, and functional uncertainty, in Mark R. BALTIN and Anthony S. KROCH, editors, *Alternative conceptions of phrase structure*, pp. 17–42, University of Chicago Press, Chicago.
- Richard S. KAYNE (1994), *The antisymmetry of syntax*, MIT Press, Cambridge, MA.
- Tracy Holloway KING (1995), *Configuring topic and focus in Russian*, CSLI Publications, Stanford, CA.
- Tracy Holloway KING and Mary DALRYMPLE (2004), Determiner agreement and noun conjunction, *Journal of Linguistics*, 40:69–104.
- András KORNAI and Geoffrey K. PULLUM (1990), The X-bar theory of phrase structure, *Language*, 66:24–50.

- Paul KROEGER (1993), *Phrase structure and grammatical relations in Tagalog*, CSLI Publications, Stanford, CA.
- Joseph LOVETRAN (2018), *Serial verb constructions in Barayin: Typology, description and Lexical-Functional Grammar*, Ph.D. thesis, University of Oxford.
- Joseph LOVETRAN and John J. LOWE (2017), Minimal c-structure: Rethinking projection in phrase structure, in Miriam BUTT and Tracy Holloway KING, editors, *Proceedings of the LFG17 Conference*, pp. 285–305, CSLI Publications, Stanford, CA.
- Jean-Philippe MARCOTTE (2014), Syntactic categories in the correspondence architecture, in Miriam BUTT and Tracy Holloway KING, editors, *Proceedings of the LFG14 Conference*, pp. 408–428, CSLI Publications, Stanford, CA.
- John T. MAXWELL, III and Ronald M. KAPLAN (1996), Unification-based parsers that automatically take advantage of context freeness, in Miriam BUTT and Tracy Holloway KING, editors, *Proceedings of the LFG96 Conference*, CSLI Publications, Stanford, CA.
- Pieter MUYSKEN (1982), Parametrizing the notion “head”, *Journal of Linguistic Research*, 2(3):57–75.
- Louise MYCOCK and John J. LOWE (2013), The Prosodic marking of discourse functions, in Miriam BUTT and Tracy Holloway KING, editors, *Proceedings of the LFG13 Conference*, pp. 440–460, CSLI Publications, Stanford, CA.
- Perna NADATHUR (2013), Weak crossover and the direct association hypothesis, in Miriam BUTT and Tracy Holloway KING, editors, *Proceedings of the LFG13 Conference*, pp. 461–481, CSLI Publications, Stanford, CA.
- Rachel NORDLINGER (1998), *A grammar of Wambaya, Northern Australia*, Pacific Linguistics, Canberra.
- John PAYNE, Rodney HUDDLESTON, and Geoffrey K. PULLUM (2010), The distribution and category status of adjectives and adverbs, *Word Structure*, 3(1):31–81.
- Carl POLLARD and Ivan A. SAG (1994), *Head-Driven Phrase Structure Grammar*, University of Chicago Press, Chicago.
- William J. POSER (1992), Blocking of phrasal constructions by lexical items, in Ivan A. SAG and Anna SZABOLCSI, editors, *Lexical matters*, pp. 111–130, CSLI Publications, Stanford, CA.
- Michael PUTNAM and Thomas STROIK (2009), Traveling without moving: the conceptual necessity of Survive-minimalism, in Michael PUTNAM, editor, *Towards a derivational syntax: Survive-minimalism*, pp. 3–20, Benjamins, Amsterdam.
- Michael PUTNAM and Thomas STROIK (2010), Syntactic relations in Survive-minimalism, in Michael PUTNAM, editor, *Exploring crash-proof grammars*, pp. 143–166, Benjamins, Amsterdam.

Louisa SADLER (1997), Clitics and the structure-function mapping, in Miriam BUTT and Tracy Holloway KING, editors, *Proceedings of the LFG97 Conference*, CSLI Publications, Stanford, CA.

Louisa SADLER and Doug ARNOLD (1994), Prenominal adjectives and the phrasal/lexical distinction, *Journal of Linguistics*, 30:187–226.

Liselotte SNIJDERS (2012), Issues concerning constraints on discontinuous NPs in Latin, in Miriam BUTT and Tracy Holloway KING, editors, *Proceedings of the LFG12 Conference*, pp. 565–581, CSLI Publications, Stanford, CA.

M. SPEAS (1986), *Phrase structure in natural language*, Ph.D. thesis, MIT.

Andrew SPENCER (2005), Case in Hindi, in Miriam BUTT and Tracy Holloway KING, editors, *Proceedings of the LFG05 Conference*, pp. 429–446, CSLI Publications, Stanford, CA.

Edward STABLER (1997), Derivational minimalism, in C. RETORÉ, editor, *Logical aspects of computational linguistics*, pp. 68–95, Springer, New York.

Timothy Angus STOWELL (1981), *Origins of phrase structure*, Ph.D. thesis, Massachusetts Institute of Technology.

Ida TOIVONEN (2003), *Non-projecting words: A case study of Swedish verbal particles*, Kluwer, Dordrecht.

Lisa deMena TRAVIS (1984), *Parameters and effects of word order variation*, Ph.D. thesis, MIT.

Stephen WECHSLER and Larisa ZLATIC (2003), *The many faces of agreement*, CSLI Publications, Stanford, CA.

John J. Lowe

© 0000-0003-0131-1575
john.lowe@ling-phil.ox.ac.uk

University of Oxford
Centre for Linguistics & Philology
Walton Street, Oxford OX1 2HG, UK

Joseph Lovstrand

© 0000-0001-7055-9465
jl119@soas.ac.uk

SOAS University of London
Thornhaugh Street, Russell Square,
London WC1H 0XG, UK

John J. Lowe and Joseph Lovstrand (2020), *Minimal phrase structure: a new formalized theory of phrase structure*, *Journal of Language Modelling*, 8(1):1–52

doi: <https://dx.doi.org/10.15398/jlm.v8i1.247>

This work is licensed under the *Creative Commons Attribution 4.0 Public License*.

cc  <http://creativecommons.org/licenses/by/4.0/>

Distinguishing between paradigmatic semantic relations across word classes: human ratings and distributional similarity

Sabine Schulte im Walde
Universität Stuttgart

ABSTRACT

This article explores the distinction between paradigmatic semantic relations, both from a cognitive and a computational linguistic perspective. Focusing on an existing dataset of German synonyms, antonyms and hypernyms across the word classes of nouns, verbs and adjectives, we assess human ratings and a supervised classification model using window-based and pattern-based distributional vector spaces. Both perspectives suggest differences in relation distinction across word classes, but easy vs. difficult class–relation combinations differ, exhibiting stronger ties between ease and naturalness of class-dependent relations for humans than for computational models.

In addition, we demonstrate that distributional information is indeed a difficult starting point for distinguishing between paradigmatic relations but that even a simple classification model is able to manage this task. The fact that the most salient vector spaces and their success vary across word classes and paradigmatic relations suggests that combining feature types for relation distinction is better than applying them in isolation.

Keywords:
semantic relations, human ratings, distributional semantics, automatic classification

Paradigmatic semantic relations such as synonymy, antonymy, hypernymy and (co-)hyponymy define relations between words that can be found in the same position in a syntagma (de Saussure 1916). They are central to the organisation of the mental lexicon (Deese 1965; Miller and Fellbaum 1991; Murphy 2003), by providing a structure for the lexical concepts that words express. According to Miller and Fellbaum (1991), this relational structure differs across word classes, as *“no single set of semantic relations [...] is adequate for the entire lexicon: nouns, adjectives, and verbs each have their own semantic relations and their own organisation determined by the role they must play in the construction of linguistic messages”*. For example, while hypernymy is considered a natural relation for organising the noun lexicon, it is regarded as less important for organising the verb lexicon, and as rather unnatural for organising the adjective lexicon. In contrast, antonymy is taken to represent the core relation for organising the adjective lexicon, and next to hypernymy, synonymy and entailment, antonymy also plays an important role in the mental lexicon for verbs.

From a computational point of view, modelling paradigmatic semantic relations is important for any application in natural language processing (NLP) such as machine translation and textual entailment, which go beyond the general notion of semantic relatedness and require distinguishing between specific semantic relations. Distributional semantic spaces (also known as vector space models) present a method of determining the meaning and the semantic relatedness between target words within a geometric setting (Budanitsky and Hirst 2006; Turney and Pantel 2010). These models rely on the Distributional Hypothesis and exploit corpus co-occurrences in vector space models to describe and compare the meanings of linguistic units such as words, phrases and sentences (Harris 1954; Firth 1957). Paradigmatic relations are notoriously difficult to be distinguished by standard distributional models, however, because the first-order co-occurrence distributions of the related words tend to be very similar across the relations. For example, in the sentence variants *“The boy/girl/person loves/hates the cat”*, the nominal

co-hyponyms *boy* and *girl* and their hypernym *person*, as well as the verbal antonyms *love* and *hate*, occur in identical contexts, respectively.

Our research presented in this article brings together perspectives from cognitive semantics and distributional semantics, and explores and compares the distinction of three major paradigmatic semantic relations across the three word classes of nouns, verbs and adjectives. We deliberately chose synonymy, antonymy and hypernymy as our target relations because (a) as illustrated above, they play a major role in the organisation of the mental lexicon but nevertheless differ in how natural and important they are for the organisation of the lexica across word classes, and because (b) they are notoriously difficult to be distinguished by distributional models. The questions we address in the current study are the following:

- Can humans and distributional approaches reliably distinguish between synonyms, antonyms and hypernyms across word classes?
- Which class–relation combinations are easy/difficult for humans and which are easy/difficult for distributional approaches?
- Does the ease in relation distinction reflect the naturalness of a relation type for a word class?

We expected that differences in the naturalness of relations across word classes should be reflected by (a) how humans perceive and distinguish semantic relatedness, and by (b) how successful standard distributional approaches are in modelling semantic relatedness.

For the cognitive perspective, we rely on an existing dataset of paradigmatic semantic relation pairs for German (Scheible and Schulte im Walde 2014). Most crucially, the dataset contains ratings of relation strength provided by human judges, for positive as well as for negative relation instances; in addition, the selection of relation pairs across word classes in the dataset is balanced for the number of positive and negative instances, semantic class, frequency and polysemy. For the computational perspective, we rely on distributional similarity scores from standard vector space models as obtained from a large

web corpus, and a simple supervised classification model.¹ Our study demonstrates that the reliable distinction between relations indeed depends on word classes, both for humans and for distributional approaches. Easy vs. difficult class–relation combinations however differ for humans vs. computational models, with stronger ties between ease and naturalness of class-dependent relations for humans.

More specifically regarding our distributional approaches, we demonstrate not only that (a) the models behave differently across word classes, but also that (b) distributional similarity by itself is indeed a difficult starting point for distinguishing paradigmatic relations; nevertheless, (c) even a simple classification model is able to distinguish between relations. Last but not least, we demonstrate that the distributional feature types in the computational models have different strengths and weaknesses in distinguishing between specific paradigmatic relations for specific word classes, which is why exploring feature variants is still a worthwhile subtask in this line of research.

In the remainder of this article we first provide an in-depth overview of previous work on paradigmatic semantic relations in the (mental) lexicon as well as variants of human rating collections and computational approaches regarding paradigmatic relation distinction (Section 2). In Section 3 we describe the human ratings and the distributional information underlying our analyses and classification experiments in the main body of this article (Section 4).

2 RELATED WORK

2.1 *Paradigmatic semantic relations in the lexicon*

The term ‘paradigmatic’ goes back to de Saussure (1916), who introduced a distinction between linguistic elements based on their position relative to each other. This distinction derives from the linear

¹Note that in this study we do not aim to offer in-depth comparisons of multiple distributional representations and algorithms but rather focus on simple standard approaches, given that our goal is not an optimisation of representations and algorithms but exploring the ground distributional information.

nature of linguistic elements, which is reflected in the fact that speech sounds follow each other in time. Saussure refers to successive linguistic elements that combine with each other as ‘syntagma’, and thus the relation between these elements is called ‘syntagmatic’. On the other hand, elements that can be found in the same position in a syntagma, and which could be substituted for each other, are in a ‘paradigmatic’ relationship. While syntagmatic and paradigmatic relations can occur between a variety of linguistic units (such as phonemes, morphemes, words, clauses, sentences), the focus of this research is on paradigmatic relations between words.

A long-standing methodology to explore semantic relations in the mental lexicon makes use of free association norms: Researchers have analysed the (semantic) relationships between target stimuli and their associations, where participants were requested to provide the first word(s) that came to mind when presented with the stimuli. Depending on the collected norms, the stimuli were drawn from just one or across several word classes. Given that the provided associations also vary across word classes, association norms provide a means to investigate the relationships between the stimuli and their associations, among which paradigmatic relations represent a dominant role. In this vein, we provide a brief overview of prominent association norms and relevant semantic analyses.

Following an idea originally suggested by Francis Galton in 1880, the first association norms were collected by Kent and Rosanoff (1910), for 100 English noun and adjective stimuli. The Kent and Rosanoff stimuli were translated into German, allowing for the collection of parallel association norms in German (Russell and Meseck 1959; Russell 1970). Another well-studied collection was assembled by Palermo and Jenkins (1964), comprising associations for 200 words across various parts-of-speech. The Edinburgh Association Thesaurus (Kiss *et al.* 1973) was a first attempt to collect association norms on a larger scale, and to create a network of stimuli and associates, starting from a small set of stimuli derived from the Palermo and Jenkins norms. On a much larger scale, the association norms from the University of South Florida (Nelson *et al.* 1998) were compiled over the course of more than 20 years. More than 6,000 participants produced nearly three-quarters of a million responses to 5,019 stimulus words. The currently largest-scale norms are being collected by de Deyne and colleagues,

who run an online² collection of associations across 13 languages, containing already >10 million stimulus-associate pairs (de Deyne *et al.* 2013).

A major line of research has relied on association norms to investigate the relations between the stimuli and their associations. Regarding paradigmatic and syntagmatic relations, Clark (1971) categorised stimulus-association relations into sub-categories by establishing rules, such as the paradigmatic *minimal-contrast rule* asserting that humans produce associations which are antonymous to the stimuli across word classes, and the syntagmatic *selectional feature realisation rule* asserting that humans produce selectionally preferred complements, also across word classes. Heringer (1986) focused on syntagmatic associations to a small selection of 20 German verbs. He asked his subjects to provide question words as associations (e.g., *wer* ‘who’, *warum* ‘why’), and used the responses to investigate the valency behaviour of the verbs. Bagger Nissen and Henriksen (2006) systematically distinguished between syntagmatic and paradigmatic relations across word classes when comparing associations to nouns, verbs and adjectives for English L1 and L2 adult speakers. They observed different response patterns across the word classes: Regarding paradigmatic relations, for both L1 and L2 they found more paradigmatic responses for nouns than for adjectives, and more for adjectives than for verbs.

To the best of our knowledge, only a small number of investigations distinguished *between* paradigmatic relations in association norms. Schulte im Walde *et al.* (2008) collected and analysed free associations to 409 German nouns and 330 German verbs. They performed detailed analyses at the syntax-semantics interface, and quantified the part-of-speech categories of the associate responses, the syntagmatic co-occurrences, and the syntagmatic and paradigmatic relationships between the stimuli and the associations. Regarding paradigmatic relations, they relied on *GermaNet* (Hamp and Feldweg 1997; Kunze 2000), the German equivalent of *WordNet* (Fellbaum 1998b), where they found paradigmatic relationships for 47% of the verb-verb stimuli-associate tokens and for 17% of the noun-noun stimuli-associate tokens. Most of the verb-verb pairs were in some hypernymy relation (43% co-hyponymy, 26% hyponymy, 21% hypernymy);

²<https://smallworldofwords.org/en/project/stats>

ditto for the noun-noun pairs (47%, 6%, 29%, respectively). Guida and Lenci (2007) replicated most of their analyses on verb association norms for 312 Italian verbs. They found a much larger proportion of verb-verb synonymy (38.3%) and antonymy (4.5%) and a smaller number of hypernymy relations (11.7% co-hyponymy, 5.9% hyponymy, 22.8% hypernymy).

Apart from research on paradigmatic relations that relied on word association norms, there is an enormous body of work that provides theoretical conceptualisations of these relations in the mental lexicon. A seminal description of lexical relations (with a strong focus on antonymy) can be found in Cruse (1986). He states that paradigmatic relations *“reflect the way infinitely and continuously varied experienced reality is apprehended and controlled through being categorised, subcategorised and graded along specific dimensions of variation”*. Cruse describes and exemplifies types and sub-types of paradigmatic relations across word classes. Murphy (2003) focuses on the representation of paradigmatic relations in the lexicon, discussing synonymy, antonymy, contrast, hyponymy and meronymy, also across word classes. In her view, antonymy is a sub-type of contrast within a binary paradigm, and as in Cruse (1986) her analyses on antonymy are *“over-represented, since it is the most controversial semantic relation in terms of whether it is an arbitrary relation among words or a predictable relation among word meanings or concepts”*. Most of her discussions concern linguistic vs. meta-linguistic representations of relations, reference of relations to words vs. concepts, and lexicon storage.

In addition, a series of linguistic and psycholinguistic studies in the 1980s and 1990s investigated paradigmatic relations, typically restricted to either nouns or adjectives, and to a selection of relations. For example, Lehrer and Lehrer (1982), Charles and Miller (1989), Gross *et al.* (1989), Justeson and Katz (1991, 1992) and Murphy and Andrew (1993) studied antonymy and synonymy of adjectives. Chaffin and Herrmann (1981, 1984) looked at various relations mainly for nouns and adjectives, and a selection of syntagmatic verb-noun relations. Winston *et al.* (1987) developed a taxonomy for nominal meronymy, and Chaffin and Glass (1990) explored reading time differences for nominal hypernyms vs. synonyms.

Closest to our work and, as far as we know, the only studies that systematically explored and compared types of paradigmatic relations

across word classes, are those related to the organisation of the Princeton *WordNet*. While the most detailed descriptions are available from a special issue in the *Journal of Lexicography* (Miller *et al.* 1990; Gross and Miller 1990; Fellbaum 1990), Miller and Fellbaum (1991) provide a meta-level summary of relational structures and decisions across word classes. As basis for the *WordNet* organisation, Miller and Fellbaum state that “*the mental lexicon is organised by semantic relations. Since a semantic relation is a relation between meanings, and since meanings can be represented by synsets, it is natural to think of semantic relations as pointers between synsets*”. The semantic relations in *WordNet* include the paradigmatic relations synonymy, hypernymy/hyponymy, antonymy, and meronymy. Because “*no single set of semantic relations [...] is adequate for the entire lexicon: nouns, adjectives, and verbs each have their own semantic relations and their own organisation determined by the role they must play in the construction of linguistic messages*”, these paradigmatic relations are instantiated across word classes to various degrees. For nouns, *WordNet* implements a hierarchical organisation of synsets (i.e., sets of synonymous word meanings) relying on hypernymy relations, and it also provides meronymy relations. For adjectives, Miller and Fellbaum regard antonymy as the central organisational relation. Verbs are considered the most complex and polysemous word class. They are organised on a verb-specific variant of hypernymy, i.e., *troponymy*: v_1 is to v_2 in some manner, that operates on semantic fields which are instantiated as synsets. Troponymy itself is conditioned on entailment and temporal inclusion. In addition to synonymy and troponymy, antonymy is also considered an important relation for verbs. Overall, the *WordNet* specifications for paradigmatic relation between word classes – which themselves rely on a large body of earlier explorations – are taken as the theoretical basis for our work.

2.2

Human ratings of paradigmatic relations

Over the years a number of datasets have been made available for studying and assessing semantic relatedness. Regarding the most famous judgements on *similarity*, Rubenstein and Goodenough (1965) obtained data from 51 subjects on 65 English noun pairs, a seminal study which was later replicated by Miller and Charles (1991)

and Resnik (1995). Finkelstein *et al.* (2002) created *WordSim353*, a set of 353 English noun-noun pairs rated by 16 subjects according to their semantic relatedness on a scale from 0 to 10. For German, Gurevych (2005) replicated Rubenstein and Goodenough's experiments after translating the original 65 word pairs into German. Schmidt *et al.* (2011) translated a subset of 280 target pairs from *WordSim353* into German, however keeping the ratings from the English source.

TOEFL (Test of English as a Foreign Language) is a common dataset for distinguishing *synonymy* from other relations. Each similarity question represents a multiple choice, with four alternatives for a given stem. Landauer and Dumais (1997) collected 80 TOEFL questions for English; Mohammad *et al.* (2007) collected 426 questions for German.

BLESS (Baroni and Lenci 2011) represents one of the earliest collections containing several semantic relations. It focuses on nouns and includes 200 distinct English concrete nouns as target concepts, equally divided between living and non-living entities, and grouped into 17 broad classes. For each target concept, *BLESS* provides related concepts connected through a semantic relation (hypernymy, co-hyponymy, meronymy, attribute, event), or through a null-relation. A similar dataset, *EVALution*, was induced from *ConceptNet* and *WordNet* and subsequently filtered (Santus *et al.* 2015). The *SimLex-999* dataset (Hill *et al.* 2015) was one of the first collections containing information across word classes. It contains 999 word pairs (666 noun, 222 verb and 111 adjective pairs) and was explicitly built to test models on capturing similarity rather than relatedness or association.

While these collections represent state-of-the-art datasets of human ratings of semantic similarity or relatedness, we are interested in judgements on specific types of relatedness and across word classes, which is covered by none of the collections. *WordNet* represents the resource that is most strongly relevant for our purposes but heavily biased towards hypernymy, while synonymy – and even more so – antonymy are represented to a much smaller degree. In addition, the strength of related pairs in *WordNet* is not quantified. Therefore, we rely on a dataset where humans first generated and then rated noun, verb and adjective pairs for synonymy, antonymy and hypernymy.

Although not many approaches in NLP have explicitly addressed the distinction between several paradigmatic semantic relations, there is a rich tradition on identifying synonyms, antonyms or hypernyms, and on distinguishing between subsets of two paradigmatic relations.

Prominent work on identifying **synonyms** was conducted by Edmonds, who employed a co-occurrence network and second-order co-occurrence (Edmonds 1997, 1998, 1999; Edmonds and Hirst 2002), and Curran who explored word-based and syntax-based co-occurrence for thesaurus construction (Curran 2002, 2003). Van der Plas and Tiedemann (2006) compared a standard distributional approach against cross-lingual alignment; Erk and Padó (2008) defined a vector space model for word meaning in context, to identify synonyms and the substitutability of verbs.

Most computational work addressing **hypernyms** was performed for nouns, cf. the lexico-syntactic patterns by Hearst (1992) and an extension of the patterns by dependency paths (Snow *et al.* 2004). Weeds *et al.* (2004), Lenci and Benotto (2012), Santus *et al.* (2014a), Levy *et al.* (2015), Shwartz *et al.* (2016) and Nguyen *et al.* (2017) represent systems that identify hypernyms in distributional spaces. Examples of approaches that addressed the automatic construction of a hypernym hierarchy (for nouns) are Caraballo (2001), Velardi *et al.* (2001), Cimini *et al.* (2004) and Snow *et al.* (2006). Hypernymy between verbs was discussed by Fellbaum (1990), Fellbaum and Chaffin (1990) and Fellbaum (1998a).

There are comparably few approaches to the automatic induction of **antonyms**. A number of studies in the early 90s tested the co-occurrence hypothesis, e.g., Charles and Miller (1989), Justeson and Katz (1991), Fellbaum (1995), and another set of approaches in the last decade elaborated on the distributional properties of antonyms regarding syntagmatic co-occurrence, their discourse functions, and their canonicity (Paradis *et al.* 2009; Jones *et al.* 2012; Paradis 2016). In natural language processing, approaches to antonymy were to a large extent driven by text understanding efforts, or embedded in a larger framework aiming to identify contradiction (Lucerto *et al.* 2004; Harabagiu *et al.* 2006; Mohammad *et al.* 2008; de Marneffe *et al.* 2008).

A main emphasis regarding the distinction *between* paradigmatic semantic relations has been on systems addressing *synonyms vs. antonyms*. Lin *et al.* (2003) used patterns and bilingual dictionaries to retrieve distributionally similar words, and relied on clear antonym patterns such as ‘either X or Y’ in a post-processing step to distinguish synonyms from antonyms. Yih *et al.* (2012) developed a Latent Semantic Analysis (LSA) approach incorporating a thesaurus. Chang *et al.* (2013) extended this approach to induce vector representations that can capture multiple relations. The study by Mohammad *et al.* (2013) evaluated a thesaurus-based approach, where word pairs that occurred in the same thesaurus category were assumed to be close in meaning and marked as synonyms, while word pairs occurring in contrasting thesaurus categories or paragraphs were marked as opposites. Whereas the above-mentioned approaches rely on additional knowledge sources, Turney (2008) developed a corpus-based approach to model relational similarity, addressing (among other tasks) the distinction between synonyms and antonyms. In a similar vein, Scheible *et al.* (2013) showed that with the use of appropriate features, the distributional difference between adjectival antonyms and synonyms can be identified via a simple word space model, and Santus *et al.* (2014c,b) used average precision to distinguish between antonyms and synonyms in standard vector spaces.

Most recently, the problem of synonym/antonym distinction has also been addressed with word embedding models. Adel and Schütze (2014) integrated coreference chains extracted from large corpora into a skip-gram model to create word embeddings that identified antonyms. Ono *et al.* (2015) proposed using thesaurus-based word embeddings to detect antonyms. They suggested the implementation of a model that trains word embeddings on thesaurus information, and one model that incorporated distributional information into the thesaurus model. Pham *et al.* (2015) introduced a multitask lexical contrast model by incorporating WordNet into a skip-gram model to train semantic vectors to predict contexts. Nguyen *et al.* (2016a) proposed two approaches that make use of lexical contrast information in distributional standard vs. word embeddings vector spaces. One approach strengthened word features that were most salient for determining word relatedness, assuming that feature overlap in synonyms is stronger than feature overlap in

antonyms; the other model was an extension of a skip-gram model with negative sampling (Mikolov *et al.* 2013) that integrated the lexical contrast information into the objective function. Nguyen *et al.* (2016b) presented a neural network model that exploited lexico-syntactic patterns from syntactic parse trees and in addition integrated the distance between the related words along the syntactic path as a feature.

Regarding pattern-based approaches to identify and distinguish lexical semantic relations in more general terms, Hearst (1992) was the first to propose lexico-syntactic patterns as empirical pointers towards relation instances, focusing on hyponymy. Girju (2003) applied a single pattern to distinguish pairs of nouns that are in a causal relationship from those that are not, and Girju *et al.* (2006) extended the work towards part-whole relations, applying a supervised, knowledge-intense approach. Chklovski and Pantel (2004) were the first to apply pattern-based relation extraction to verbs, distinguishing five non-disjoint relations (similarity, strength, antonymy, enablement, happens-before). Pantel and Pennacchiotti (2006) developed *Espresso*, a weakly-supervised system that exploits patterns in large-scale web data to distinguish between five noun-noun relations (hypernymy, meronymy, succession, reaction, production). Similarly to Girju *et al.* (2006), they used generic patterns, but relied on a bootstrapping cycle combined with reliability measures, rather than manual resources.

Whereas each of the aforementioned approaches considered maximally two paradigmatic relations and one word class, only a small number of approaches were systematically explored across these relations and classes: Yap and Baldwin (2009) employed syntactic pre-processing and an SVM-based classifier, and experimented with different corpora, to distinguish antonymy, hypernymy and synonymy, while focusing on English nouns. Schulte im Walde and Köper (2013) relied on standard corpus-based patterns to distinguish between the same three paradigmatic relations, proposing a unified framework for German nouns, verbs and adjectives. Roth and Schulte im Walde (2014) extended the pattern-based approach by incorporating discourse markers and applied their model across the same relations and the three word classes, both for English and for German.

DATA

3

The following two subsections describe the two types of data our explorations rely on: the cognitive resource with human ratings of paradigmatic relations (Section 3.1), and the distributional information used in the computational models (Section 3.2).

Human ratings of paradigmatic relations

3.1

Our database of semantic relations for German adjectives, nouns and verbs focuses on the three types of paradigmatic relations referred to as *sense-relations* by Lyons (1968, 1977): synonymy, antonymy, and hypernymy. For the collection of the database, we implemented two experiments involving human participants (Scheible and Schulte im Walde 2014). Starting with a set of target words, in the first experiment participants were asked to propose suitable synonyms, antonyms, and hypernyms for each of the targets. For example, for the target verb *befehlen* ('to command'), participants proposed synonyms such as *anordnen* ('to order'), antonyms such as *gehorschen* ('to obey'), and hypernyms such as *sagen* ('to say'). In the second experiment, participants were asked to rate the strength of a given semantic relation with respect to a word pair on a given scale. For example, participants would be presented with the pair *befehlen*–*gehorschen* and asked to rate the strength of antonymy between the two words. All word pairs were assessed with respect to all three relation types.

In the following, Section 3.1.1 provides an overview of GermaNet, from which the set of target words was drawn. Section 3.1.2 introduces the platform used to implement the experiments, Amazon Mechanical Turk. Sections 3.1.3 and 3.1.4 then describe the two experiments to collect the human rating data. The dataset is publicly available at <http://www.ims.uni-stuttgart.de/data/sem-rel-database>.

Target source: GermaNet

3.1.1

GermaNet is a lexical-semantic word net that provides information on semantic relations for German nouns, verbs, and adjectives. GermaNet has been modelled along the lines of the Princeton WordNet for English (Miller *et al.* 1990; Fellbaum 1998b) and shares its general design principles (Hamp and Feldweg 1997; Kunze and Wagner

1999; Lemnitzer and Kunze 2007). For example, lexical units denoting the same concept are grouped into synonym sets ('synsets'). These are in turn interlinked via conceptual-semantic relations (such as hypernymy) and lexical relations (such as antonymy). For each of the major word classes, the databases further take a number of semantic categories into consideration, expressed with top-level nodes in the semantic network (such as *Artefakt* 'artifact', *Geschehen* 'event', *Gefühl* 'feeling'). In contrast to WordNet, GermaNet also includes so-called 'artificial concepts' to fill lexical gaps and thus enhance network connectivity, and to avoid unsuitable co-hyponymy (e.g. by providing missing hypernyms or hyponyms). GermaNet also differs from WordNet in the way in which it handles parts-of-speech. For example, while WordNet employs a clustering approach for structuring adjectives, GermaNet uses a hierarchical structure similar to the one employed for the noun and verb hierarchies. Finally, WordNet and GermaNet also differ in size: While WordNet 3.0 contains a total of 117,659 synsets and 155,287 lexical units, the respective numbers for GermaNet 6.0 (which we used in the current study) are considerably smaller, with 69,594 synsets and 93,407 lexical units.

Since GermaNet is the largest database of its kind for German, and given that it encodes all types of relations that are of interest for us (synonymy, antonymy, and hypernymy), it represented a suitable starting point for our purposes.³ Relying on GermaNet version 6.0⁴ and the respective *JAVA API*, we used a stratified sampling technique to randomly select 99 nouns, 99 adjectives and 99 verbs from the GermaNet files. The random selection was balanced for:

1. the **size of the semantic classes**,⁵ accounting for the 16 semantic adjective classes and the 23 semantic classes each for nouns and verbs, as represented by the file organisation;

³For reasons why we did not use GermaNet to directly extract relation pairs (i.e., it is unbalanced regarding relation types; does not contain relation quantification or negative evidence; etc.), see the end of Section 2.2.

⁴When we started the collection, GermaNet 6.0 represented the latest version. Information about current statistics can be found at <http://www.sfs.uni-tuebingen.de/GermaNet/>.

⁵For example, if an adjective GermaNet class contained 996 word types, and the total number of adjectives over all semantic classes was 8,582, and with

2. **three polysemy classes** according to the number of GermaNet senses: I) monosemous, II) two senses and III) > two senses;
3. **three frequency classes** according to the type frequency in the German web corpus *SdeWaC* (Faaß and Eckart 2013), which contains approx. 880 million words: I) *low* (200–2,999), II) *mid* (3,000–9,999) and III) *high* ($\geq 10,000$).

The total number of 99 targets per word class resulted from distinguishing 3 polysemy classes and 3 frequency classes, $3 \times 3 = 9$ categories, and selecting 11 instances from each polysemy–frequency category, in proportion to the semantic class sizes.

Experimental platform: Mechanical Turk

3.1.2

The experiments described below were implemented in Amazon Mechanical Turk (AMT),⁶ a web-based crowdsourcing platform which allows simple tasks (so-called HITs) to be performed by a large number of people in return for a payment. In our first experiment, human associations were collected for different semantic relation types, where AMT workers were asked to propose suitable synonyms, antonyms, and hypernyms for each of the targets. The second experiment was based on a subset of the generated synonym/antonym/hypernym pairs and asked the participants to rate each pair for the strength of synonymy, antonymy, and hypernymy between them, on a scale between 0 (minimum strength) and 5 (maximum strength). To control for non-native speakers of German and spammers, each batch of HITs included two examples of ‘non-words’ (i.e., invented words following German morphotactics) in random positions. If participants did not recognise the invented words, we excluded all their ratings from consideration. While we encouraged participants to complete all HITs in a given batch, we also accepted a smaller number of submitted HITs, as long as the workers had a good overall feedback score.

99 stimuli collected in total, we wanted that proportion out of 99 stimuli that corresponded to the proportion of the class size relative to the total number of adjectives $996/8,582$ and thus randomly selected 11 adjectives from this class: $99 * 996/8,582 \approx 11.49$.

⁶<https://www.mturk.com>

3.1.3

Generation experiment

The goal of the generation experiment was to collect human associations for the semantic relation types synonymy, antonymy, and hypernymy. For each of our 3×99 adjective, noun, and verb targets, we asked 10 participants to propose a suitable synonym, antonym, and hypernym. Targets were bundled randomly in 9 batches per word class, each including 9 targets plus two invented words. The experiment consisted of separate runs for each relation type to avoid confusion between them, with participants first generating synonyms, then antonyms, and finally hypernyms for the targets, resulting in $3 \text{ word classes} \times 99 \text{ targets} \times 3 \text{ relations} \times 10 \text{ participants} = 8,910$ target–response pairs. Table 1 provides some examples of the generated target–response pairs for each word class and each paradigmatic relation, accompanied by the number of times a specific response was given (with a maximum of 10 responses).

3.1.4

Rating experiment

In the second experiment, Mechanical Turk workers were asked to rate a given semantic relation with respect to a word pair on a 6-point scale between 0 (minimum strength) and 5 (maximum strength). The main purpose of this experiment was to identify and distinguish between “strong” and “weak” examples for specific relations across word classes. The number of times a specific response was given in the generation experiment does not necessarily indicate the strength of the relation. This is especially true for responses that were suggested by only one or two participants, where it is difficult to tell if the response is an error, or if it relates to an idiosyncratic sense of the target word that the other participants did not think of in the first instance. Crucially, in the rating experiment all word pairs were assessed with respect to all three relation types, thus asking not only for positive but also for negative evidence of semantic relation instances.

The set of word pairs used as an input was a carefully selected subset of responses acquired in the generation experiment. For each of the 99 targets and each of the semantic relations (antonymy, synonymy, and hypernymy), we included two responses: the *response with the highest frequency* (random choice if several available) and a *response*

with a low frequency (2, if available, otherwise 1; random choice if several available). Multi-word responses and blanks were excluded.

In theory, each target should have 6 associated pairs ($2 \times \text{ANT}$, $2 \times \text{HYP}$, $2 \times \text{SYN}$). In practice, there are sometimes fewer than 6 pairs per target in the dataset, because (a) for some targets, only one response was available for a given relation (e.g., if all 10 participants provided the same response), or (b) no valid response of the required frequency type was available. The resulting dataset includes 1,684 target–response pairs altogether, 546 of which are adjective pairs, 574 noun pairs, and 564 verb pairs. To avoid confusion, the ratings were collected in separate experimental settings, i.e., for each word class and each relation type, all generated pairs were first evaluated for the strength of one relation, and then for the strength of another relation. Table 2 provides some examples of mean ratings for target–response pairs and the three semantic relations, together with the original relation (see column *Generation*) and the strength of generation (1–10).

3.2

Corpora and distributional information

As a corpus for our distributional models we relied on one of the currently largest German web corpora, *DECOW14AX*, with approx. 12 billion words (Schäfer and Bildhauer 2012). It was already lemmatised and assigned part-of-speech tags by the *Tree Tagger* (Schmid 1994).

We induced two types of distributional information from the web corpus in order to create two types of vector space models (Bullinaria and Levy 2007; Turney and Pantel 2010), one using window co-occurrence and one using lexico-syntactic patterns. Regarding *window co-occurrence*, we created a standard vector space for all target and response words that were part of our relation pairs. We relied on co-occurrence frequencies from a sentence-internal 20-word window (i.e., 20 words to the left and 20 words to the right of a word in the corpus but not going beyond sentence borders, as sentences in *DECOW14AX* are scrambled) to determine the co-occurring content words and the strengths of co-occurrence. For example, if *schnurren* ‘to purr’ occurred a total of 235 times in the context of *Katze* ‘cat’ – where the context of *Katze* is defined as the 20 preceding and the 20 following words – then the dimension *schnurren* for the target word

Katze in the vector space was assigned the frequency 235. To compare different windows sizes and vector space strengths, we also used co-occurrence information from a 5-word window, and we also compared co-occurrence frequencies with *local mutual information (lmi)* scores, cf. Evert (2005), which often provide better estimates for word co-occurrence strength. The window co-occurrence information refers to words (i.e., it provides co-occurrence vectors for target or response words such as *Katze* ‘cat’) rather than to target–response word pairs (such as *Katze–Tier* ‘cat–animal’), so Section 4.2 will explain how to induce vectors for word pairs from the vectors of individual words.

Regarding *lexico-syntactic patterns*, we directly induced a vector space for the word-relation pairs (Hearst 1992; Chklovski and Pantel 2004, i.a.). I.e., we relied on the linear word sequences $l_1 \dots l_n$ in the corpus between any two related words w_i and w_j (representing synonyms, antonyms or hypernyms) to initiate the vector space dimensions for the relation pair w_i-w_j . For example, if we saw the hypernymy pair *Katze–Tier* ‘cat–animal’ in the token sequence “... **Tier** wie *Huhn, Taube, Katze* ...”, the respective lexico-syntactic pattern (and, correspondingly, one dimension in the vector space) was the intermediate sequence “*wie Huhn, Taube,*”. We distinguished between two sub-types of patterns in our vector representations, those taking into account the linear order of the words w_i and w_j (i.e., patterns distinguishing between $w_i l_1 l_2 \dots l_n w_j$ and $w_j l_1 l_2 \dots l_n w_i$), and those without taking the direction into account.

DISTINGUISHING PARADIGMATIC RELATIONS

4

As outlined in the Introduction, our research brings together perspectives from cognitive lexical semantics and distributional semantics, and compares the distinction of paradigmatic semantic relations for German across the three word classes of nouns, verbs and adjectives:

- Can humans and distributional approaches reliably distinguish between synonyms, antonyms and hypernyms across word classes?

- Which class–relation combinations are easy/difficult for humans and which are easy/difficult for distributional approaches?
- Does the ease in relation distinction reflect the naturalness of a relation type for a word class?

We expect that differences in the naturalness of paradigmatic relations across word classes are reflected in how humans perceive and distinguish semantic relatedness (Section 4.1), and in the performance of corpus-based distributional approaches (Section 4.2).

4.1

Human distinction

For the cognitive perspective, we rely on the dataset of human-generated paradigmatic semantic relation pairs rated for their relation strength as described in the rating experiment in Section 3.1.4. We disregarded relation pairs that were originally generated only once, and we also disregarded ambiguous pairs, i.e. pairs that were generated for more than one relation type. For example, the noun *Erde* ('soil') was generated both as synonym (3 times) and as hypernym (twice) for the target noun *Torf* ('peat').

Table 3 shows the numbers of relation pairs across word classes and relation types with respect to the originally generated relation. The table also compares pairs excluding vs. including ambiguity (–/+*amb*, respectively). It is already interesting to observe that for relation pairs involving hypernymy and synonymy (HYP and SYN) there was considerably more ambiguity among the generated relation pairs than for antonymy (ANT): For verbs and adjectives, for which hypernymy represents a less natural semantic relation than for nouns, only 29.3–34.2% of the considered generated pair types

Table 3:
Number of relation pairs
in the dataset

		ANT	HYP	SYN	<i>all</i>
NOUN	– <i>amb</i>	101	91	82	274
	+ <i>amb</i>	118	159	151	428
VERB	– <i>amb</i>	122	66	63	251
	+ <i>amb</i>	132	193	193	518
ADJ	– <i>amb</i>	127	54	58	239
	+ <i>amb</i>	133	184	189	506

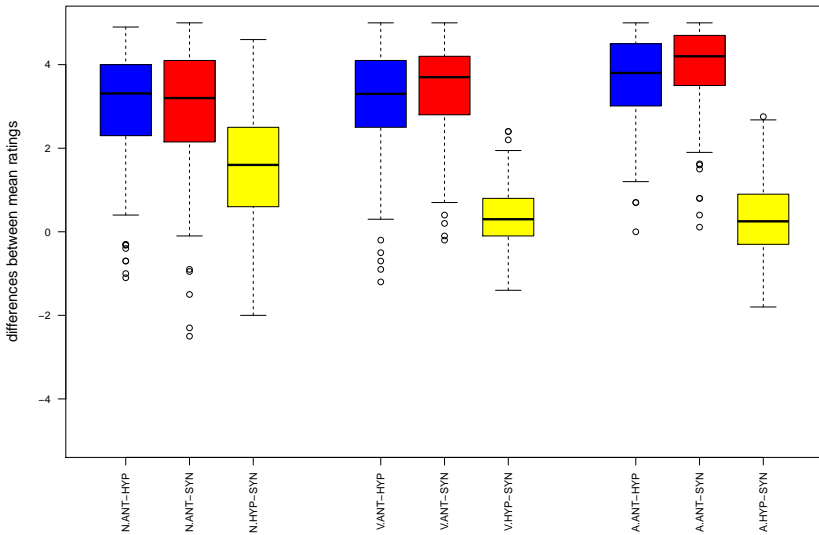
were unambiguous, while for nouns the unambiguous pairs correspond to $\approx 55\%$. As mentioned above, the *-amb* dataset represents the basis for exploring differences in relation distinction across word classes by humans. For completeness, Appendix A.1 provides the human distinction results for ambiguous pairs, in comparison to those for unambiguous pairs.

In order to assess how well the experiment participants could distinguish between the paradigmatic relations, we calculated the differences in mean ratings for a specific relation pair. For example, we obtained a mean rating of 4.4 on our scale 0–5 from the experiment participants for the antonym pair *befehlen–gehörchen* (‘command–obey’) regarding antonymy, and we obtained a mean rating of only 0.3 for this pair regarding hypernymy, so the difference in the mean ratings was 4.1. Obviously, the experiment participants were rather sure that the target pair represented antonymy, and they were also rather sure that the target pair did not represent hypernymy. In contrast, the difference in mean ratings for the antonym pair *bedürfen–verzichten* (‘require–abstain’) regarding antonymy vs. hypernymy ratings was only 2.1, demonstrating that the latter antonym pair represented a weaker instance of antonymy for the experiment participants. Table 2 provides differences in mean ratings for further example target–response pairs (see column ‘Difference’).

Figures 1 and 2 present these mean differences for each word class and across all relation pairings. Figure 1 provides a coarse view on relation distinction and does not tell us which relation was the original relation and which was the rated relation (e.g., whether a pair has been generated as a synonym pair and then rated for synonymy vs. antonymy, or whether a pair has been generated as an antonym pair and then rated for antonymy vs. synonymy); Figure 2 then incorporates this distinction.

The figures illustrate that the experiment participants found it easier to distinguish between antonyms and hypernyms (ANT–HYP, blue boxes) as well as between antonyms and synonyms (ANT–SYN, red boxes), where the differences in mean ratings between the original and the rated relation are larger, in comparison to distinguishing between hypernyms and synonyms (HYP–SYN, yellow boxes), where the differences in mean ratings for the two relations are smaller. These findings hold across word classes, but we can also see that the ten-

Figure 1:
Human
distinction of
paradigmatic
relations
(coarse)



density is stronger for adjectives and verbs (in comparison to nouns) where the differences are ≈ 0 , i.e., the mean ratings for synonyms and hypernyms regarding a specific word pair were nearly identical.

The fine-grained analysis in Figure 2 in addition demonstrates that adjectival HYP–ANT is more difficult for the humans than adjectival ANT–HYP, and that adjectival HYP–SYN is more difficult than adjectival SYN–HYP (in both cases the boxes do not even overlap); to a lesser degree we find the same dispute between verbal HYP–ANT and ANT–HYP (where the median of the former is outside the box of the latter). Interestingly, in all these three cases the differences between mean ratings were lower when the original relation was hypernymy, which represents a less natural semantic relation for verbs and adjectives than for nouns, i.e. the experiment participants did not perceive the generated hypernyms as strong instances of that relation type in comparison to the respective other paradigmatic relation.

Overall, the differences in mean ratings suggest (a) that humans clearly distinguish antonyms from synonyms and also from hypernyms, but have more difficulties in distinguishing between synonyms and hypernyms, and (b) that distinguishing hypernymy from the other two relations is more difficult for adjectives and verbs (in comparison to nouns), for which hypernymy represents a less natural semantic relation. The boxplots in Appendix A.1 – which compare the coarse- and

Distinguishing paradigmatic semantic relations

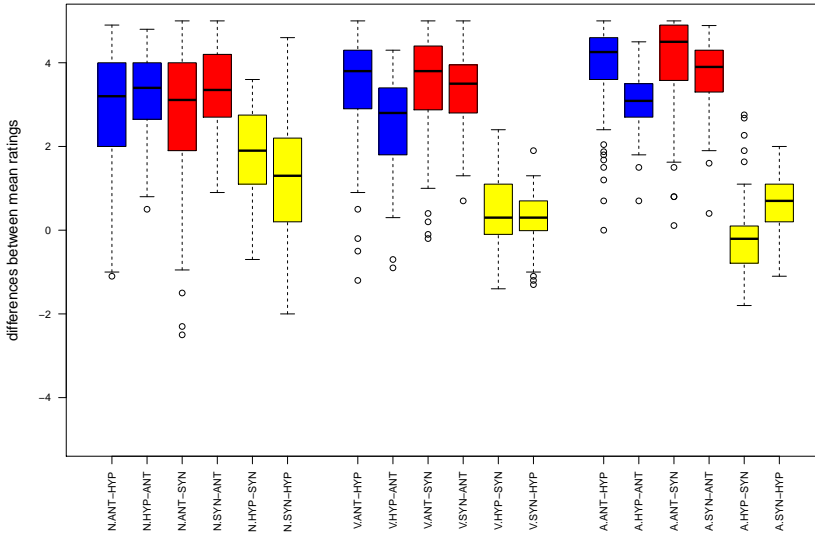


Figure 2:
Human
distinction of
paradigmatic
relations (**fine**)

fine-grained analyses in Figures 1 and 2 against the respective analyses on relation pairs including ambiguity – confirm these insights.

Distributional classification models

4.2

For the computational perspective, we explore two levels of processing the distributional co-occurrence information in the standard vector space models introduced in Section 3.2. We start out with cosine distances between any two word pairs within the set of target–response pairs, in order to illustrate the difficult basis of a distributional model for distinguishing between paradigmatic relations (Section 4.2.1). In a series of supervised classification experiments we then present the results of automatically categorising the target–response pairs into semantic relations (Section 4.2.2).

Cosine similarities between relation pairs

4.2.1

As explained in Section 3.2, we rely on corpus co-occurrences to activate and quantify dimensions in word vectors (Bullinaria and Levy 2007; Turney and Pantel 2010). The geometric distance between two word vectors then determines the distance between the two words. The closer two vectors are in the vector space, the more semantically

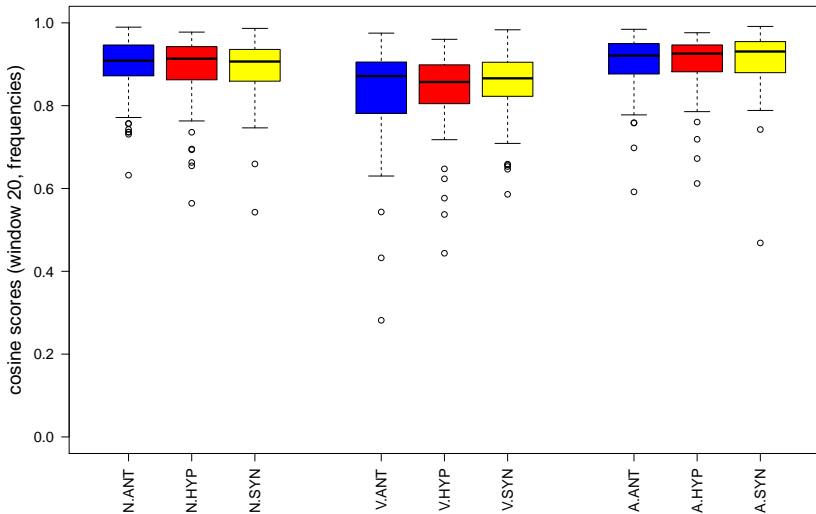
related we expect the represented words to be, based on the Distributional Hypothesis (Harris 1954; Firth 1957).

Regarding paradigmatic semantic relations, the generally agreed upon assumption is that the related word pairs are relatively close to each other in word space across the relation types, because for all paradigmatic relations the related words are distributionally similar to each other. In the following, we explore this assumption for our dataset.

We calculated the cosine scores between the target words and the response words for each target–response pair. The cosine score specifies the angle between two vectors, with 1 indicating minimal distance (i.e., identity, and therefore maximal relatedness) between the vectors. We used the same set of unambiguous rated pairs as exploited by Figures 1 and 2, together with the respective co-occurrence vector spaces. Figures 3 and 4 present boxplots of cosine scores for all word pairs across word classes and semantic relations, relying on co-occurrence frequencies within 20-word windows, and on the corresponding vectors with lmi scores.

The plots illustrate that the cosine values are indeed very similar across our three paradigmatic relations for a specific word class, with slightly lower scores for verb relatedness. The lmi scores obviously influence the magnitudes of the cosine scores, and they manage

Figure 3:
Boxplots of cosine scores across classes and relations (window 20, frequencies)



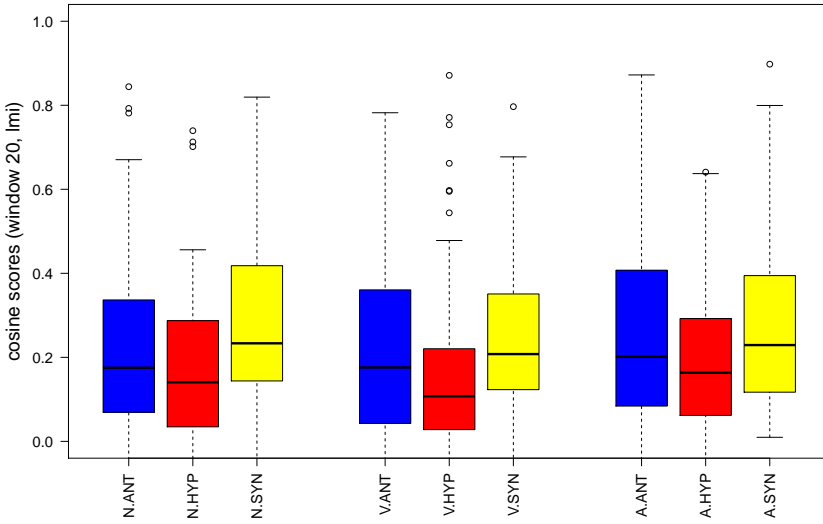


Figure 4: Boxplots of cosine scores across classes and relations (window 20, lmi scores)

to disperse them. Appendix A.2 illustrates that the same tendencies can be observed for 5-word co-occurrence windows, and also when extending the underlying dataset with ambiguous word pairs.

Automatic classification of relation pairs

4.2.2

In a series of classification experiments relying on the distributional word spaces we explored whether automatic approaches are able to categorise word pairs according to their paradigmatic semantic relations, even though the vectors of the word pairs are all very close in vector space. In the following, we present classification results of a simple *nearest-centroid classifier* (also known as *Rocchio classifier*, cf. Manning *et al.* 2008) that compares window-based co-occurrence features against pattern-based co-occurrence features. A subset of the classification experiments was previously described by Schulte im Walde and Köper (2013) and David (2014), but was re-implemented and re-run for the current article to ensure the same underlying target pairs and corpus data across approaches.

The classification was done as follows. For each word class separately, we calculated three mean vectors: one for each lexical semantic relation (synonymy, antonymy, hypernymy), as based on a set of training pairs. We then predicted the semantic relation for a set of test pairs, by choosing for each test pair the most similar class mean vector

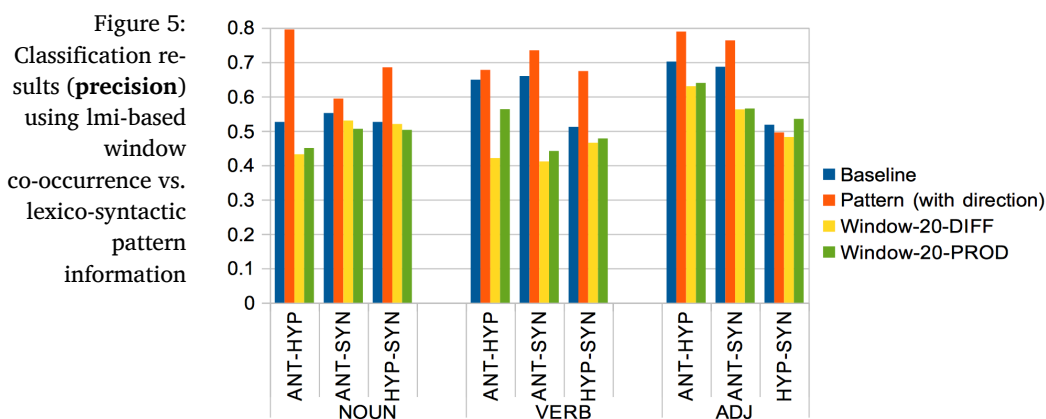
as determined by the respective cosine scores. Across the experiments, we used 5-fold cross-validation for training and testing.

The classification setup for the pattern-based vectors is straightforward, because a pattern vector represents a word pair. The window co-occurrence vectors however represent words and not word pairs and thus require a preprocessing step to obtain vectors for word pairs. We applied two variants to initiate window co-occurrence vectors for the target–response pairs, as based on their individual word vectors:

WINDOW-DIFF: For each target–response word pair, we calculated the difference vector between the two involved word vectors, i.e., the value of each dimension in the difference vector is computed as the absolute difference between the respective values in the two word vectors. The centroids of the relation classes correspond to mean difference vectors.

WINDOW-PROD: For each target–response word pair, we calculated the product vector for the two involved word vectors, i.e., the value of each dimension in the product vector is computed as the product of the respective values in the two word vectors. The centroids of the relation classes correspond to mean product vectors.

Figures 5 and 6 present the results of the nearest-centroid classifier across word classes, relations, and types of distributional information. While Figure 5 shows the results in terms of precision (i.e., the average proportion of correct class assignments among all classified instances of relation pairs), Figure 6 shows the results in terms



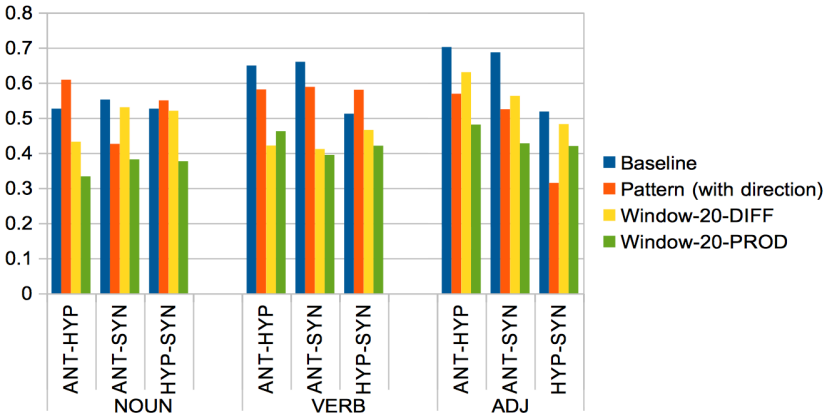


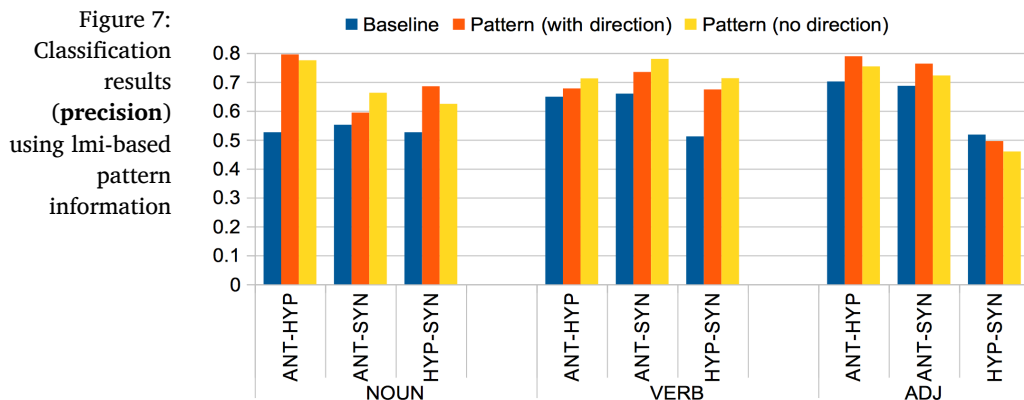
Figure 6: Classification results (**accuracy**) using lmi-based window co-occurrence vs. lexico-syntactic pattern information

of accuracy (i.e., the average proportion of correct class assignments among all existing instances of relation pairs).

Both Figures 5 and 6 compare vector spaces with lmi scores for pattern-based features with direction (i.e., patterns distinguishing between $w_i \langle pattern \rangle w_j$ and $w_j \langle pattern \rangle w_i$), and window-based features relying on a 20-word co-occurrence window. We decided in favour of lmi-score vector spaces rather than frequency vector spaces, because our analyses in Section 4.2.1 indicated that lmi scores disperse the cosine scores in the vectors. Results of other classification variants (i.e., relying on frequencies; pattern-based features without direction information; window-based features relying on a 5-word co-occurrence window) are described in Appendix A.3.

Figure 5 shows that – regarding precision – pattern information in most cases outperforms not only the respective majority baseline but also the two variants of window information. The only exception takes place for distinguishing between adjectival HYP–SYN. And also except for this very case, classification based on window features is consistently worse than the baselines. WINDOW-DIFF and WINDOW-PROD results do not show consistent differences, except for verbal ANT–HYP, for which WINDOW-PROD clearly outperforms WINDOW-DIFF. Figures 14 and 15 in Appendix A.3 illustrate that the same tendencies are found for frequency-based vector spaces, which are however overall worse than lmi-based vector spaces.

Figure 6 shows that – regarding accuracy – most of the classification results are below the majority baseline. Pattern-based results



are only above baseline results for nominal ANT–HYP and HYP–SYN as well as for verbal HYP–SYN distinctions; window-based results do not outperform the baselines in any of the scenarios. In cases where both pattern-based and window-based results are below the baselines, pattern-based results outperform window-based results for all verbal relation distinctions; window-based results outperform pattern-based results for all adjectival relation distinctions, and for nominal ANT–SYN. In most cases, WINDOW-DIFF clearly outperforms WINDOW-PROD.

Figure 7 provides a view that is quite alike Figure 5, zooming into the overall best results⁷ when using pattern-based information. First of all, Figure 7 compares the classification results with/without using pattern direction information. We can see that there are no consistent differences between the two representations: the patterns without directional information are slightly better for verbs; and the patterns with directional information are slightly better for adjectives. The results for nouns depend on the relation types. Appendix A.3 provides additional information illustrating in the same manner that lmi-based patterns in general outperform lmi-based window information, both for a 20-word and a 5-word window.

Moreover, comparing our pattern-based classification results in Figure 7 with the coarse view on human relation distinction in Figure 1,

⁷ For the remainder of the paper, we will explore precision rather than accuracy results because we are interested in the qualitative feature potential, disregarding data sparsity issues.

we do not see much overlap in general tendencies. In relation to the respective baselines, nominal ANT–HYP and nominal and verbal HYP–SYN distinctions are handled particularly well in the automatic classifications; adjectival HYP–SYN distinction is particularly bad. This provides a very different story than the human distinctions, where HYP–SYN were consistently distinguished more poorly than the other relation combinations, across word classes.

Tables 4–9 provide confusion matrices for a more detailed view on correct and wrong relation classifications. Here we took into account all class assignments of relation pairs in the respective 5-fold cross-validation, a total of $N = 1,528$ across word classes and relation combinations. For each word class and relation, we calculated the number of pairs classified correctly/wrongly, or not at all.⁸ The diagonal numbers in bold font indicate the correct class assignments, and the accuracy acc_N indicates the proportion of those correct classifications regarding N .

Comparing the lmi-based Tables 4–6, the acc_N scores confirm that pattern-based information outperforms both variants of window-based information. We can also observe differences across word classes and relation types. For example, the patterns are extremely useful for identifying verbal antonyms, while WINDOW-DIFF is crucial for discover-

⁸ A word pair was not classified at all if all vector feature values of at least one of the words were zero. This happened if one or both of the words did not occur in the corpus, or if the words did not co-occur (in the case of patterns), or after multiplying feature values.

		ANT	HYP	SYN	NONE	<i>all</i>
NOUN	ANT	101	21	24	56	202
	HYP	10	135	9	28	182
	SYN	33	34	54	43	164
VERB	ANT	152	31	18	43	244
	HYP	21	90	15	6	132
	SYN	21	21	51	33	126
ADJ	ANT	139	21	20	74	254
	HYP	7	44	25	32	108
	SYN	11	11	52	42	116
		$acc_N = 0.5353$			$N = 1,528$	

Table 4:
Confusion matrix for class assignment using **lmi-based pattern** features (with direction)

Table 5:
Confusion matrix
for class assignment using
lmi-based WINDOW-DIFF
features

		ANT	HYP	SYN	NONE	<i>all</i>
NOUN	ANT	118	77	7	0	202
	HYP	32	145	5	0	182
	SYN	79	78	7	0	164
VERB	ANT	73	91	80	0	244
	HYP	18	91	23	0	132
	SYN	29	46	51	0	126
ADJ	ANT	189	25	40	0	254
	HYP	42	27	39	0	108
	SYN	41	19	56	0	116
$acc_N = 0.4954$						$N = 1,528$

Table 6:
Confusion matrix
for class assignment using
lmi-based
WINDOW-PROD features

		ANT	HYP	SYN	NONE	<i>all</i>
NOUN	ANT	79	51	20	52	202
	HYP	27	97	16	42	182
	SYN	47	49	23	45	164
VERB	ANT	117	29	65	33	244
	HYP	38	29	38	27	132
	SYN	27	21	68	10	126
ADJ	ANT	135	21	34	64	254
	HYP	28	33	23	24	108
	SYN	28	18	45	25	116
$acc_N = 0.4097$						$N = 1,528$

Table 7:
Confusion matrix
for class assignment using
frequency-based pattern
features (with direction)

		ANT	HYP	SYN	NONE	<i>all</i>
NOUN	ANT	132	19	17	34	202
	HYP	17	136	7	22	182
	SYN	46	47	43	28	164
VERB	ANT	135	47	32	30	244
	HYP	18	94	18	2	132
	SYN	26	27	51	22	126
ADJ	ANT	140	34	33	47	254
	HYP	7	67	31	3	108
	SYN	10	25	62	19	116
$acc_N = 0.5628$						$N = 1,528$

Distinguishing paradigmatic semantic relations

		ANT	HYP	SYN	NONE	<i>all</i>
NOUN	ANT	121	42	39	0	202
	HYP	36	121	25	0	182
	SYN	51	47	66	0	164
VERB	ANT	136	58	50	0	244
	HYP	36	65	31	0	132
	SYN	32	30	64	0	126
ADJ	ANT	151	41	62	0	254
	HYP	39	34	35	0	108
	SYN	33	16	67	0	116

$acc_N = 0.5399$ $N = 1,528$

Table 8:
Confusion matrix
for class assignment using
frequency-based
WINDOW-DIFF features

		ANT	HYP	SYN	NONE	<i>all</i>
NOUN	ANT	109	43	32	18	202
	HYP	40	101	29	12	182
	SYN	42	43	61	18	164
VERB	ANT	44	99	97	4	244
	HYP	10	103	19	0	132
	SYN	14	43	67	2	126
ADJ	ANT	108	62	62	22	254
	HYP	22	57	29	0	108
	SYN	18	32	60	6	116

$acc_N = 0.4647$ $N = 1,528$

Table 9:
Confusion matrix
for class assignment using
frequency-based
WINDOW-PROD features

ing verbal hypernyms. WINDOW-PROD seems to overall classify more poorly than the other two feature types; it slightly outperforms them in only one case, for verbal synonyms. WINDOW-DIFF has a particular strength in that it classifies all N relation pairs (NONE = 0 for all class–relation combinations). Obviously the vectors are less sparse than for the patterns, and they do not become more sparse in the vector pair creation, differently to the WINDOW-PROD vectors.

Looking at the frequency-based Tables 7–9, we find the same tendencies regarding acc_N as for Tables 4–6, but the frequency-based acc_N values are consistently higher than the respective lmi-based acc_N values. This is in contrast to what the precision results presented in Appendix A.3 show, where the frequency-based precision results for the patterns are worse than the respective lmi-based precision results, and the results for the window-based vector spaces vary. Comparing

Tables 7 and 8, we can observe that both patterns and WINDOW-DIFF are strong in identifying antonyms across word classes; that the patterns are also strong in identifying hypernyms (and WINDOW-DIFF is less strong) across word classes; and that WINDOW-DIFF is strong in identifying synonyms (and the patterns are less strong) across word classes. Thus, the confusion matrices demonstrate in more detail than the plots that the most successful vector spaces each have their strengths and weaknesses regarding specific relation types.

5

CONCLUSION

In this article, we explored the distinction between the three paradigmatic semantic relations of synonymy, antonymy, and hypernymy, both from a cognitive linguistic perspective and a computational linguistic perspective. We expected differences in how natural relations are across word classes to be reflected in how humans perceive and distinguish semantic relatedness, and in the extent that corpus-based distributional approaches are successful in modelling semantic relatedness. More specifically, we addressed the following questions in this study:

- Can humans and distributional approaches reliably distinguish between synonyms, antonyms and hypernyms across word classes?
- Which class–relation combinations are easy/difficult for humans and which are easy/difficult for distributional approaches?
- Does the ease in relation distinction reflect the naturalness of a relation type for a word class?

Regarding the human distinction between the three paradigmatic relations, we first of all observed that among the human-generated relation pairs involving hypernymy and synonymy there was considerably more ambiguity than for antonymy. Especially for verbs and adjectives, for which hypernymy represents a less natural semantic relation than for nouns, a large proportion of the considered generated pair types were ambiguous between hypernymy and synonymy.

In addition, when looking at the differences in mean relation ratings we found (a) that humans clearly distinguished antonyms from synonyms and also from hypernyms, but had more difficulties in distinguishing between synonyms and hypernyms, and (b) that distinguishing hypernymy from the other two relations was more difficult for adjectives and verbs (in comparison to nouns), for which hypernymy represents a less natural semantic relation.

When comparing our best automatic classification results with human relation distinction, we did not find much overlap in general tendencies. Distinguishing between hypernyms and antonyms/synonyms for nouns worked particularly well, just as distinguishing hypernyms and synonyms for verbs. Overall, this provides a very different story than in the case of human distinctions, where hypernyms and synonyms were consistently distinguished more poorly than the other relation combinations across word classes.

The most interesting insights from the computational perspective arose from comparing the various feature types, where each of them showed rather different strengths and weaknesses. Overall – regarding precision – the pattern-based vector spaces clearly outperformed not only the respective majority baselines but also the two variants of window information (WINDOW-DIFF and WINDOW-PROD) for both 20-word and 5-word windows and across almost all class–relation combinations. When taking a more fine-grained look at the confusion matrices for all 1,528 individual class assignments of relation pairs, the picture was more diverse: The patterns were extremely useful in identifying verbal antonyms, while WINDOW-DIFF was crucial in discovering verbal hypernyms. WINDOW-PROD seemed to generally classify more poorly than the other two feature types; it slightly outperformed them in only one case, for verbal synonyms. WINDOW-DIFF showed a particular strength in that it classified all relation pairs; obviously the vectors were less sparse than for the patterns, and they did not become more sparse in the vector pair creation, differently to WINDOW-PROD vectors.

Overall, even though distributional similarity per se represents a difficult starting point for distinguishing paradigmatic relations (which we illustrated for our dataset), our computational explorations demonstrated that distributional classification models successfully distinguish between them. The most salient feature types

and their success varied across word classes and paradigmatic relation types.

So both for humans and for the automatic approaches, the reliable distinction between relations depends on the specific class–relation combinations. However, easy vs. difficult class–relation combinations differ for humans and computational models, exhibiting stronger ties between ease and naturalness of class-dependent relations for humans than for computational models on the one hand, and strong ties between vector space parameters and relation types on the other hand. For future work on automatic relation distinction, the latter suggests combining feature types (for example, in an ensemble) rather than applying them in isolation.

A

APPENDIX

A.1 *Human distinction of relation pairs in-/excluding ambiguity*

Figures 8 and 9 compare human distinctions of relation pairs excluding ambiguity (left panels, identical to Figures 1 and 2) against human distinctions of relation pairs including ambiguity (right panels). The plots suggest that our conclusions for relation distinction regarding relation pairs excluding ambiguity (cf. Section 4.1) apply similarly to relation pairs including ambiguity: (a) humans clearly distinguish antonyms from synonyms and also from hypernyms, but have more difficulties in distinguishing between synonyms and hypernyms, and (b) distinguishing hypernymy from the other two relations is more difficult for adjectives and verbs (in comparison to nouns), for which hypernymy represents a less natural semantic relation.

A.2 *Cosine similarities between relation pairs*

Figures 10 to 13 illustrate that neither (a) relying on 5-word instead of 20-word window co-occurrences nor (b) relying on lmi scores instead of co-occurrence frequencies nor (c) including ambiguous rela-

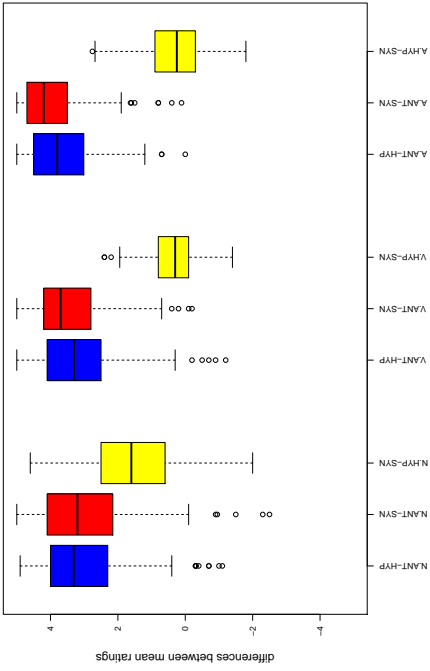
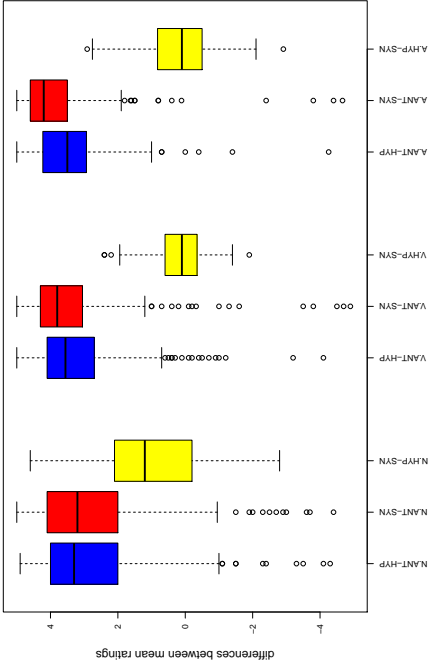


Figure 8: Human distinction of paradigmatic relations (coarse): including vs. excluding ambiguity

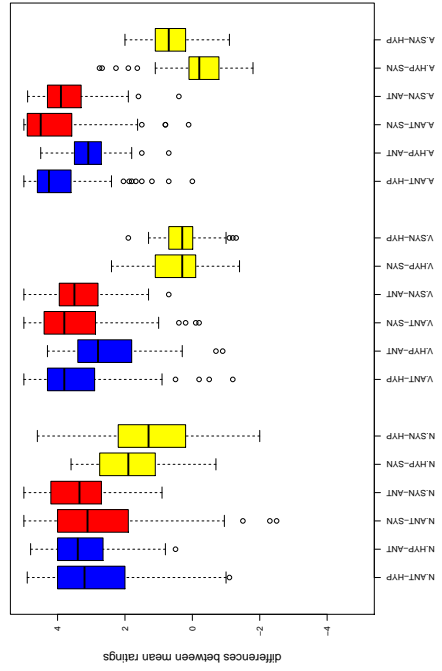
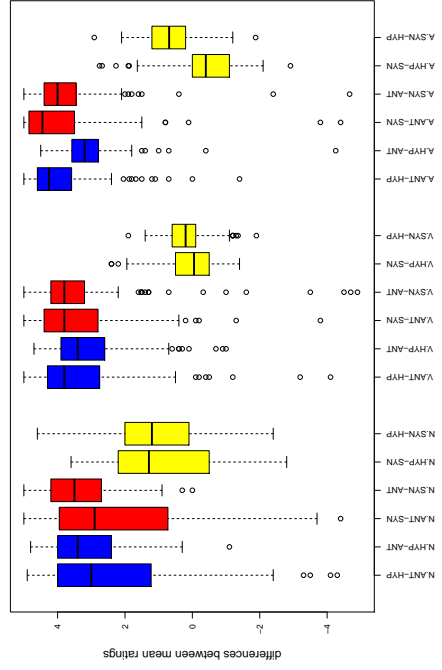


Figure 9: Human distinction of paradigmatic relations (fine): including vs. excluding ambiguity

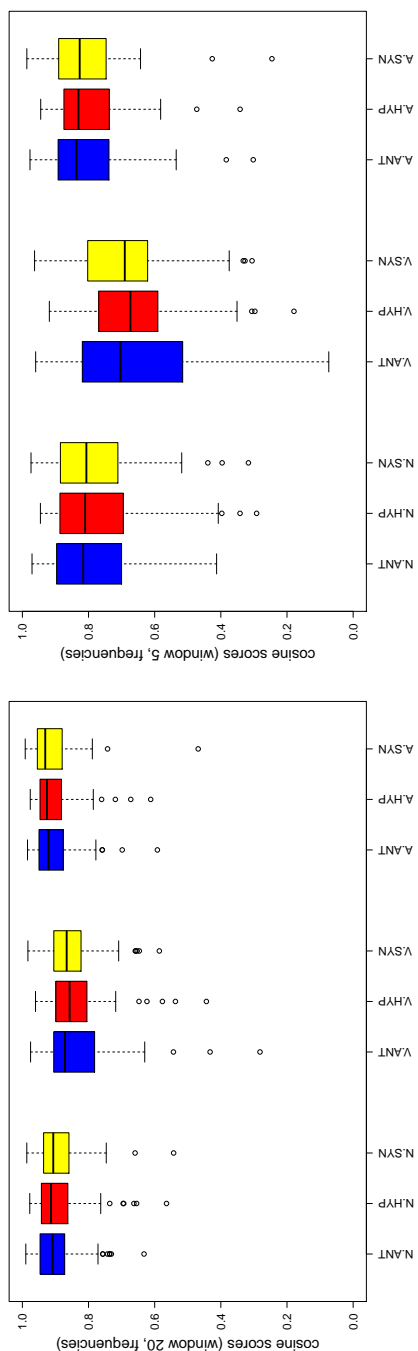


Figure 10: Boxplots of cosine scores across classes and relations (windows: 20 (left) vs. 5, frequencies)

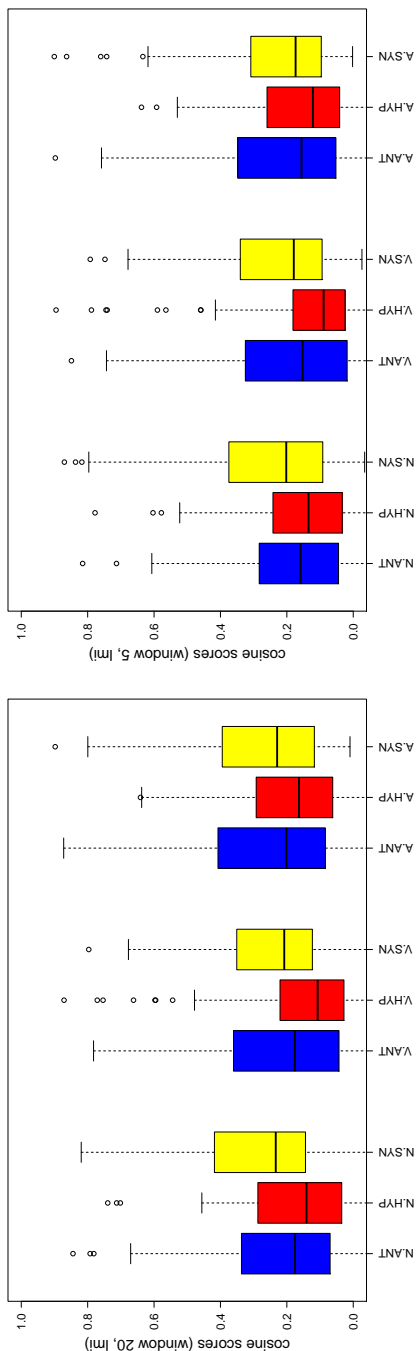


Figure 11: Boxplots of cosine scores across classes and relations (windows: 20 (left) vs. 5, lmi scores)

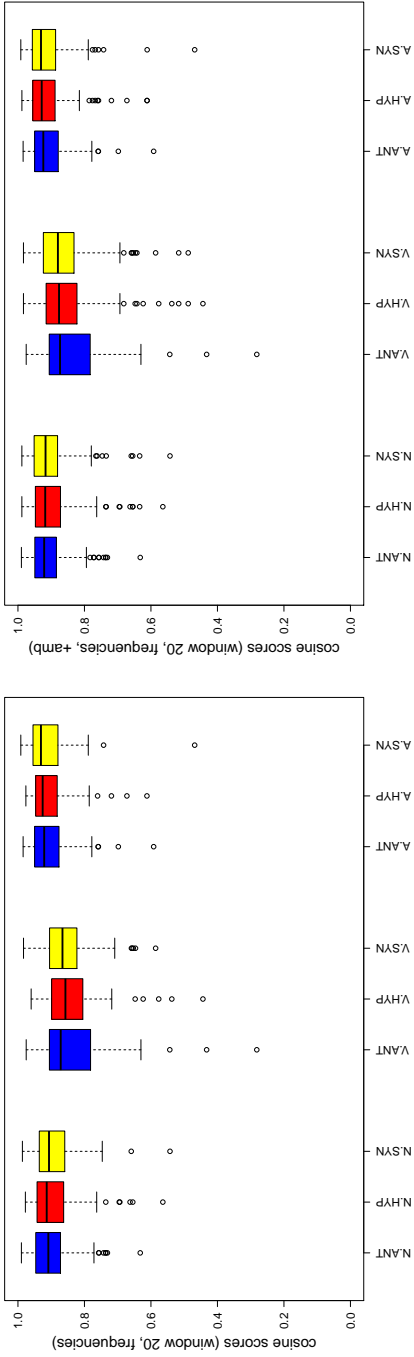


Figure 12: Boxplots of cosine scores across classes and relations (window 20, frequencies, excluding (left) vs. including ambiguity)

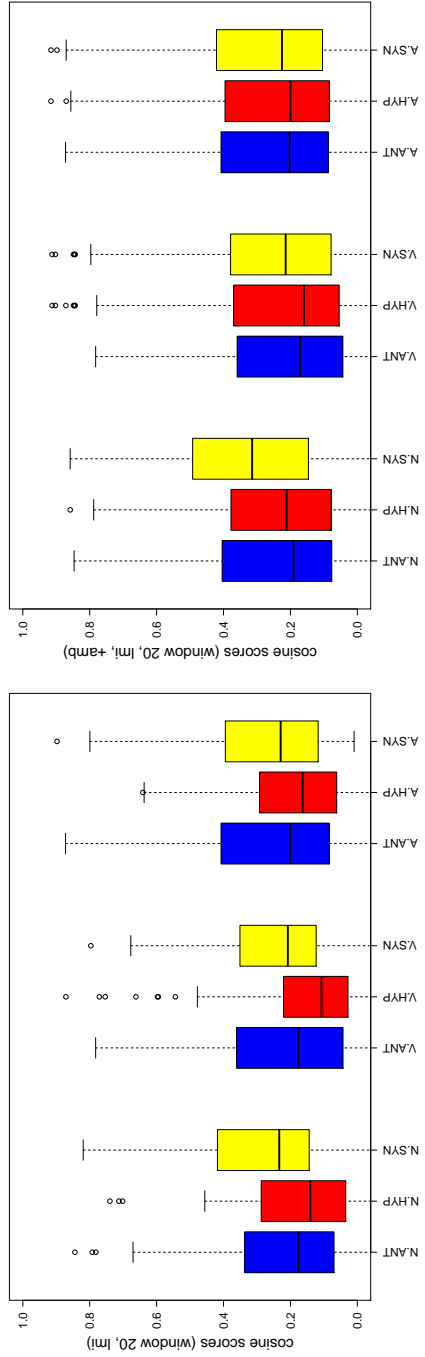
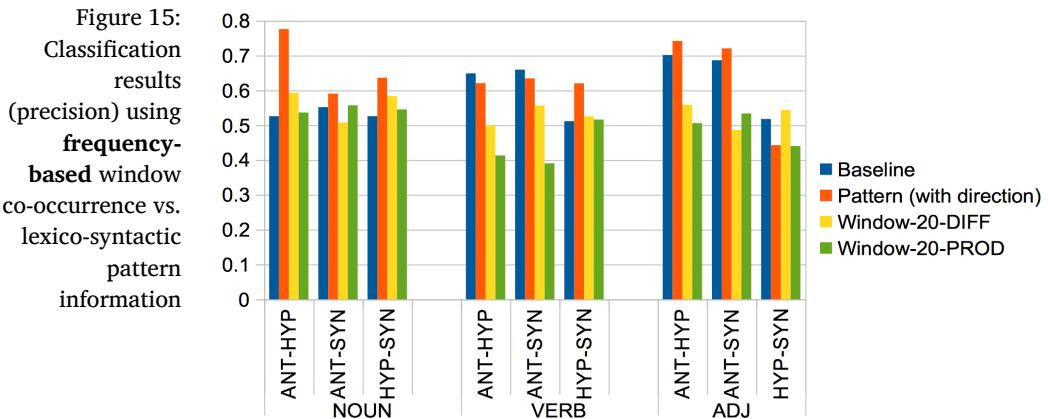
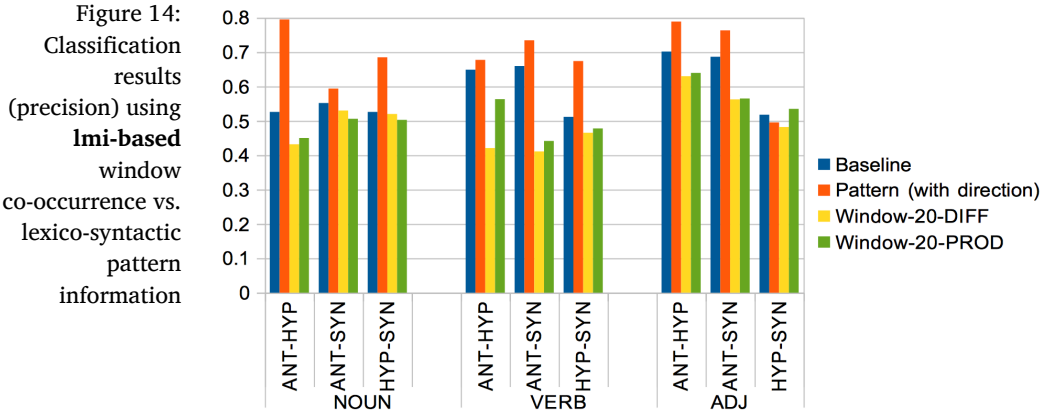


Figure 13: Boxplots of cosine scores across classes and relations (window 20, lmi scores, excluding (left) vs. including ambiguity)

tion pairs changes the overall picture that cosine scores are indeed very similar across our three target paradigmatic relations for a specific word class, cf. our conclusions in Section 4.2.1.

A.3 Automatic classification of relation pairs

Figures 14 and 15 illustrate the differences in classification results when relying on vector spaces with lmi scores (Figure 14, identical to Figure 5) vs. raw frequencies (Figure 15). Using pattern-based features, the plots clearly show consistently better results when using lmi scores in comparison to frequencies. Using window-based features, the results differ more strongly: the WINDOW-20-DIFF results



are better for frequency-based vector spaces than for lmi-based vector spaces, and while they are rather similar to the WINDOW-20-PROD results in the lmi-based spaces, they generally outperform them in the frequency-based spaces.

Figures 16–18 compare lmi-based pattern and window spaces. They once more illustrate that the patterns in Figure 16 (identical to Figure 7) outperform window information, both for a 20-word and a 5-word window. Comparing Figures 17 and 18, we can also see that there are no strong differences regarding the window sizes (20 vs. 5).

The 5-word windows relying on frequencies (right panel in Figure 10) slightly lower the range of the cosine scores, and enlarge the second and third quartiles while the medians stay highly similar, when comparing against the corresponding 20-word windows relying on

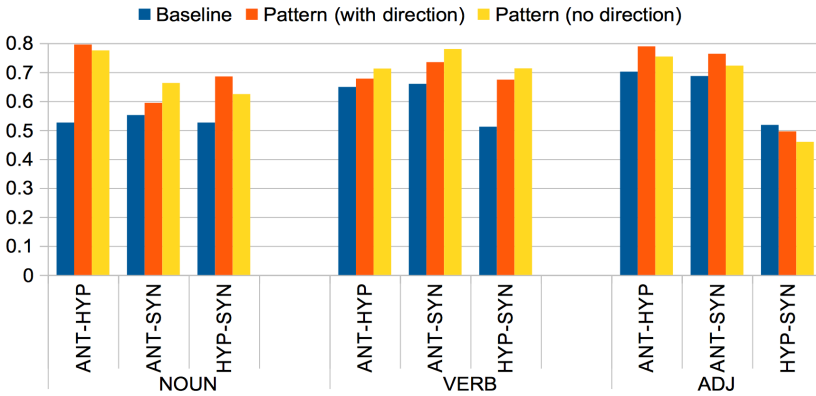


Figure 16: Classification results (precision) using lmi-based **pattern** information

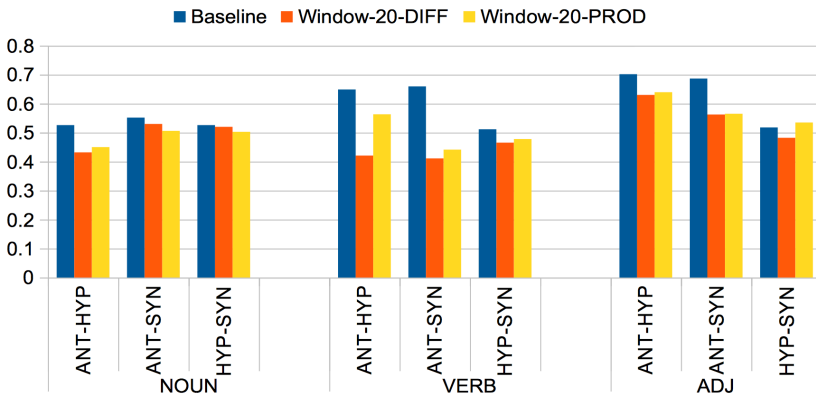
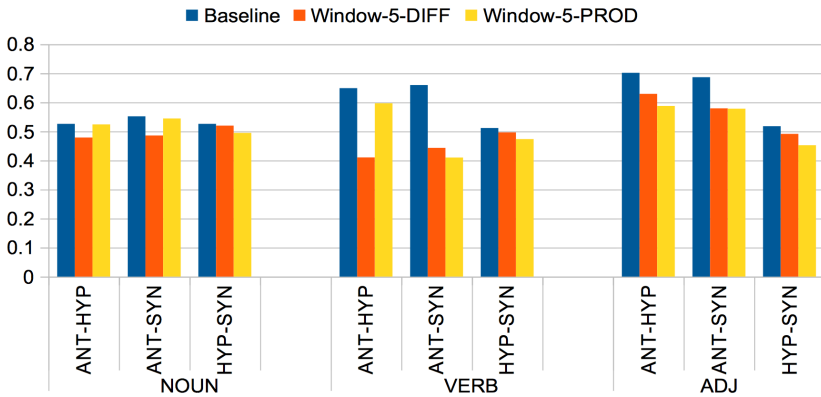


Figure 17: Classification results (precision) using lmi-based **window-20** information

Figure 18:
Classification
results
(precision) using
lmi-based
window-5
information



frequencies (left panel in Figure 10). The lmi scores in comparison to the frequencies strongly influence the magnitudes of the cosine scores, and slightly disperse them (see left and right panels in Figure 11 in comparison to the corresponding ones in Figure 10).

Figures 12 and 13 show for 20-word windows relying on frequencies and lmi scores, respectively, that including ambiguous relation pairs (right panels) hardly changes the overall picture at all, in comparison to the left panels which are identical to those in Figures 10 and 11 and exclude ambiguous pairs.

ACKNOWLEDGEMENTS

The research was supported by the DFG Heisenberg Fellowship SCHU-2580/1, the DFG Research Grant SCHU 2580/2 and the DFG Collaborative Research Centre SFB 732.

REFERENCES

Heike ADEL and Hinrich SCHÜTZE (2014), Using mined coreference chains as a resource for a semantic task, in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1447–1452, Doha, Qatar.

Henriette BAGGER NISSEN and Birgit HENRIKSEN (2006), Word class influence on word association test results, *International Journal of Applied Linguistics*, 16(3):389–408.

Marco BARONI and Alessandro LENCI (2011), How we BLESSED distributional semantic evaluation, in *Proceedings of the EMNLP Workshop on Geometrical Models for Natural Language Semantics*, pp. 1–10, Edinburgh, UK.

Alexander BUDANITSKY and Graeme HIRST (2006), Evaluating WordNet-based measures of lexical semantic relatedness, *Computational Linguistics*, 32(1):13–47.

John A. BULLINARIA and Joseph P. LEVY (2007), Extracting semantic representations from word co-occurrence statistics: a computational study, *Behavior Research Methods*, 39(3):510–526.

Sharon A. CARABALLO (2001), *Automatic acquisition of a hypernym-labeled noun hierarchy from text*, Ph.D. thesis, Brown University.

Roger CHAFFIN and Arnold GLASS (1990), A comparison of hyponym and synonym decisions, *Journal of Psycholinguistic Research*, 19(4):265–280.

Roger CHAFFIN and Douglas HERRMANN (1981), Comprehension of semantic relationships and the generality of categorization models, *Bulletin of the Psychonomic Society*, 17(2):69–72.

Roger CHAFFIN and Douglas HERRMANN (1984), The similarity and diversity of semantic relations, *Memory and Cognition*, 12(2):134–141.

Kai-Wei CHANG, Wen-tau YIH, and Christopher MEEK (2013), Multi-Relational Latent Semantic Analysis, in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1602–1612, Seattle, WA, USA.

Walter CHARLES and George MILLER (1989), Contexts of antonymous adjectives, *Applied Psycholinguistics*, 10:357–375.

Timothy CHKLOVSKI and Patrick PANTEL (2004), VerbOcean: mining the Web for fine-grained semantic verb relations, in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 33–40, Barcelona, Spain.

Philipp CIMIANO, Lars SCHMIDT-THIEME, Aleksander PIVK, and Steffen STAAB (2004), Learning taxonomic relations from heterogeneous evidence, in *Proceedings of the ECAI Workshop on Ontology Learning and Population*, Valencia, Spain.

Herbert H. CLARK (1971), Word associations and linguistic theory, in John LYONS, editor, *New Horizon in Linguistics*, chapter 15, pp. 271–286, Penguin.

D. Allan CRUSE (1986), *Lexical semantics*, Cambridge Textbooks in Linguistics, Cambridge University Press, Cambridge, UK.

James CURRAN (2002), Ensemble methods for automatic thesaurus extraction, in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 222–229, Philadelphia, PA.

- James CURRAN (2003), *From distributional to semantic similarity*, Ph.D. thesis, Institute for Communicating and Collaborative Systems, School of Informatics, University of Edinburgh.
- Benjamin DAVID (2014), *Comparison and combination of feature types for the automatic classification of semantic relation pairs*, Diplomarbeit, Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart.
- Simon DE DEYNE, Daniel J. NAVARRO, and Gert STORMS (2013), Better explanations of lexical and semantic cognition using networks derived from continued rather than single word associations, *Behavior Research Methods*, 45(2):480–498.
- Marie-Catherine DE MARNEFFE, Anna N. RAFFERTY, and Christopher D. MANNING (2008), Finding contradictions in text, in *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 1039–1047, Columbus, OH.
- Ferdinand DE SAUSSURE (1916), *Cours de linguistique générale*, Payot.
- James DEESE (1965), *The structure of associations in language and thought*, The John Hopkins Press, Baltimore, MD.
- Philip EDMONDS (1997), Choosing the word most typical in context using a lexical co-occurrence network, in *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pp. 507–509, Madrid, Spain.
- Philip EDMONDS (1998), Translating near-synonyms: possibilities and preferences in the interlingua, in *Proceedings of the AMTA/SIG-IL 2nd Workshop on Interlinguas*, pp. 23–30, Langhorne, PA.
- Philip EDMONDS (1999), *Semantic representations of near-synonyms for automatic lexical choice*, Ph.D. thesis, Department of Computer Science, University of Toronto, published as technical report CSRI-399.
- Philip EDMONDS and Graeme HIRST (2002), Near-synonymy and lexical choice, *Computational Linguistics*, 28(2):105–144.
- Katrin ERK and Sebastian PADÓ (2008), A structured vector space model for word meaning in context, in *Proceedings of the Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 897–906, Waikiki, Hawaii, USA.
- Stefan EVERT (2005), *The statistics of word co-occurrences: word pairs and collocations*, Ph.D. thesis, Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart.
- Gertrud FAAß and Kerstin ECKART (2013), SdeWaC – a corpus of parsable sentences from the Web, in *Proceedings of the International Conference of the German Society for Computational Linguistics and Language Technology*, pp. 61–68, Darmstadt, Germany.

Distinguishing paradigmatic semantic relations

- Christiane FELLBAUM (1990), English verbs as a semantic net, *Journal of Lexicography*, 3(4):278–301.
- Christiane FELLBAUM (1995), Co-occurrence and antonymy, *Lexicography*, 8(4):281–303.
- Christiane FELLBAUM (1998a), A semantic network of English verbs, in Fellbaum (1998b), pp. 69–104.
- Christiane FELLBAUM, editor (1998b), *WordNet – an electronic lexical database*, Language, Speech, and Communication, MIT Press, Cambridge, MA, USA.
- Christiane FELLBAUM and Roger CHAFFIN (1990), Some principles of the organization of verbs in the mental lexicon, in *Proceedings of the 12th Annual Conference of the Cognitive Science Society of America*, pp. 420–427.
- Lev FINKELSTEIN, Evgeniy GABRILOVICH, Yossi MATIAS, Ehud RIVLIN, Zach SOLAN, Gadi WOLFMAN, and Eytan RUPPIN (2002), Placing search in context: the concept revisited, *ACM Transactions on Information Systems*, 20(1):116–131.
- John R. FIRTH (1957), *Papers in Linguistics 1934-51*, Longmans, London, UK.
- Roxana GIRJU (2003), Automatic detection of causal relations for question answering, in *Proceedings of the ACL Workshop on Multilingual Summarization and Question Answering – Machine Learning and Beyond*, pp. 76–83, Sapporo, Japan.
- Roxana GIRJU, Adriana BADULESCU, and Dan MOLDOVAN (2006), Automatic discovery of part-whole relations, *Computational Linguistics*, 32(1):83–135.
- Derek GROSS, Ute FISCHER, and George A. MILLER (1989), Antonymy and the representation of adjectival meanings, *Memory and Language*, 28(1):92–106.
- Derek GROSS and Katherine J. MILLER (1990), Adjectives in WordNet, *International Journal of Lexicography*, 3(4):265–277.
- Annamaria GUIDA and Alessandro LENCI (2007), Semantic properties of word associations to Italian verbs, *Italian Journal of Linguistics*, 19(2):293–326.
- Iryna GUREVYCH (2005), Using the structure of a conceptual network in computing semantic relatedness, in *Proceedings of the 2nd International Joint Conference on Natural Language Processing*, pp. 767–778, Jeju Island, Korea.
- Birgit HAMP and Helmut FELDWEG (1997), GermaNet – a lexical-semantic net for German, in *Proceedings of the ACL Workshop on Automatic Information Extraction and Building Lexical Semantic Resources for NLP Applications*, pp. 9–15, Madrid, Spain.
- Sanda M. HARABAGIU, Andrew HICKL, and Finley LACATUSU (2006), Negation, contrast and contradiction in text processing, in *Proceedings of the 21st National Conference on Artificial Intelligence*, pp. 755–762, Boston, MA, USA.
- Zellig HARRIS (1954), Distributional Structure, *Word*, 10(23):146–162.

- Marti HEARST (1992), Automatic acquisition of hyponyms from large text corpora, in *Proceedings of the 14th International Conference on Computational Linguistics*, pp. 539–545, Nantes, France.
- Hans Jürgen HERINGER (1986), The verb and its semantic power: association as the basis for valence, *Journal of Semantics*, 4:79–99.
- Felix HILL, Roi REICHART, and Anna KORHONEN (2015), SimLex-999: evaluating semantic models with (genuine) similarity estimation, *Computational Linguistics*, 41(4):665–695.
- Steven JONES, M. Lynne MURPHY, Carita PARADIS, and Caroline WILLNERS (2012), *Antonyms in English: construals, constructions and canonicity*, Studies in English Language, Cambridge University Press, Cambridge, UK.
- John S. JUSTESON and Slava M. KATZ (1991), Co-occurrence of antonymous adjectives and their contexts, *Computational Linguistics*, 17:1–19.
- John S. JUSTESON and Slava M. KATZ (1992), Redefining antonymy: the textual structure of a semantic relation, *Literary and Linguistic Computing*, 7(3):176–184.
- Grace Helen KENT and Aaron J. ROSANOFF (1910), A study of association in insanity, *American Journal of Insanity*, 67(37–96):317–390.
- George R. KISS, Christine ARMSTRONG, Robert MILROY, and James PIPER (1973), An associative thesaurus of English and its computer analysis, in *The computer and literary studies*, Edinburgh University Press, <http://www.eat.rl.ac.uk/>.
- Claudia KUNZE (2000), Extension and use of GermaNet, a lexical-semantic database, in *Proceedings of the 2nd International Conference on Language Resources and Evaluation*, pp. 999–1002, Athens, Greece.
- Claudia KUNZE and Andreas WAGNER (1999), Integrating GermaNet into EuroWordNet, a multilingual lexical-semantic database, *Sprache und Datenverarbeitung*, 23(2):5–19.
- Thomas K. LANDAUER and Susan T. DUMAIS (1997), A solution to Plato's Problem: the latent semantic analysis theory of acquisition, induction and representation of knowledge, *Psychological Review*, 104(2):211–240.
- Adrienne LEHRER and Keith LEHRER (1982), Antonymy, *Linguistics and Philosophy*, 5:483–501.
- Lothar LEMNITZER and Claudia KUNZE (2007), *Computerlexikographie*, Gunter Narr Verlag, Tübingen, Germany.
- Alessandro LENCI and Giulia BENOTTO (2012), Identifying hypernyms in distributional semantic spaces, in *Proceedings of the 1st Joint Conference on Lexical and Computational Semantics*, pp. 75–79, Montréal, Canada.

Omer LEVY, Steffen REMUS, Chris BIEMANN, and Ido DAGAN (2015), Do supervised distributional methods really learn lexical inference relations?, in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 970–976.

Dekang LIN, Shaojun ZHAO, Lijuan QIN, and Ming ZHOU (2003), Identifying synonyms among distributionally similar words, in *Proceedings of the International Conferences on Artificial Intelligence*, pp. 1492–1493, Acapulco, Mexico.

Cupertino LUCERTO, David PINTO, and Héctor JIMÉNEZ-SALAZAR (2004), An automatic method to identify antonymy relations, in *Proceedings of the IBERAMIA Workshop on Lexical Resources and the Web for Word Sense Disambiguation*, pp. 105–111, Puebla, Mexico.

John LYONS (1968), *Introduction to theoretical linguistics*, Cambridge University Press, Cambridge, England.

John LYONS (1977), *Semantics*, Cambridge University Press, Cambridge, UK.

Christopher D. MANNING, Prabhakar RAGHAVAN, and Hinrich SCHÜTZE (2008), *Introduction to information retrieval*, Cambridge University Press, Cambridge, UK.

Tomas MIKOLOV, Wen TAU YIH, and Geoffrey ZWEIG (2013), Linguistic regularities in continuous space word representations, in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 746–751, Atlanta, GA, USA.

George A. MILLER, Richard BECKWITH, Christiane FELLBAUM, Derek GROSS, and Katherine J. MILLER (1990), Introduction to WordNet: an on-line lexical database, *International Journal of Lexicography*, 3(4):235–244.

George A. MILLER and Walter G. CHARLES (1991), Contextual correlates of semantic similarity, *Language and Cognitive Processes*, 6(1):1–28.

George A. MILLER and Christiane FELLBAUM (1991), Semantic networks of English, *Cognition*, 41:197–229.

Saif MOHAMMAD, Bonnie DORR, and Graeme HIRST (2008), Computing word-pair antonymy, in *Proceedings of the Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 982–991, Waikiki, Hawaii, USA.

Saif M. MOHAMMAD, Bonnie J. DORR, Graeme HIRST, and Peter D. TURNEY (2013), Computing lexical contrast, *Computational Linguistics*, 39(3).

Saif M. MOHAMMAD, Iryna GUREVYCH, Graeme HIRST, and Torsten ZESCH (2007), Cross-lingual distributional profiles of concepts for measuring semantic distance, in *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 571–580, Prague, Czech Republic.

Gregory L. MURPHY and Jane M. ANDREW (1993), The conceptual basis of antonymy and synonymy in adjectives, *Journal of Memory and Language*, 32(3):1–19.

M. Lynne MURPHY (2003), *Semantic relations and the lexicon*, Cambridge University Press, Cambridge, UK.

Douglas L. NELSON, Cathy L. MCEVOY, and Thomas A. SCHREIBER (1998), The University of South Florida word association, rhyme, and word fragment norms, <http://www.usf.edu/FreeAssociation/>.

Kim Anh NGUYEN, Maximilian KÖPER, Sabine SCHULTE IM WALDE, and Ngoc Thang VU (2017), Hierarchical embeddings for hypernymy detection and directionality, in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 233–243, Copenhagen, Denmark.

Kim-Anh NGUYEN, Sabine SCHULTE IM WALDE, and Thang VU (2016a), Integrating distributional lexical contrast into word embeddings for antonym-synonym distinction, in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pp. 454–459, Berlin, Germany.

Kim-Anh NGUYEN, Sabine SCHULTE IM WALDE, and Thang VU (2016b), Neural-based noise filtering from word embeddings, in *Proceedings of the 26th International Conference on Computational Linguistics*, pp. 2699–2707, Osaka, Japan.

Masataka ONO, Makoto MIWA, and Yutaka SASAKI (2015), Word embedding-based antonym detection using thesauri and distributional information, in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 984–989, Denver, Colorado, USA.

David S. PALERMO and James J. JENKINS (1964), *Word association norms: grade school through college*, University of Minnesota Press, Minneapolis, USA.

Patrick PANTEL and Marco PENNACCHIOTTI (2006), Espresso: leveraging generic patterns for automatically harvesting semantic relations, in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pp. 113–120, Sydney, Australia.

Carita PARADIS (2016), Corpus methods for the investigation of antonyms across languages, in Päivi V. JUVONEN and Maria KOPTJEVSKAJA-TAMM, editors, *The lexical typology of semantic shifts*, volume 58 of *Cognitive Linguistics Research*, pp. 131–156, Mouton de Gruyter.

Carita PARADIS, Caroline WILLNERS, and Steven JONES (2009), Good and bad opposites: using textual and experimental techniques to measure antonym canonicity, *The Mental Lexicon*, 4(3):380–429.

- Nghia The PHAM, Angeliki LAZARIDOU, and Marco BARONI (2015), A multitask objective to inject lexical contrast into distributional semantics, in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pp. 21–26, Beijing, China.
- Philip RESNIK (1995), Using information content to evaluate semantic similarity in a taxonomy, in *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pp. 448–453, San Francisco, CA, USA.
- Michael ROTH and Sabine SCHULTE IM WALDE (2014), Combining word patterns and discourse markers for paradigmatic relation classification, in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pp. 524–530, Baltimore, MD, USA.
- Herbert RUBENSTEIN and John B. GOODENOUGH (1965), Contextual correlates of synonymy, *Communications of the ACM*, 8(10):627–633.
- Wallace A. RUSSELL (1970), The complete German language norms for responses to 100 words from the Kent-Rosanoff Word Association Test, in Leo POSTMAN and Geoffrey KEPPEL, editors, *Norms of word association*, pp. 53–94, Academic Press, New York, USA.
- Wallace A. RUSSELL and O.R. MESECK (1959), Der Einfluss der Assoziation auf das Erinnern von Worten in der deutschen, französischen und englischen Sprache, *Zeitschrift für Experimentelle und Angewandte Psychologie*, 6:191–211.
- Enrico SANTUS, Alessandro LENCI, Qin LU, and Sabine SCHULTE IM WALDE (2014a), Chasing hypernyms in vector spaces with entropy, in *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 38–42, Gothenburg, Sweden.
- Enrico SANTUS, Qin LU, Alessandro LENCI, and Chu-Ren HUANG (2014b), Taking antonymy mask off in vector space, in *Proceedings of the 28th Pacific Asia Conference on Language, Information and Computation*, Phuket, Thailand.
- Enrico SANTUS, Qin LU, Alessandro LENCI, and Chu-Ren HUANG (2014c), Unsupervised antonym-synonym discrimination in vector space, in *Atti della Conferenza di Linguistica Computazionale Italiana*, Pisa, Italy.
- Enrico SANTUS, Frances YUNG, Alessandro LENCI, , and Chu-Ren HUANG (2015), EVALution 1.0: an evolving semantic dataset for training and evaluation of distributional semantic models, in *Proceedings of the 4th Workshop on Linked Data in Linguistics*, pp. 64–69, Beijing, China.
- Roland SCHÄFER and Felix BILDHAUER (2012), Building large corpora from the Web using a new efficient tool chain, in *Proceedings of the 8th International Conference on Language Resources and Evaluation*, pp. 486–493, Istanbul, Turkey.
- Silke SCHEIBLE and Sabine SCHULTE IM WALDE (2014), A database of paradigmatic semantic relation pairs for German nouns, verbs and adjectives, in

Proceedings of the COLING Workshop on Lexical and Grammatical Resources for Language Processing, pp. 111–119, Dublin, Ireland.

Silke SCHEIBLE, Sabine SCHULTE IM WALDE, and Sylvia SPRINGORUM (2013), Uncovering distributional differences between synonyms and antonyms in a word space model, in *Proceedings of the 6th International Joint Conference on Natural Language Processing*, pp. 489–497, Nagoya, Japan.

Helmut SCHMID (1994), Probabilistic part-of-speech tagging using decision trees, in *Proceedings of the 1st International Conference on New Methods in Language Processing*.

Sebastian SCHMIDT, Philipp SCHOLL, Christoph RENSING, and Ralf STEINMETZ (2011), Cross-lingual recommendations in a resource-based learning scenario, in *Proceedings of the 6th European Conference on Technology Enhanced Learning*, pp. 356–369, Palermo, Italy.

Sabine SCHULTE IM WALDE and Maximilian KÖPER (2013), Pattern-based distinction of paradigmatic relations for German nouns, verbs, adjectives, in *Proceedings of the 25th International Conference of the German Society for Computational Linguistics and Language Technology*, pp. 184–198, Darmstadt, Germany.

Sabine SCHULTE IM WALDE, Alissa MELINGER, Michael ROTH, and Andrea WEBER (2008), An empirical characterisation of response types in German association norms, *Research on Language and Computation*, 6(2):205–238.

Vered SHWARTZ, Yoav GOLDBERG, and Ido DAGAN (2016), Improving hypernymy detection with an integrated path-based and distributional method, in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pp. 2389–2398, Berlin, Germany.

Rion SNOW, Daniel JURAFSKY, and Andrew Y. NG (2004), Learning syntactic patterns for automatic hypernym discovery, *Advances in Neural Information Processing Systems*, 17:1297–1304.

Rion SNOW, Daniel JURAFSKY, and Andrew Y. NG (2006), Semantic taxonomy induction from heterogenous evidence, in *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pp. 801–808, Sydney, Australia.

Peter D. TURNEY (2008), A uniform approach to analogies, synonyms, antonyms, and associations, in *Proceedings of the 22nd International Conference on Computational Linguistics*, pp. 905–912, Manchester, UK.

Peter D. TURNEY and Patrick PANTEL (2010), From frequency to meaning: vector space models of semantics, *Journal of Artificial Intelligence Research*, 37:141–188.

Lonneke VAN DER PLAS and Jörg TIEDEMANN (2006), Finding synonyms using automatic word alignment and measures of distributional similarity, in *Proceedings of the 21st International Conference on Computational Linguistics and*

Distinguishing paradigmatic semantic relations

the 44th Annual Meeting of the Association for Computational Linguistics, pp. 866–873, Sydney, Australia.

Paola VELARDI, Paolo FABRIANI, and Michele MISSIKOFF (2001), Using text processing techniques to automatically enrich a domain ontology, in *Proceedings of the International Conference on Formal Ontology in Information Systems*, pp. 270–284, Ogunquit, ME.

Julie WEEDS, David WEIR, and Diana MCCARTHY (2004), Characterising measures of lexical distributional similarity, in *Proceedings of the 20th International Conference of Computational Linguistics*, pp. 1015–1021, Geneva, Switzerland.

Morton E. WINSTON, Roger CHAFFIN, and Douglas HERRMANN (1987), A taxonomy of part-whole relations, *Cognitive Science*, 11:417–444.

Willy YAP and Timothy BALDWIN (2009), Experiments on pattern-based relation learning, in *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, pp. 1657—1660, Hong Kong.

Wen-Tau YIH, Geoffrey ZWEIG, and John C. PLATT (2012), Polarity inducing latent semantic analysis, in *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 1212–1222, Jeju Island, Korea.

Sabine Schulte im Walde

© 0000-0002-8975-6255

schulte@ims.uni-stuttgart.de

Institut für Maschinelle Sprachverarbeitung
Universität Stuttgart
Pfaffenwaldring 5B
70569 Stuttgart
Germany

Sabine Schulte im Walde (2020), *Distinguishing between paradigmatic semantic relations across word classes: human ratings and distributional similarity*, *Journal of Language Modelling*, 8(1):53–101

doi <https://dx.doi.org/10.15398/jlm.v8i1.199>

This work is licensed under the *Creative Commons Attribution 4.0 Public License*.

cc  <http://creativecommons.org/licenses/by/4.0/>

Neural network models for phonology and phonetics

Paul Boersma¹, Titia Benders², and Klaas Seinhorst¹

¹ University of Amsterdam

² Macquarie University

ABSTRACT

This paper^{1,2} argues that if phonological and phonetic phenomena found in language data and in experimental data all have to be accounted for within a single framework, then that framework will have to be based on neural networks. We introduce an artificial neural network model that can handle stochastic processing in production and comprehension. With the “inoutstar” learning algorithm, the model is

Keywords:
phonology,
neural networks,
speech perception,
historical
linguistics

¹Parts of this work were presented at the 31st Annual Meeting of the Deutsche Gesellschaft für Sprachwissenschaft, Osnabrück, March 2009; the 45th Annual Meeting of the Chicago Linguistic Society, April 2009; the KNAW Academy Colloquium on Language Acquisition and Optimality Theory, Amsterdam, July 2009; the 5th International Conference on Native and Non-native Accents of English, Łódź, December 2011; the 10th International Conference on Computational Processing of Portuguese Language in Coimbra, April 2012; the 20th Manchester Phonology Meeting, May 2012; the 2012 International Child Phonology Conference, Minneapolis, June 2012; the 19th Frysk Filologekongres, Ljouwert, June 2012; the 13th Conference on Laboratory Phonology, Stuttgart, July 2012; and the EGG Summerschool, Wrocław, August 2012. We thank the audiences, as well as Silke Hamann and Kateřina Chládková for their input. The research was sponsored by NWO grant 277-70-008 to Boersma, and NWO grant 021.002.095 to Benders.

²The learning algorithm developed in this manuscript has subsequently been used and/or elaborated by Benders (2013); Chládková (2014); ter Schure *et al.* (2016); and Seinhorst *et al.* (2019).

A model of phonological and phonetic representations and knowledge

1.1

If the model contains levels of representation, it may look like Figure 1, which can be thought of as containing the minimum number of levels needed for a sensible description: phonetics seems to require at least an Auditory Form (AudF, specifying a continuous stream of sound) and an Articulatory Form (ArtF, specifying muscle activities), and phonology seems to require at least an Underlying Form (UF, containing at least lexically contrastive material) and a Surface Form (SF, containing a whole utterance divided up in prosodic structure such as syllables); the Morpheme level connects the phonology to the syntax and the semantics in the lexicon.

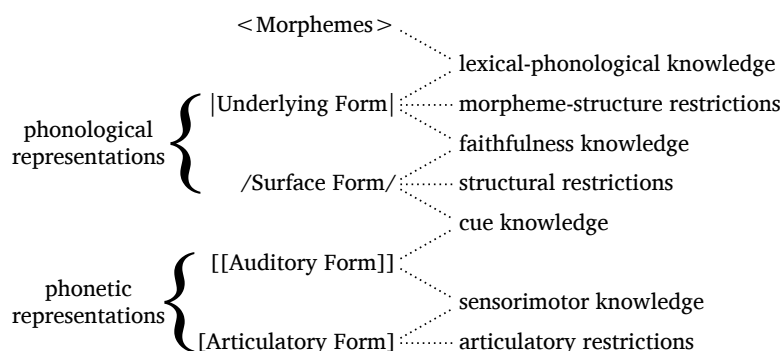


Figure 1: Levels of representation and stored knowledge in a model of phonology and phonetics

The five levels in Figure 1 are a simplified combination of what phonologists have been proposing in models of phonological production (e.g. van Wijk 1936: 323; Trubetzkoy 1939; Kiparsky 1982) and what psycholinguists have been proposing in models of comprehension (e.g. McClelland and Elman 1986; Cutler *et al.* 1987) and production (e.g. Levelt *et al.* 1999). These specific five levels, and the special way in which they are connected in Figure 1, were proposed by Boersma (1998, 2007) and Apoussidou (2007). In numerous papers, Boersma and co-workers have investigated the capability of this “Bidirectional Phonology and Phonetics” (BiPhon) model to account for experimental as well as linguistic data (for an overview, see Boersma 2011). The model has hitherto used the decision mechanism of Optimality Theory (OT) and can therefore be called BiPhon-OT.

The present paper introduces the neural-network (NN) edition of the model, which we call BiPhon-NN.

Language users have knowledge of the relationships between levels of representation. In Figure 1, such relationships exist between adjacent levels only, so that the language user has knowledge about sensorimotor, cue, faithfulness (phonological) and lexical relationships. The language user also has knowledge about restrictions within levels: the articulatory, structural and morpheme-structure restrictions. In OT, all this knowledge is represented as a grammar consisting of ranked constraints; in NN models, this knowledge is represented as a long-term memory consisting of connection weights.

1.2

Phonological and phonetic processes

A comprehensive model has to take into account the behaviour of the speaker, the listener, and the learner. Figure 2 shows the various *processes* that can be distinguished when travelling the levels of representation of Figure 1. Globally, the path from AudF to Morphemes following the upward arrows in Figure 2 is *comprehension*, i.e. the task of the listener, and the path from Morphemes to ArtF following the downward arrows is *production*, the task of the speaker. More locally, there are partial processes. The local mapping from UF to SF is *phonological production*, an example being the mapping from an underlying two-word sequence |an#pa| (“#” denotes a word boundary) to the phonological surface structure /.am.pa./ (“.” denotes a syllable boundary) in a language with nasal place assimilation. At the interface between phonetics and phonology, the local mapping from AudF to SF is (prelexical) *perception*, an example being the mapping from concrete continuous formant values to abstract discrete vowel categories.

The partial processes and their acquisition have been modelled in various frameworks. Phonologists have been modelling phonological production within OT since Prince and Smolensky (1993/2004), and its acquisition since Tesar and Smolensky (1998). Word recognition was modelled with neural networks by Norris (1994) in the Shortlist model, and prelexical perception was modelled with neural networks by Weenink (2006) and within BiPhon-OT by Boersma (1997) and Escudero and Boersma (2004). The present paper in Section 5 models

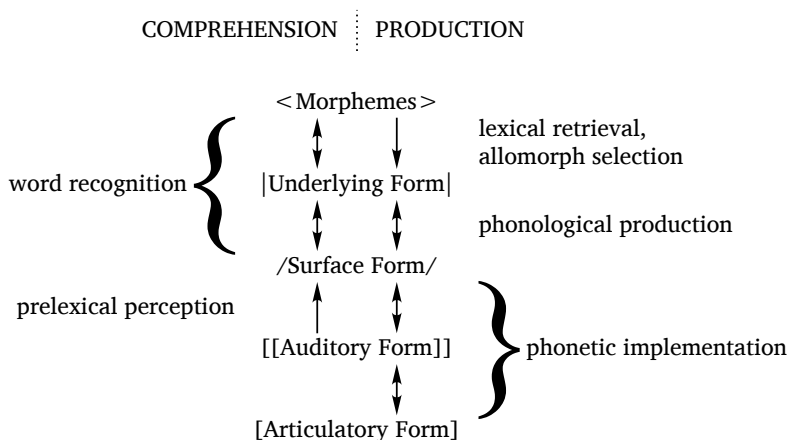


Figure 2: Processes in a comprehensive model of phonology and phonetics

the development of *category creation* in the AudF-to-SF mapping. The emergence of an early stage of category creation, namely the perceptual magnet effect (which was observed in the lab by Kuhl 1991), has been modelled before with neural networks by Guenther and Gjaja (1996) and with BiPhon-OT by Boersma *et al.* (2003).

The way in which the language user’s knowledge is represented in Figure 1 suggests that the same knowledge is used for both directions of processing in Figure 2, i.e. for comprehending and producing speech. Within OT, this *bidirectionality* was first argued for by Smolensky (1996). Specifically, it has often been argued that the same structural constraints play a role in comprehension as well as in production (Tesar 1997; Tesar and Smolensky 1998, 2000; Boersma 1998, 2000, 2007, 2009; Pater 2004), sometimes with very dissimilar effects (Boersma and Hamann 2009). For the present paper it is relevant that the “cue knowledge” at the interface of phonology and phonetics is bidirectional, i.e. used in both prelexical perception and phonetic implementation (Boersma 2009): the same knowledge that allows one to perceive a loud high-frequency noise as /s/ forces one to implement the surface phoneme /s/ as a sound with a loud high-frequency noise. In Section 6 we model within BiPhon-NN the acquisition of *auditory dispersion*, i.e. the evolution of optimal distances at AudF between the members of phoneme inventories at SF. This acquisition has been modelled before within exemplar theory by Wedel (2004, 140–169; 2006, 261–269) and in BiPhon-OT by Boersma and Hamann (2008); in both cases, bidirectionality

was a crucial element of the explanation, as explained in detail in Section 6.

Thus, the perceptual magnet effect and auditory dispersion have both been modelled before, although rarely within the same framework (with BiPhon-OT as a possible exception).

1.3 *The need to model it all at the same time*

There are at least two reasons why one would want to model all the processes of Section 1.2 within a single comprehensive model. One reason is that there are phenomena whose complete explanation necessarily requires all levels of representation, and the other reason is that there seem to exist processes that require an interaction between levels that are far away from each other in Figure 1 or 2. We discuss these reasons now, with the goal of finding candidate comprehensive modelling frameworks.

1.3.1 Comprehensive processes

There exist seemingly unitary processes whose explanation nevertheless requires all levels of representation. One such process is loanword adaptation, where the input (the foreign stream of sound that impinges on the borrower's ear) and the output (the borrower's phonetic production) are the only direct observables. If one wants to understand this phenomenon solely on the basis of acquired L1 behaviour, one has to assume that the borrower starts by filtering the incoming auditory form through L1-specific cue knowledge and L1-specific structural constraints into a phonological surface structure (see Figures 1 and 2), then stores it as a new morpheme in the lexicon with an appropriate underlying form. When speaking, the borrower takes this morpheme and underlying form, filters the latter with her L1-specific phonological knowledge, then filters the result again with her phonetic implementation device, which computes an auditory form and an articulatory form, perhaps filtered by L1-specific articulatory restrictions. An explanation of loanword adaptation, therefore, requires all arrows in Figure 2, as has been argued in detail by Boersma and Hamann (2009). Another phenomenon whose explanation requires all levels of representation is first-language acquisition. This happens

much more slowly than the initial adaptation of a loanword, but is also much more central to linguistic theory and experimentation. The search we have to embark on, therefore, is for a single comprehensive framework.

Distant interactions

1.3.2

The arrows in Figure 2 only connect levels that are adjacent. Thus, an incoming sound at AudF first activates a representation at SF, which then activates a representation at UF, which then activates one or more morphemes at the topmost level; there are no more direct routes that skip a level.

However, there is evidence that the partial processes are not entirely sequential. Feedback from “later” to “earlier” levels of representation has been identified experimentally and theoretically in several locations of processing, and several models that exhibit such interactions have already been proposed. In comprehension, lexical influence (from the Morpheme level) back to prelexical perception (AudF-to-SF) was attested by Ganong III (1980), who found that an auditory sequence that is ambiguous between /dæ/ or /tæ/ (for English listeners) is perceived as /dæ/ if followed by [ʃ] and as /tæ/ if followed by [sk], simply because |dæʃ| and |tæsk| correspond to English words, while |tæʃ| and |dæsk| do not; this effect was modelled with neural networks by McClelland and Elman (1986) and with BiPhon-OT by Boersma (2009, 2011). Likewise, semantic considerations above the Morpheme influence the access of underlying forms in the mapping from SF to UF (Warren and Warren 1970). In production, allomorph selection at UF or higher is sometimes determined by “later” considerations at SF: the choice between |vjø| and |vjej| ‘old-MASC’ in French is determined by whether the next word happens to start with a consonantal segment or not, as modelled with BiPhon-OT by Boersma and van Leussen (2017). Likewise, phonetic considerations such as articulatory effort (at ArtF) and cue quality (between SF and AudF) may influence choices in the phonology (between UF and SF), as modelled by Boersma (1998, 2007). Also, cue knowledge and articulatory constraints must interact with each other in the phonetic implementation process.

As a result of these examples of *interactive* processing, most of the arrows in Figure 2 are two-sided. Levels that are activated “later”

in comprehension or production can thereby influence “earlier” levels backwards. In NN models, interactivity is implemented by having activity spread bidirectionally (McClelland and Elman 1986); in BiPhon-OT the interactivity is implemented by having candidates be entire paths from AudF to Morpheme in comprehension or from Morpheme to ArtF in production (Boersma 2007, 2009, 2011; Apoussidou 2007; Berent *et al.* 2009).

The existence of such feedback in processing is controversial in some locations (Norris *et al.* 2000 deny the influence of the lexicon on prelexical perception, and Hale and Reiss 2000 deny any influence of phonetic considerations on phonological production). For the time being, however, we assume interactivity is everywhere. The need for a comprehensive model does not depend on whether such interactivity is only apparent or is an integral element of the underlying mechanism.

1.4

Choosing the framework that models it all: neural networks

When discussing existing models in Section 1.1 through Section 1.3, we identified three frameworks: neural networks, exemplar theory, and OT.

At first sight, BiPhon-OT might seem to be the best framework, because it provided an account of all of the processes mentioned. However, this is deceptive, because it did not provide an account of all the processes *combined*. When modelling phonological category creation (Boersma 1998: ch.8; Boersma *et al.* 2003), the BiPhon model shares with NN category creation models (Guenther and Gjaja 1996) the assumption that phonological categories emerge from the distributions of auditory forms in the child’s environment. Both computational models successfully arrive at a stage of continuous perceptual warping (an incoming sound is received as a slightly different sound because of distributional learning), but linguistic modelling in OT has to stop there, because it has to assume that categories are discrete. This discrepancy between the gradience (continuity) of category creation that is needed in an emergentist model, and the discreteness of categories that is needed to do OT phonology entails the failure of OT

as a comprehensive framework for emergentist phonology and phonetics. Moreover, OT's biological plausibility is low, because it works with nearly infinite lists of candidates, which is especially problematic if we have five levels of representation; typically, the number of candidate paths to evaluate is exponential in the length of the input (both in comprehension and in production) as well as exponential in the number of levels of representation.

Superficially, exemplar theory (Goldinger 1996) might be expected to do better with respect to a transition from continuous to discrete, because this theory can at least be seen to handle the reverse transition when massive storage of single discrete events leads to observed continuous knowledge. However, despite its long existence, the theory has not yet been able to model even the most straightforward of phonological processes, such as productive nasal place assimilation (Boersma 2012). More crucially, work specifically addressing the acquisition of categories (Kruschke 1992; Pierrehumbert 2001) presupposes pre-existing category labels, i.e. it models the emergence of the link between categories and sound but not the emergence of the category labels themselves.

This leaves neural network modelling as the only option. If Figure 1 is implemented in a neural network, each of the five levels of representation should be thought of as a large set of network *nodes*, each of which can be active or inactive (or, in a time-smoothed view, more active or less active). The pattern of activity of these nodes forms the current representation at that level. The processes of Figure 2 can be regarded as the spreading of activity between and within levels; the knowledge in Figure 1 is stored as connection weights, i.e. the strengths of the connections between the nodes. We show in Section 5 that if the elements of representations are distributed over multiple nodes, they can start out as continuous and gradually come to exhibit more discrete behaviour during acquisition, thus ensuring the compatibility between underlying continuity and observed discreteness. One and the same framework, then, succeeds in accounting for both symbolic and subsymbolic behaviour. As far as biological plausibility goes, neural networks form the best of the three frameworks as well: the number of connections in a NN model tends to rise linearly with the number of levels of representation, and linearly or quadratically with the size of the representations.

Following Figure 2, phonological production, viewed in isolation, is the mapping from Underlying Form (UF) to Surface Form (SF). Using terms that are familiar from both the neural network literature (Rosenblatt 1958) and OT (Prince and Smolensky 1993/2004, Section 1.1), the Underlying Form is the *input* of this mapping and the Surface Form is the *output*. For simplification, we start with a toy language that models only the relationship between UF and SF, although we do so in both directions of processing.

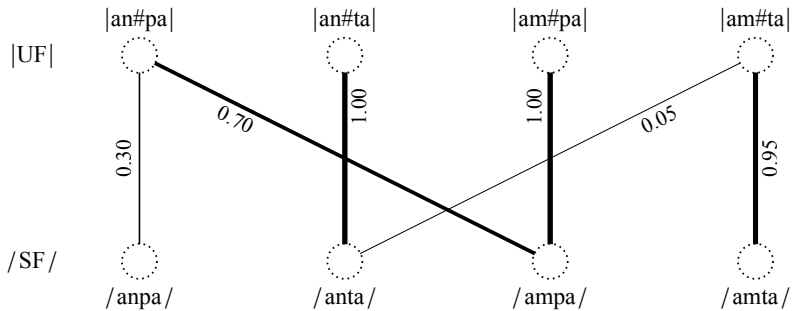
Our toy language has only four possible underlying utterances, each of which consists of two words. The first word is either underlyingly |an| or |am|, and the second word is either |pa| or |ta|. The four underlying utterances are therefore |an#pa|, |an#ta|, |am#pa| and |am#ta|, where “#” stands for the word boundary. In the surface form, the language exhibits nasal place assimilation in a manner reminiscent of Dutch: an underlying coronal nasal tends to assimilate to the place of any following consonant, so that underlying |an#pa| becomes /ampa/ on the surface; meanwhile, an underlying labial nasal tends not to assimilate: |am#ta| becomes /amta/. As in real languages, the tendencies are not true 100% of the time: the assimilation of the coronal nasal is optional, and likewise, the labial nasal does assimilate in a small minority of cases. For our example we suppose that underlying |an#pa| becomes assimilated /ampa/ on the surface 70% of the time, but the “faithful” form /anpa/ 30% of the time, and that underlying |am#ta| becomes faithful /amta/ 95% of the time, and assimilated /anta/ 5% of the time.

This probabilistic state of affairs is a situation that (Stochastic) OT is known to be able to represent (e.g. Boersma and Hayes 2001), because an existing learning algorithm for Stochastic OT (the “GLA”) typically turns a learner into a probability matcher. In comprehension, an auditory form that was intended by the speaker as the surface form /A/ in 70% of the cases and as the surface form /B/ in 30% of the cases, will come to be perceived by the GLA perception learner as /A/ in 70% of the cases and as /B/ in 30% of the cases (Boersma 1997). In production, an underlying form that is produced in the learner’s language environment as the surface form /C/ in 70% of the cases and

as the surface form /D/ in 30% of the cases will come to be produced by the GLA production learner as /C/ in 70% of the cases and as /D/ in 30% of the cases (Boersma and Hayes 2001). Our NN model should be able to replicate this or a similar kind of optimal behaviour.

There are several ways to represent this toy language in a neural network. The most straightforward and OT-like (and probably least realistic) way is to represent each possible underlying utterance (input) with one *node*, and each possible output utterance as one node. This is done in Figure 3, where each of the four possible underlying forms shows up as a single node along the top and each of the four surface candidates shows up as a single node along the bottom.

Figure 3:
A network
that performs
phonological
production



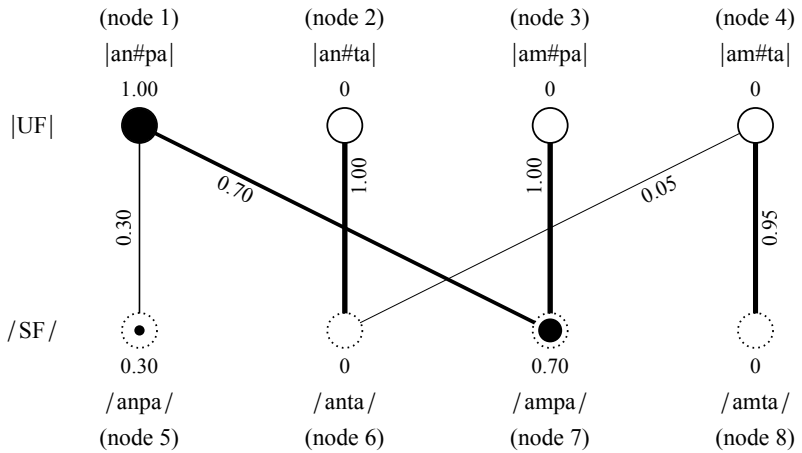
Biologically, a node can be regarded as representing a neuron (or small group of neurons) in the cerebral cortex. Representing an entire linguistic form with a single node (a *local* representation), as we do here, is an unrealistic oversimplification, employed here only for purposes of illustration; more realistic *distributed* representations, where a single phonological category is represented by multiple nodes, appear in Section 5.

In Figure 3, each node is visualized as a dotted circle. Each of the four UF nodes is *connected* to each of the four SF nodes, although only six of the 16 connections are visible. Biologically, a connection corresponds to a synapse (point of contact) between an outgoing branch of one neuron and a receiving branch of another neuron. Such a synapse is unidirectional: it permits an electric signal to flow from one neuron to another. In general, therefore, the total strength of the synapses that carry signals from neuron A to neuron B is not equal to the total strength of the synapses that carry signals

from neuron B to neuron A. Nevertheless, we maintain in this paper the simplification that the strength of the connection from node A to node B equals the strength of the connection from node B to node A, and that it can therefore be called the strength of *the* connection *between* nodes A and B. Such bidirectional connections are known to provide stability in neural network models (Hopfield 1982; O'Reilly 1996), and they guarantee the bidirectionality (Section 1.2) of the BiPhon model, thus providing the desired dispersion effect in Section 6. The present paper can do with, and indeed crucially employs, bidirectional connections; if in future modelling this simplification turns out to be untenable, bidirectionality should then be dispensed with.

In NN modelling, connection strengths are called *weights*. The weight of the connection between the input node |an#pa| and the output node /anpa/ is 0.30, and this is visualized in Figure 3 in two ways: the number 0.30 is written next to this line, and the thickness of the connection line is 0.30. Biologically, the connection weight indeed corresponds to the thickness of the synapse, i.e. the area with which the sending neuron is connected to the receiving neuron. When a biological neuron fires, a neuron with which it has a thick (strong) synapse will be influenced more strongly than a neuron with which it has a thinner (weaker) synapse (our simplified artificial neurons do not actually fire; see Sections 2.2–2.5). In the figure, therefore, thicker lines denote stronger information flows than thinner lines. For instance, the weight of the connection between |an#pa| and /ampa/ is 0.70, which is stronger than that between |an#pa| and /anpa/, because the underlying form |an#pa| should send stronger signals to /ampa/ than to /anpa/ in this toy language. Likewise, the weight of the connection between |an#pa| and /anta/ is zero, because we never want |an#pa| to be realized as /anta/; this zero-weight connection is not visible in the figure (the line has zero width). Also, an underlying “homorganic” |an#ta| is always realized as /anta/, and this is reflected with the number 1.00 next to the relevant connection line in the figure. We will show that with these chosen connection weights, the network in Figure 3 can indeed simulate the data of the toy language if the network has four common additional properties: all-or-none activation of the input nodes (Section 2.2), additive excitation of the output nodes (Section 2.3), a linear

Figure 4:
The production
of underlying
|an#pa|



excitation-to-activity function (Section 2.4), and a linear activity-to-probability function (Section 2.5). We illustrate these concepts with Figure 4, which shows the production of underlying |an#pa|.

2.2

All-or-none activation of the input nodes

To compute how the network handles an incoming underlying form, we apply an *activity pattern* to UF and compute from it the activity pattern that will arise at SF. To see what the network does to an underlying |an#pa|, we activate the |an#pa| node by setting its activity to 1.00. This is shown in two ways in Figure 4: by painting the whole node in black, and by drawing the number 1.00 above the node. At the same time, we set the activities of the three remaining underlying forms to 0, which is indicated in the figure by not painting these three nodes.

Biologically, an activity can be thought of as a firing rate. A node with an activity of 1.00 can be seen as a neuron (or group of neurons) with a maximum firing frequency of, say, 10 spikes per second (Buzsáki and Mizuseki 2014); a node with an activity of 0 can be seen as a neuron (or group of neurons) with a minimum firing frequency (say, 0.1 spikes per second). In this paper we ignore the separate spikes and employ only continuous activities, usually between 0 and 1 (see Section 2.5).

The circles for the UF nodes in Figure 4 look different from those for the SF nodes. In the phonological production process, the UF level is the input, so that the activities of the four UF nodes will be held constant during evaluation. In neural-network terminology, the UF nodes are *clamped* (kept fixed). This is indicated in the figure by the circles for the UF nodes now having solid rather than dotted edges. By contrast, the SF level is the output of the process, so that the activities of the four SF nodes must be free to adapt themselves to the activities of the input nodes; dotted circles in the figure visualize the fact that the output nodes are *unclamped*.

Additive excitation of the output nodes

2.3

When an input node is activated, as node |an#pa| is in Figure 4, the information about its activity will spread toward the nodes with which it is connected: the activity will *excite* every connected node to some extent. For instance, in Figure 4, node |an#pa| has activity 1.00 and the connection between |an#pa| and /ampa/ has weight 0.70. The amount to which |an#pa| will excite /ampa/ is the product of the input activity and the connection weight, i.e. $1.00 \cdot 0.70 = 0.70$. Likewise, node |am#pa| has activity 0 and the connection between |am#pa| and /ampa/ has weight 1.00; |am#pa| will therefore excite /ampa/ by an amount $0 \cdot 1.00 = 0$. Node |an#ta| excites /ampa/ by an amount 0 (the activity of |an#ta|) times 0 (the weight of the connection from |an#ta| to /ampa/), which is $0 \cdot 0 = 0$, and so does |am#ta|.

Biologically, these four excitations can be regarded as “post-synaptic potentials”, rises in the potential (in millivolts) of the membrane of the receiving neuron. These rises tend to be *additive*, i.e. all the small excitations add up to yield the total excitation of the receiving neuron (Lorente de Nó 1938). Artificial neural network models also tend to assume additive excitation. Thus, the total excitation of /ampa/ becomes $0.70 + 0 + 0 + 0 = 0.70$. In a formula, the excitation of the output nodes, i.e. nodes 5 through 8, can be computed as

$$(1) \quad e_j = \sum_{i=1}^4 w_{ij} a_i \quad (\text{for } j = 5..8)$$

where a_i is the activity of UF node i , and w_{ij} is the weight of the connection between UF node i and SF node j .

2.4

Activity of the output nodes

When a node is excited, it becomes active itself. Biologically, this corresponds to the fact that if the membrane potential of a neuron rises, the probability that it will fire increases; in a continuous (and simplified) view of neuronal activity (Perkel and Bullock 1969) this means that if the time-averaged membrane potential rises, the firing frequency of the neuron will rise as well. The simplest assumption about the relation between excitation and activity is that it is *linear*, i.e. the activity rises and falls with the excitation by a constant factor. If this factor is 1, the activity of an SF node in our example becomes equal to its excitation:

$$(2) \quad a_j = e_j \quad (\text{for } j = 5..8)$$

With this *identity activation function*, activating |an#pa| causes an activity of 0.70 in node /ampa/. This number is written over the node in the figure and is also visible as the size of the black disk in that node. Likewise, activating |an#pa| causes an activity of 0.30 in node 5, which is visualized in the figure as the small black disk in that node.

Other excitation-to-activity functions are possible. If one wants to make sure that the activities of the SF nodes do not become negative, (which seems reasonable, given the biological interpretation of the activity as a firing frequency), one could simply clip the activity from below, maintaining linearity of all activities above 0:

$$(3) \quad a_j = \max(0, e_j) \quad (\text{for } j = 5..8)$$

For our toy example, this *rectifying activation function* (Hahnloser *et al.* 2000) works in the same way as the identity activation function of (2), because all excitations are non-negative; in Sections 5 and 6, however, the clipping will be crucial (see Section 5.9.5 for details). Finally, if one wants to take into account that biological firing frequencies have

not just a minimum but also a maximum, one could apply a “top-sigmoid” clipping, which is linear for small excitations and goes to 1 smoothly for large excitations:

$$(4) \quad a_j = \max\left(0, \frac{2}{1 + e^{-2e_j}} - 1\right) \quad (\text{for } j = 5..8)$$

For our toy example, combining the assumption of additive excitation (the contributions from the four underlying forms are added up) and the assumption of the identity excitation-to-activity function (the activity of an output node equals its excitation) causes the activity of an SF node to become the sum of the activities from the input nodes, weighted by the weights of the connections.

*Probabilistic interpretation of the activity
of the output nodes*

2.5

Having computed the activities of the output nodes is not the end of the story. If we want to use neural networks to model linguistic behaviour, we will have to provide a behavioural interpretation of the result in Figure 4. After all, there is no third level of representation that the activities on nodes 5 through 8 could feed into (in this toy example). The only behaviour one can then think of is that the virtual speaker chooses one of the four surface forms to actually produce. The question is: which SF will the virtual speaker choose?

One possible answer is that the speaker chooses the node that has the highest activity, i.e. the node /ampa/. This is an option often found in neural network modelling, especially in competitive learning (Grossberg 1976, 1987; Rumelhart and Zipser 1985). Here, however, this option would throw away the /anpa/ candidate entirely, and such nonstochastic behaviour is not desirable if we want to model the 70–30 variation of our toy language.

Another possible answer is that the speaker somehow produces both /ampa/ and /anpa/. Such a mix might be imaginable at a continuous level of representation such as ArtF, where we can imagine what mixed gestures would look like, but the notion of mixed phonological representations at SF is difficult to envision (but see Section 5.6).

The third possible answer is that the activities denote probabilities: /ampa/, with an activity of 0.70, is chosen with a probability

of 70%, and the only other competing candidate /anpa/, which has an activity of 0.30, is chosen with a probability of 30%. This means that if we ask the network to produce an SF from the input |an#pa| 1000 times, the network will say “/ampa/” approximately 700 times, and “/anpa/” approximately 300 times. In general, then, the probability of output candidate j is its activity, scaled by the sum of all output activities:

$$(5) \quad P_j = \frac{a_j}{\sum_{k=5}^8 a_k} \quad (\text{for } j = 5..8)$$

Thus, since the candidate /ampa/ has an activity of 0.70 and the other candidates have activities of 0.30, 0, and 0, the probability of /ampa/ can be computed under the linear activity-to-probability assumption of (5) as $0.70/(0.30 + 0.70 + 0 + 0) = 70\%$. Equation (5) is known in psychology as *Luce's choice axiom* (Luce 1959: 23), and it can apply to any type of non-negative numbers a_j that represent strengths (or weights, or activations, or saliences) of the candidates j .

Such an interpretation of an activity as a relative probability has a biological correlate. If activity can be regarded as firing frequency, and /ampa/'s activity is 0.70 while /anpa/'s activity is 0.30, then node /ampa/ fires 2.333 times as often as node /anpa/ in any given period of time. This means that if, from a certain moment in time on, one waits until either node /ampa/ or node /anpa/ fires, the odds will be 7 to 3 that node /ampa/ fires earlier than node /anpa/. In other words, there will be a probability of 70% that node /ampa/ fires first, and a probability of 30% that node /anpa/ fires first. If the first node to fire determines the speaker's behaviour, the relative activities have apparently determined the relative probabilities of the behaviour.

Different interpretations of the relation between activity and probability are nevertheless possible. In the *Boltzmann machine* (Ackley *et al.* 1985), the probabilities are

$$(6) \quad P_j = \frac{e^{a_j/T}}{\sum_{k=5}^8 e^{a_k/T}} \quad (\text{for } j = 5..8)$$

where T is called the *temperature*. Equation (6), known in modern machine learning as *softmax*, is due to Boltzmann (1868), is a special

case of Luce's choice axiom, and can apply to any type of numbers a_j (even negative ones) that represent strengths of the candidates j . The simpler linear relation of (5), however, suffices for the present paper, because we work solely with non-negative activities (see especially Section 5.6).

Bidirectionality violated?

2.6

The network of Figure 3 works correctly in the production direction, i.e. with UF as the input and SF as the output. In the spirit of the BiPhon model we would like it to work equally well in the comprehension direction, i.e. with SF as the input and UF as the output. To model the recognition of an incoming /ampa/ as an underlying sequence of words, we can start by clamping the four SF nodes by keeping the /ampa/ node at a constant activity of 1.00 and the other three nodes constantly at zero. According to Figure 3 and the procedure of (1) and (2), the underlying form |an#pa| will get an activity of 0.70 and the underlying form |am#pa| will get an activity of 1.00. Apparently, the network prefers |am#pa| over |an#pa| when it listens.

This situation is fine if the underlying forms |an#pa| and |am#pa| occur equally often in the language environment: the network's preference then mimics the likelihood with which each of the two underlying forms was intended, given the surface form /ampa/. If, however, coronals occur in word-final position three times more often than labials do (which is approximately true for Dutch and English), the underlying form |an#pa| is three times more likely a priori than |am#pa| is. According to Bayes (Laplace 1812), this should shift the preference of a listener toward |an#pa|, but in the network of Figure 3 this is not taken into account. In fact, the weights are conditional probabilities on UF only, not on SF.

This asymmetry between comprehension and production is a general property of symmetric connections. It cannot be completely solved, but it can be made equally (un)problematic for both directions of processing, as we do in Section 4.

Section 2 has shown that an artificial neural network can replicate the decision mechanism of (Stochastic) OT or (Noisy) HG; in other words,

the network mimics the decision mechanism of a probabilistic grammar. It is unsatisfying, though, that each full utterance is represented as a single node. In a more realistic network, the representation of each phonological element will be *distributed* over multiple nodes. Such a network is discussed in Section 5. Understanding such a network, however, requires understanding how the activities of equation (1) come about in processing (Section 3), and how the weights in Figure 3 come about in learning (Section 4).

3

ACTIVITY SPREADING

In the example of Section 2, the initially unknown activities of the unclamped (output) nodes could be computed directly by equations (1) and (2) from the given activities of the clamped (input) nodes. Such a direct computation is possible for simple two-level mappings as in that example, but with larger networks, in which information flows bottom-up, top-down and within levels simultaneously, a direct computation is no longer possible, because the activities of some unclamped nodes come to depend on the activities of other unclamped nodes that themselves are not known from the start.

The general solution is to compute the activity in the unclamped nodes iteratively, i.e. in small steps, from the given activities of the clamped nodes, and let the network gradually approach its equilibrium, i.e. a final state in which its activities stop changing. Such gradual activity spreading bears similarities with how activity spreads through biological neural networks, and proceeds as follows. After applying some known activities to the clamped nodes, we let the excitations (and activities) of the unclamped nodes start at zero, and we then update these excitations in small steps several hundreds of times. In the example of Section 2, the excitation in the output nodes 5 through 8 starts at zero, and is incremented at every time step (say, every millisecond) by an amount Δe_j given by

$$(7) \quad \Delta e_j = 0.01 \cdot \left(\sum_{i=1}^4 w_{ij} a_i - e_j \right) \quad (\text{for } j = 5..8)$$

where the factor of 0.01 is the *spreading rate*.

For our specific toy example, it is easy to show that the general equation (7) indeed produces the end result of equation (1) after some time. Consider the situation for the output node /ampa/ at time 0. We already know that $\sum_{i=1}^4 w_{i7} a_i = 0.70$, so at time zero, when $e_7 = 0$, Δe_7 will be $0.01 \cdot (0.70 - 0) = 0.007$. Therefore, e_7 becomes 0 (its previous value) plus 0.007 (the increment), which makes 0.007. At the next time step, $\sum_{i=1}^4 w_{i7} a_i$ is still 0.70, but e_7 is 0.007, so that the increment Δe_7 is $0.01 \cdot (0.70 - 0.007) = 0.00693$, just 1% smaller than the previous increment. As a result, the new value of e_7 becomes $0.007 + 0.00693 = 0.01393$. Figure 5 shows what happens if this procedure is repeated 500 times (i.e. for, say, half a second): while the increment decreases exponentially by a factor of 0.99 at each time step, the excitation (and therefore the activity) of output node 7 grows asymptotically toward 0.70.

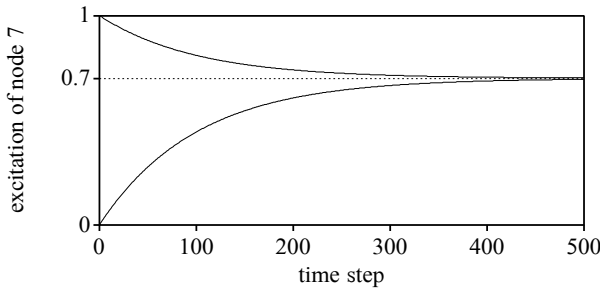


Figure 5:
The time path of the excitation (and activity) of node /ampa/.
Bottom curve: starting from 0.
Top curve: starting from 1.00

One could have predicted the end state of our toy example directly from (7), by realizing that in the equilibrium situation Δe_7 goes to zero. Equation (7) tells us that in that case $\sum_{i=1}^4 w_{i7} a_i - e_7$ must go to zero as well. This means that e_7 goes to $\sum_{i=1}^4 w_{i7} a_i$, i.e. to 0.70, so the activity, by (2), also goes to 0.70, which is the activity in Figure 4. This also shows that the starting value of the excitation does not matter: the excitation will go to 0.70 no matter where it started; as an illustration, Figure 5 also shows how the excitation develops if it starts at 1.00. This kind of reasoning from zero increments is a general trick to predict what the final situation will look like, given a formula for increments.

The evolution of the activities toward a constant final state, as in Figure 5, is general for symmetric networks (Hopfield 1982; Ackley *et al.* 1985). After enough time, each node j reaches a stable equilibrium state where its excitation stops changing, i.e. where Δe_j ap-

proaches zero. As a result, the whole network reaches equilibrium, i.e. the excitations of all its nodes stop changing. Symmetric networks, where w_{ij} equals w_{ji} , are guaranteed to move toward such a stable final state.

The general formula for the activity spreading toward an unclamped node j from its (clamped or unclamped) neighbours i is

$$(8) \quad \Delta e_j = \eta_\alpha \left(\sum_{\text{connected nodes } i} (w_{ij} - \text{shunting } e_j) a_i - \text{excitationLeak } e_j \right)$$

Here, η_α is the spreading rate, which in our simulations is kept constant at a value of 0.01. The *excitation leak* factor was set to 1 in (7), but could be set to higher values if we want to reduce the ultimate activity values. The *shunting* factor (Grossberg 1976) is included here only for completeness; it is set to 0 in all simulations in this paper.

4

A LEARNING RULE
FOR BIDIRECTIONAL LINGUISTICS:
INOUTSTAR

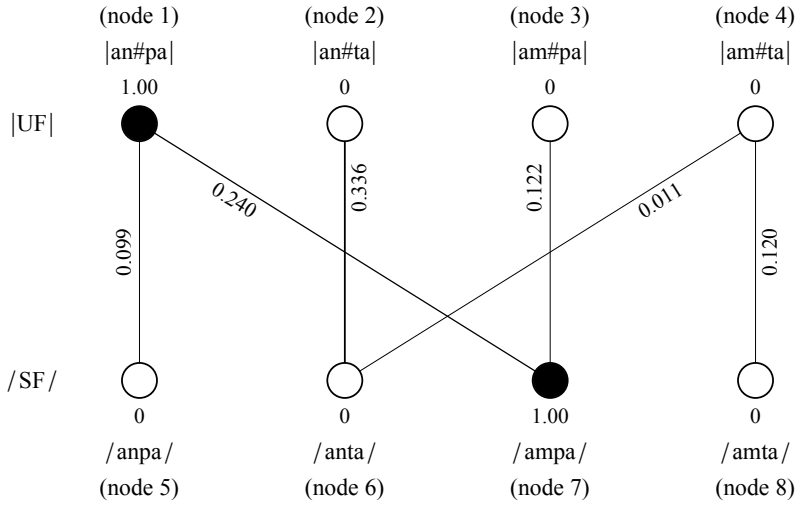
The representations and processes discussed in Sections 2–3 are transient things: they come and go every few seconds as the listener receives more speech or the speaker produces more speech. The connection weights contain more persistent information, namely the aspects of knowledge seen in Figure 1. These weights can *learn* from experience: they change only slowly over the months and years as the child is acquiring her language. In this section we identify a learning rule for our stochastic bidirectional artificial networks: we show that out of a family of Hebbian-like learning rules the only rule that meets the requirements of stochasticity and symmetric bidirectionality is what we call *inoutstar*. Learning rules that are more familiar from the literature are either not stochastic at all (clipped learning) or do not match the conditional probabilities in the environment (leaky learning, *instar*, *outstar*).

Suppose we have the toy language of Section 2.1, with the coronal bias of Section 2.6: the UF $|an\#pa|$ occurs 37.5% of the time, of which the SF will be $/ampa/$ 70% of the time and $/anpa/$ 30% of the time; the UF $|an\#ta|$ occurs 37.5% of the time, yielding the SF $/anta/$ 100% of the time; the UF $|am\#pa|$ occurs 12.5% of the time, yielding the SF $/ampa/$ 100% of the time; and the UF $|am\#ta|$ occurs 12.5% of the time, yielding the SF $/amta/$ 95% of the time and $/anta/$ 5% of the time. The task for the virtual learner is to start with the network of Figure 3, but with all weights set to 0 (or a small random number), and then to adapt these weights under supervision from the language data.

For this purpose, we feed the network with a large number, say 100,000, of UF-SF pairs randomly drawn from the language environment. Thus we feed the learner with the pair $|an\#ta|$ - $/anta/$ in 37.5% of these 100,000 cases, and with $|an\#pa|$ - $/ampa/$ 26.25% of the time (70% of 37.5% is 26.25%); also with $|am\#pa|$ - $/ampa/$ 12.5% of the time, with $|am\#ta|$ - $/amta/$ 11.875% (95% of 12.5%) of the time, with $|an\#pa|$ - $/anpa/$ 11.25% (30% of 37.5%) of the time, and with $|am\#ta|$ - $/anta/$ the remaining 0.625% (5% of 12.5%) of the time. In Figure 3 we see that the five most common pairs are represented in the working network with the five strongest weights (though not in exactly the same order). The intuition, then, is that the learning algorithm should make those weights strong that connect nodes that are associated with each other in the data.

Now, what does it mean to “feed” UF-SF data to the network? It means that if at a certain point during learning we want to feed the network with, say, the pair $|an\#pa|$ - $/ampa/$, we set the activity of nodes 1 ($|an\#pa|$) and 7 ($/ampa/$) to 1.00 and the activities of the other six nodes to 0. This is the situation in Figure 6. We then let activity settle down by having the activity spread 500 times (this does nothing in this case, because all eight nodes are clamped). After this, we change all 16 connection weights by a small amount. This whole procedure of selecting an UF-SF pair, setting the activities, vacuously spreading the activities, and changing the weights, is repeated 100,000 times, as said. In Section 4.2 through Section 4.7 we discuss six ways to do the weight changes and compare their suitability for implementing bidirectional probability matching.

Figure 6:
Supervised
two-level
learning: all
nodes are
clamped, and
only one node
is on in UF
as well as SF



4.2

Unbounded linear learning

The simplest way to react to the shared activity of nodes 1 and 7 is to raise the weight of their connection ($w_{1,7}$) by a small amount, say 0.01, and not change the weight of any of the other 15 connections. This can be achieved by the following “Hebbian learning” formula:

$$(9) \quad \Delta w_{ij} = \eta_w a_i a_j \quad (\text{for } i = 1..4, j = 5..8)$$

where η_w is the *learning rate*, which is 0.01 here. This works correctly, because for $i = 1$ and $j = 7$, $a_i a_j$ equals 1 (because both a_i and a_j are 1.00), whereas for all 15 remaining i - j combinations either a_i is 0, or a_j is 0, or both a_i and a_j are 0. So $w_{1,7}$ is indeed the only weight that changes. The rule is named after Hebb (1949), who proposed that a synaptic strength increases when two neurons fire together, though he did not actually propose formula (9).

There is a problem with learning rule (9). If it goes on for 1000 times, $w_{1,7}$ will change approximately 250 to 275 times, because the network will be fed the |an#pa|-/ampa/ pair 26.25% of the time. A simulation with 2000 randomly drawn pairs is shown in Figure 7. We see that w_{ij} increases linearly with time, and goes on to do so without bounds. It has been known from the beginning of neural network modelling that the “pure Hebbian learning” of (9) exhibits this pathological behaviour (Rochester *et al.* 1956). Various devices have been

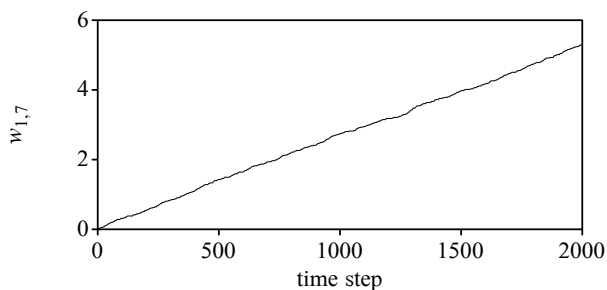


Figure 7:
The development of a weight
in pure Hebbian learning:
linear and without bounds

proposed in the literature to keep w_{ij} within bounds; in Sections 4.3–4.7 we discuss their suitability for our bidirectional toy case.

Clipped linear learning

4.3

A brute-force method to keep w_{ij} within bounds is to clip w_{ij} from below by a value w_{min} (e.g. 0) and from above by a value w_{max} (e.g. 1). This method has the tendency of ultimately pushing most weights toward either w_{min} or w_{max} . If the input is such that a single node i is on (and all other input nodes are off), and there are 10 output candidates (= nodes), then e.g. 3 output candidates will be maximally activated (namely those for which w_{ij} equals 1) and 7 candidates will be off (namely those for which w_{ij} equals 0). This means that under the first or third scenario from Section 2.5, three output candidates have a probability of 1/3 to win, and the remaining seven output candidates have a probability of 0 to win (the second scenario from Section 2.5 is not interpretable). This situation is not good for stochastic decision-making, where we want probabilities to move gradually from 0 to 1 or the reverse. In our simulations in Sections 5 and 6 we therefore work with activities that are not clipped from above (although they are clipped from below at 0, so we get some zero probabilities).

Leaky learning

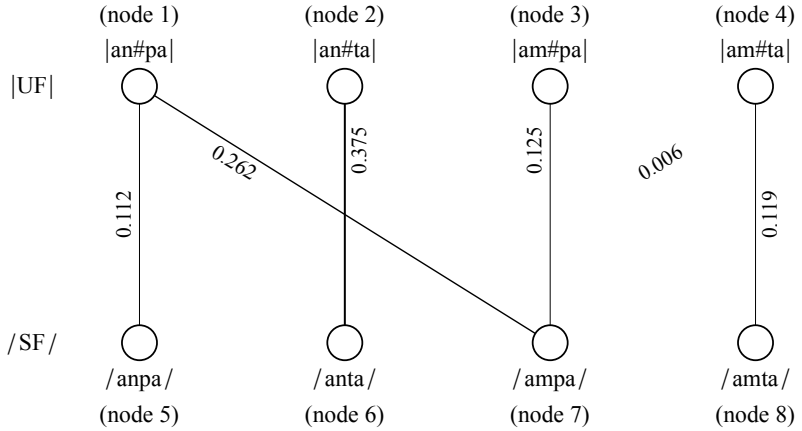
4.4

A more gradual way to keep w_{ij} within bounds is to introduce leak:

$$(10) \quad \Delta w_{ij} = \eta_w (a_i a_j - w_{ij}) \quad (\text{for } i = 1..4, \quad j = 5..8)$$

The weights now start to rise exactly as in Figure 6, but after some time they start to rise more slowly, growing exponentially toward an equilibrium in very much the same way as in Figure 5, albeit with never-ending fluctuations because of the stochasticity of the input. After many pieces of data (UF–SF pairs), the weights come to hover around those in Figure 8.

Figure 8:
The average
end state
of leaky learning
in the language
environment
of Section 4.1



In this final (asymptotic) situation after learning, each weight has become exactly the probability of the relevant UF–SF pair as mentioned in Section 4.1; the sum of all the weights in Figure 8 is 1. We could have predicted this result theoretically by realizing that in the equilibrium situation the expected weight change $\langle \Delta w_{ij} \rangle$ must be 0 for each connection; in other words: for each i and j the average of Δw_{ij} over all possible UF–SF pairs that could come in next, weighted by the probabilities of these pairs according to Section 4.1, must be zero. Equation (10) then tells us that the expectation value $\langle a_i a_j - w_{ij} \rangle$ will then move toward zero, so that the weight w_{ij} will ultimately go toward the correlation between a_i and a_j :

$$(11) \quad w_{ij} \rightarrow \langle a_i a_j \rangle$$

Thus, the asymptotic behaviour of w_{ij} can be predicted if we know the statistics of the activity pattern. For instance, 26.25% of the time node 1 is on ($a_1 = 1$) and node 5 is off ($a_5 = 0$), 11.25% of the time nodes 1 and 5 are both on ($a_1 = a_5 = 1$), 62.5 percent of the time nodes 1 and 5 are both off ($a_1 = a_5 = 0$), and 0%

of the time node 1 is off ($a_1 = 0$) and node 5 is on ($a_5 = 1$); the weight of the connection between nodes 1 and 5 will therefore go to $\langle a_i a_j \rangle = 0.2625 \cdot 1 \cdot 0 + 0.1125 \cdot 1 \cdot 1 + 0.625 \cdot 0 \cdot 0 + 0 \cdot 0 \cdot 1 = 0.1125$. Since three of the four terms are zero if node 1 and node 5 are not both on, this expectation value necessarily equals the probability that both node 1 and node 5 are on simultaneously. This is a general result if all activities can take on only the values 0 and 1:

$$(12) \quad w_{ij} \rightarrow P(a_i = 1 \wedge a_j = 1)$$

Such pure correlation learning looks nicely simple, but has a disadvantage. Relatively rare inputs will lead to weak connections: |am#pa| has a three times weaker connection in Figure 8 than the three times more common input |an#ta|. This disregards the perfect degree to which the SF /ampa/ can be predicted from |am#pa|. The frequency difference between |am#pa| and |an#ta| thus leads to a large difference in the activities at SF, which means that further on in processing the rare UF counts *much* less heavily than the more frequent UF. A learning rule that focuses on reliability rather than frequency alone may fare better in this respect. Another problem is that the small output activities for rare inputs (such as 0.125 for /ampa/) do not reflect the full activity that occurred during learning (which was 1 for /ampa/).

Outstar learning

4.5

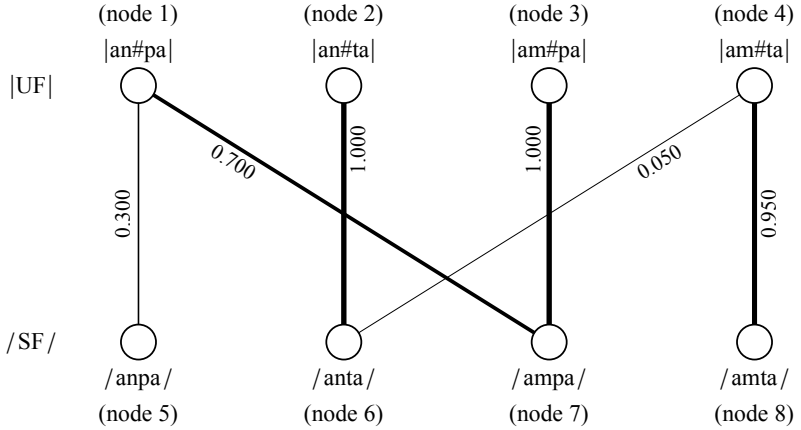
The cause of the problems with leaky learning is that the algorithm leaks *too much*: connections get weaker even if their two nodes are both inactive. One way to remedy the problem is to use the *outstar* learning rule (Grossberg 1969):

$$(13) \quad \Delta w_{ij} = \eta_w (a_i a_j - a_i w_{ij}) \quad (\text{for } i = 1..4, \quad j = 5..8)$$

This learning rule does nothing with a connection if its input node is off ($a_i = 0$). A property that none of the learning algorithms discussed above share is that for outstar learning we have to assign a direction to the process, for instance to define UF as the input level and SF as the output level; so we choose the production view here, as in Section 2.

For the example in Figure 6, outstar learning will strengthen the connection between nodes 1 and 7, weaken the connections 1–5, 1–6 and 1–8, and leave the remaining 12 connections alone. After many learning steps with UF–SF pairs from our toy language, the weights come to hover around the equilibrium values in Figure 9.

Figure 9:
The average
end state of
outstar learning
in the language
environment
of Section 4.1



In the end, the weights turn out to have become the conditional probabilities of SF given UF (as in Figure 3), so outstar learning exhibits the probability-matching behaviour that we wanted; the sum of the weights going out from each UF node is 1. This could have been predicted theoretically, by realizing that in the equilibrium situation $0 = \langle a_i a_j - a_i w_{ij} \rangle = \langle a_i a_j \rangle - \langle a_i \rangle w_{ij}$, so if learning converges, it must move the weights asymptotically toward

$$(14) \quad w_{ij} \rightarrow \frac{\langle a_i a_j \rangle}{\langle a_i \rangle}$$

For cases where all activities during learning can only be 0 and 1, equation (14) reduces to the conditional probability that output node j is on given that input node i is on:

$$(15) \quad w_{ij} \rightarrow \frac{P(a_i = 1 \wedge a_j = 1)}{P(a_i = 1)} = P(a_j = 1 | a_i = 1)$$

Outstar learning has several advantages. As the weights in outstar learning come to reflect conditional probabilities, the weights

naturally stay within the limits of 0 and 1. Furthermore, outstar learning fares better than correlation learning with respect to reliability, mimicking the GLA for Stochastic OT: the connections from |am#pa| and |an#ta| are now equally strong, reflecting the fact that their SF outputs can be equally reliably predicted from the UF. Also, the activities at SF will now be 1 for these two inputs, just as during learning.

Outstar learning also has a disadvantage over the leaky learning model in (10): it loses all dependency of SF activity on the frequency of the input. A way to have both reliability and frequency influences could be to somehow combine (10) with (13). There is a problem with both (10) and (13), though: some nodes at SF, such as /anpa/, are very *specific* for certain UF forms, and this is not rewarded with a strong connection; in other words, (15) does not take into account whether or not output node j is on if input node i is off. One can look at this in terms of the reliability of the reverse process, i.e. the mapping from SF to UF in word recognition: the connection in Figure 9 from the SF /anpa/ to the UF |an#pa| is only 0.300, although the UF can be predicted with 100% reliability from the SF. We tackle this problem in Section 4.6.

Outstar learning is close to the *delta rule* of supervised learning algorithms (Widrow and Hoff 1960), where the weight update is proportionate to the *error* that the network would make when allowed to run freely (i.e. with UF clamped but SF unclamped); the error is the difference between the desired activity at SF (i.e. the number of 0 or 1, as used as a_j in the SF clamping above) and the activity that the SF node j would get when only the input UF nodes are clamped, which is $\sum a_i w_{ij}$ in the examples of Section 2:

$$(16) \quad \Delta w_{ij} = \eta_w \left(a_i a_j - a_i \sum_{k=1}^4 a_k w_{kj} \right) \quad (\text{for } i = 1..4, \quad j = 5..8)$$

This, together with the property of probabilities conditional to the input, makes this algorithm a good candidate for replicating results previously found with Stochastic OT. This algorithm is therefore expected to be of use when in Section 6 we model auditory dispersion, a phenomenon previously modelled successfully with Stochastic OT (Boersma and Hamann 2008).

4.6

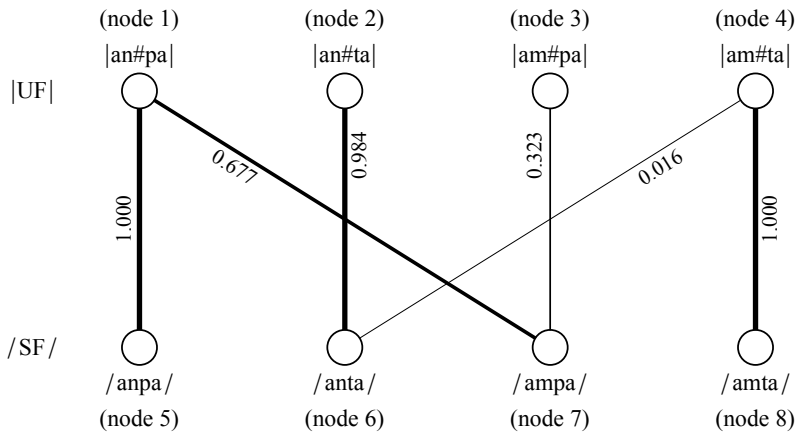
Instar learning

To take the specificity of SF (Section 4.5) into account, we can apply the instar learning rule (Grossberg 1969, 1976; Rumelhart and Zipser 1985),³ which is the outstar learning rule in the opposite direction of processing:

$$(17) \quad \Delta w_{ij} = \eta_w (a_i a_j - a_j w_{ij}) \quad (\text{for } i = 1..4, \quad j = 5..8)$$

This learning rule does nothing with a connection if its output node is off ($a_j = 0$). As with outstar, we explicitly have to define what the input and what the output level are (again, we take the production view, with UF as input and SF as output). For the example in Figure 6, instar learning will strengthen the connection between nodes 1 and 7, weaken the connections 2–7, 3–7 and 4–7, and leave the 12 remaining connections alone. For our toy language, the weights come to hover around the values in Figure 10.

Figure 10:
The average end state
of instar learning
in the language
environment
of Section 4.1



Asymptotically, the weights turn out to become the conditional probabilities of UF given SF; the sum of the weights coming in at each SF node is 1. In the theoretical equilibrium situation,

$$(18) \quad w_{ij} \rightarrow \frac{\langle a_i a_j \rangle}{\langle a_j \rangle}$$

³Oja (1982) has a formulation in which the second a_j is squared.

For cases where all activities during learning can only be 0 and 1, equation (18) reduces to the conditional probability that input node i is on given that output node j is on:

$$(19) \quad w_{ij} \rightarrow \frac{P(a_i = 1 \wedge a_j = 1)}{P(a_j = 1)} = P(a_i = 1 | a_j = 1)$$

The two problems with rare inputs are not addressed, but the specificity problem is solved: the connection from the SF /anpa/ to its only possible UF |an#pa| has a weight of 1. The effect of the different frequencies of the different underlying forms has also returned, with the connection from /ampa/ to |an#pa| now being stronger than the connection from /ampa/ to |am#pa|, as in leaky learning but not as in outstar learning. The drawback is that the infrequent UF |am#pa| will now produce a much smaller activity pattern in SF (a total of 0.323) than the more frequent UF |an#pa| (a total of 1.677). We address this problem in Section 4.7.

Instar learning is known from work on competitive learning (Grossberg 1976, 1987; Rumelhart and Zipser 1985). This algorithm is therefore expected to be of use when in Section 5 we model phonological category creation, a phenomenon that has been partially modelled before with competitive learning (Guenther and Gjaja 1996).

Inoutstar learning

4.7

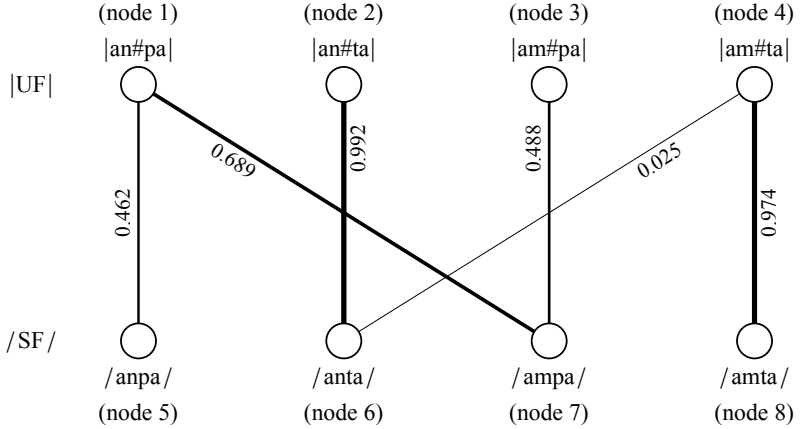
To model category creation we seem to need unsupervised instar learning (Section 4.6), and to model auditory dispersion we seem to need supervised outstar learning (Section 4.5). However, both processes occur in the AudF–SF interface, so the same network will have to model them both. Our goal, therefore, is to model both category creation and auditory dispersion with a single learning algorithm, perhaps a compromise between instar and outstar. We call this the “inoutstar” learning rule:

$$(20) \quad \Delta w_{ij} = \eta_w \left(a_i a_j - \frac{a_i + a_j}{2} w_{ij} \right) \quad (\text{for } i = 1..4, \quad j = 5..8)$$

This learning rule does nothing with a connection if both of its nodes are off. For the example in Figure 6, inoutstar learning will strengthen

the connection between nodes 1 and 7, weaken the connections where one node is on and the other off (1–5, 1–6, 1–8, 2–7, 3–7 and 4–7), and leave the remaining nine connections alone. For our toy language, the weights come to hover around the values in Figure 11.

Figure 11:
The average
end state
of inoutstar
learning
in the language
environment
of Section 4.1



Asymptotically, each weight turns out to become the harmonic mean of the weights of Figures 9 and 10. In the theoretical equilibrium situation,

$$(21) \quad w_{ij} \rightarrow \frac{2 \langle a_i a_j \rangle}{\langle a_i + a_j \rangle}$$

For cases where all activities during learning can only be 0 and 1, equation (21) reduces to the harmonic mean of the two conditional probabilities:

$$(22) \quad w_{ij} \rightarrow \frac{2 P(a_i = 1 \wedge a_j = 1)}{P(a_i = 1) + P(a_j = 1)} \\ = \frac{2 P(a_i = 1 | a_j = 1) P(a_j = 1 | a_i = 1)}{P(a_i = 1 | a_j = 1) + P(a_j = 1 | a_i = 1)}$$

Inoutstar learning was used before by McMurray *et al.* (2009) to simulate word–object mappings. It combines the desirable properties of instar and outstar: it tackles all problems mentioned to some extent, though none of them perfectly: it does some probability matching, it

has some specificity, and it is even a bit frequency-dependent in both directions (because instar and outstar are both frequency-dependent in one direction). It has the additional advantage over both instar and outstar learning that it is symmetric in input and output: the formula stays the same if i and j are swapped, i.e. the inoutstar learning rule does not care about the direction of processing. This will even be true if there are separate weights in the beginning, i.e. if w_{ij} is not equal to w_{ji} at the beginning of learning: equation (22) shows that inoutstar learning causes the weights to become symmetric. Inoutstar can therefore be expected to implement quite well the bidirectionality of models such as the one in Figure 2.

Conclusion

4.8

A general formula for the change in the weight between input node i with activity a_i and output node j with activity a_j could be

$$(23) \quad \Delta w_{ij} = \eta_w (a_i a_j - \text{instar } a_j w_{ij} - \text{outstar } a_i w_{ij} - \text{weightLeak } w_{ij})$$

We investigated pure Hebbian learning ($\text{instar} = 0$, $\text{outstar} = 0$, $\text{weightLeak} = 0$), leaky learning ($\text{instar} = 0$, $\text{outstar} = 0$, $\text{weightLeak} = 1$), instar learning ($\text{instar} = 1$, $\text{outstar} = 0$, $\text{weightLeak} = 0$), outstar learning ($\text{instar} = 0$, $\text{outstar} = 1$, $\text{weightLeak} = 0$), and inoutstar learning ($\text{instar} = 0.5$, $\text{outstar} = 0.5$, $\text{weightLeak} = 0$). Of these, inoutstar learning combines to some extent some of the good properties of the other learning algorithms, such as symmetry (insensitivity to the direction of processing), probability matching in both directions of processing, specificity in both directions of processing, and sensitivity to the frequency of the input in both directions. In Sections 5 and 6 we investigate the suitability of this algorithm for two hitherto separately modelled phenomena, namely category creation and auditory dispersion.

The equations from Sections 2 through 4 that we use for the simulations in Sections 5 and 6 are only the simplest ones that meet the requirements above, namely (7), (3) and (20). We summarize them here in their generalized forms that work not only for the toy example of Sections 2 through 4 but for any network with a combination of clamped and unclamped nodes, including the networks of Sections 5

and 6. As for activation spreading, every clamped node j has a constant activity a_j , and every unclamped node j starts with excitation $e_j = 0$ and activity $a_j = 0$ after which its excitation changes 100 or 500 times according to

$$(24) \quad \Delta e_j = 0.01 \cdot \left(\sum_i w_{ij} a_i - e_j \right)$$

where the index i runs over all nodes connected to j . After each of these time steps, the activity of every unclamped node j is immediately determined from its excitation by the simple rectifying excitation function, which prevents negative activities:

$$(25) \quad a_j = \max(0, e_j)$$

After cycling through all the time steps, the activities of all unclamped nodes should almost have settled, and the weight of the connection between any pair of nodes i and j is updated by the (symmetric and bidirectional) inoutstar learning rule:

$$(26) \quad \Delta w_{ij} = \eta_w \left(a_i a_j - \frac{a_i + a_j}{2} w_{ij} \right)$$

5 PHONOLOGICAL CATEGORY CREATION

In this section we present a neural network that can model the emergence of simple phonological categories in the language-acquiring child. In terms of Figures 1 and 2, phonological categories, such as feature values, are present in the adult phonological Surface Form (SF). In the comprehension direction of Figure 2, the cue knowledge at the adult phonology–phonetics interface classifies the thousands of different sounds that can occur in the Auditory Form (AudF) into a small number of discrete categories at SF. In terms of neural networks, a “category” can only be defined as a stable, or “attractive”, activity pattern. That is, an adult network at the phonetics–phonology interface should “filter” the thousands of possible activity patterns at AudF into only a small number of possible activity patterns at SF.

In existing models of phonology category learning (Guenther and Gjaja 1996; Boersma *et al.* 2003) the adult state of the grammar or network comes about by training the grammar or network with a large number of auditory values at AudF, without telling the grammar or network what the intended category was. Such “unsupervised” learning is also employed here. In Section 5.3 we describe how this learning proceeds, after having described the network structure in Section 5.1 and the AudF input in Section 5.2. The resulting adult network is presented in Section 5.4 and understood in Section 5.5, after which we investigate its behaviour in perception (Section 5.6) and production (Section 5.7) and compare this behaviour to the existing literature (Section 5.8). In-depth investigations of the underlying mechanism (Section 5.9) and its response to variable environments (Section 5.10) follow. Finally, we compare the network’s performance and assumptions to the existing literature (Section 5.11).

A network for category emergence

5.1

Figure 12 shows the structure of the network that should learn the task of categorizing auditory input. The network contains only two levels of representation: the phonetic Auditory Form, which is the input for the listening learner, and the phonological Surface Form, which is the listener’s perceptual output. As we model only the phonological part of comprehension, we do not include the higher levels of Figures 1 and 2 (Underlying Form and Morphemes). Moreover, as most of Section 5 models only the comprehension direction and not the production direction (the exception being Section 5.7), we do not include the

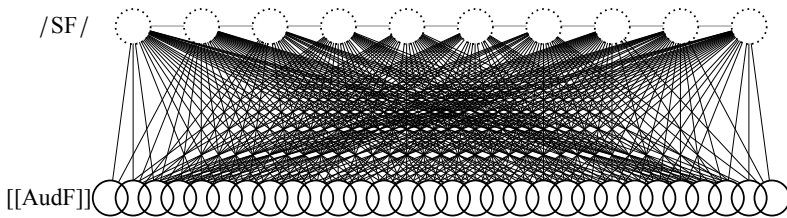


Figure 12: The initial state of a network for category creation, with continuous sound coming in at clamped AudF and discrete behaviour emerging at unclamped SF

Articulatory Form (see Section 6 for that), although including such a level would not change any of the perception or category creation simulations, as we explain in Section 5.7.

The Auditory Form represents an auditory continuum, such as the frequency spectrum along the basilar membrane. While the basilar membrane has 3,500 inner hair cells, each of which is connected to a fiber in the auditory nerve, we represent the spectrum here with only 30 nodes for reasons of visualizability (and computation time). Figure 12 arranges the nodes in a natural order, with the leftmost node (node 1) representing the lowest audible frequency of the continuum, and the rightmost node (node 30) representing the highest audible frequency.

As a simplification we allow the incoming sound to activate only one small region of AudF (as e.g. in Figure 14); this means that AudF can only represent a unitary spectral continuum, and for this we choose the spectral centre of gravity (CoG).

The Surface Form in Figure 12 will come to represent phonological “sibilant place”, because that is the feature that has CoG as its main auditory correlate. Every category that the SF in Figure 12 has to be able to represent, is therefore a value of the feature sibilant place. Languages seem to have between one and four primary sibilant place values, so our SF should be able to represent between one and four categories. Even if we restrict the activity patterns at SF in such a way that each node is either “on” (1) or “off” (0), the SF in Figure 12 can represent as many as $2^{10} = 1024$ different categories; and if “on” nodes cannot be shared between categories, the SF in Figure 12 can represent 10 different categories. In either case, our 10 nodes should be more than enough to represent any number of feature values between one and four in a distributed way.

As can be seen in Figure 12, AudF and SF are fully connected to each other: there are 300 connections between them, one for each pair of AudF node and SF node. Initially, these weights are small and random: uniformly distributed between 0 and 0.1, as shown as black lines in the figure. This randomness is meant to ensure that in its initial state the network is poor at classifying incoming sounds into stable categories: in perception (with a clamped AudF and an unclamped SF, as in Figure 12), any local activity peak in AudF will just lead to a small and random pattern at SF (as can be seen for example in Figure 14).

As illustrated in Section 5.4, this situation will change when the network learns from incoming sounds at AudF: the weights will become larger and less random. As Section 5.6 shows, the result is the desired emergence of categorical behaviour in the network.

Finally, Figure 12 shows 45 connections within SF: one for each pair of SF nodes (i.e. not just between nodes that happen to look “adjacent” in the visually one-dimensional set-up of Figure 12). These connections have negative weights of -0.1 (shown in light gray) in order to make sure that the SF nodes inhibit each other’s activities. As a result, learning causes the SF nodes to become connected to different AudF patterns, which is illustrated in Section 5.4 and explained in 5.5. This ensures that different categories from the network’s language environment lead to different categorical patterns in the learner’s own SF. This mutual inhibition is a mechanism we borrow from competitive learning models (Grossberg 1976, 1987; Rumelhart and Zipser 1985). The negative weights do not change during learning.

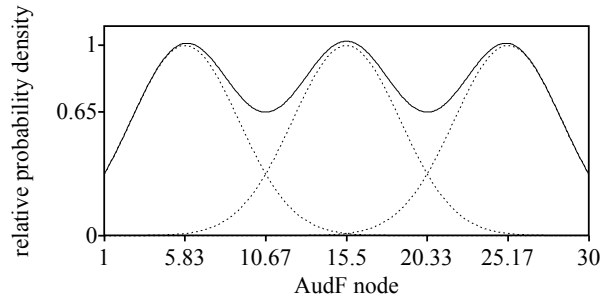
An input distribution for sibilant place

5.2

As said, the network will be trained with the auditory distribution alone, i.e. it will have to learn from incoming CoG values from a language environment, without supervision. Thus, the virtual learner repeatedly hears an incoming sound but is never told to what category it belongs and is never told any of the associated higher levels of representation, such as meaning. Neither is the learner told how many categories the language has.

For the coming sections of this paper, we investigate a very simple language environment that consists of three sibilant fricatives, namely /ʃ/, /ç/ and /s/, as in Polish or Mandarin. The spectral centre of gravity of each sibilant is distributed according to a Gaussian distribution, as in the three dotted curves in Figure 13. The distance between the peaks is one third of the range of the continuum, i.e. 9.667 nodes, and the standard deviation of each peak is one third of that (i.e. 3.222 nodes). The three sibilants are equally frequent in the language environment, so that the total distribution of CoG values is the solid curve in Figure 13.

Figure 13:
A CoG distribution in a language
with three sibilant places



The beginning learner does not yet know that there are three curves; she only hears input tokens one by one without category labels, and the summed distribution of these input tokens gradually and incrementally grows toward the total CoG distribution. The valleys in this curve are rather shallow, namely approximately 64% of the average height of the three peaks. In the end, it is on the basis of input drawn from the summed distribution, with its shallow valleys, that the learner will have to figure out that there are three categories.

5.3

Unsupervised learning from the distribution

A full description of a language learning procedure involves describing how each input is applied to the learner, how the learner processes this input, and how the learner then changes her grammar. In our case, the input to the network is formed by the learner's language environment repeatedly producing a single CoG value randomly drawn from the summed distribution (equivalently, the language environment randomly selects one of the three sibilants, then randomly draws a CoG value from that sibilant's Gaussian distribution; the important restriction is that the learner is not told which sibilant was selected). The learner receives this CoG value as an activity at AudF, then processes it by spreading this activity to SF, and finally updates the connection weights between AudF and SF on the basis of the activities at AudF and SF. We will show here that after 20,000 or so incoming CoG values, this procedure leads to the emergence of categorical behaviour at SF.

Whenever a CoG value is applied to AudF, this produces an activity pattern at AudF of the form shown in Figure 14. The CoG value

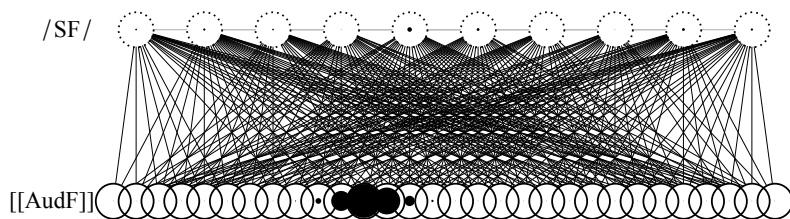


Figure 14:
Applying
an input

is an (unrounded) node number between 1.0 and 30.0. In Figure 14, the CoG value is 12.3. The nodes in the vicinity of location 12.3 are then activated according to a Gaussian shape with a height of 1 and a standard deviation of 4 percent of the extent of the continuum (i.e. $0.04 \cdot 29 = 1.16$ nodes), mirroring the width of a region of activity on the basilar membrane.⁴ This activates node 12 most strongly (at a distance of 0.3), node 13 a bit less strongly (distance 0.7), node 11 (distance 1.3) even less strongly, and so on; the activities of nodes further away than nodes 14 and 10 are too weak to be visible in the figure. Independently of whether the centre of the Gaussian bump is located on a node or somewhere between two nodes, the total activity in AudF is always around 2.908 (if the CoG value is very close to the left or right edge, the total activity is less, because a part of the bump is cut off).

After the input is applied to AudF, the AudF nodes in Figure 14 are clamped (as shown by the solid edges of their circles), i.e. their activities are kept at the applied values (those seen in the figure) throughout the spreading of activities. The SF nodes, by contrast, are unclamped (as shown by their dotted circumferences), i.e. their activities adapt to the activities of the AudF nodes as well as to the activities of other SF nodes throughout the spreading of activities. The activities at SF start at zero, after which the activities of AudF excite the nodes at SF according to equation (24) (with positive w_{ij}); as SF activity grows, the SF nodes start to inhibit each other, again according to equation (24) (with negative w_{ij}). These excitations and inhibitions occur with a spreading rate of 0.01, with the summation in (24) running over all AudF and SF nodes. The computation of activity from excitation fol-

⁴If node 1 is at 16 ERB (1095 Hz), and node 30 is at 33 ERB (9611 Hz), then this standard deviation is $0.04 \cdot 17 = 0.68$ ERB.

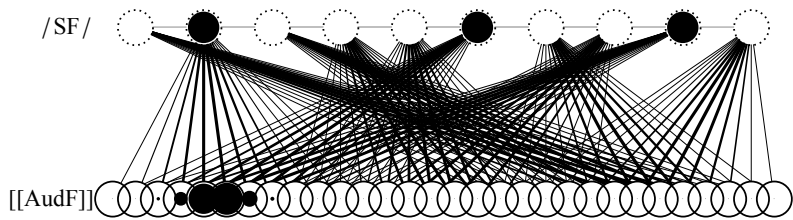
lows equation (25): the activities are clipped from below at zero (i.e. negative activities are not allowed, but large positive activities are). Spreading goes on in this way for 100 time steps. The result is that ultimately the whole network would move toward equilibrium, if the spreading were not truncated after 100 time steps.

After activity spreading, the network is allowed to learn by the inoustar rule, i.e. equation (26) applied to all 300 connections between AudF and SF, with a learning rate of $\eta_w = 0.01$. There is only one learning step per incoming CoG value.

5.4 Result after learning: the perception of three categories has emerged

After 20,000 incoming CoG values, the weights of the network have become those in Figure 15. At SF, nodes 2, 6 and 9 (i.e. the three that

Figure 15:
A network that has been trained on three peaks and has thereby become capable of categorizing



are on in the figure) have become associated to low ([ʒ]-like) CoG values, nodes 4, 5 and 8 to intermediate ([ç]-like) CoG values, and nodes 1, 3, 7 and 10 to high ([s]-like) CoG values. In other words, each node at SF has specialized in one of three areas of AudF, and each of these three areas of AudF is associated with approximately one third (i.e. three or four) of the SF nodes.

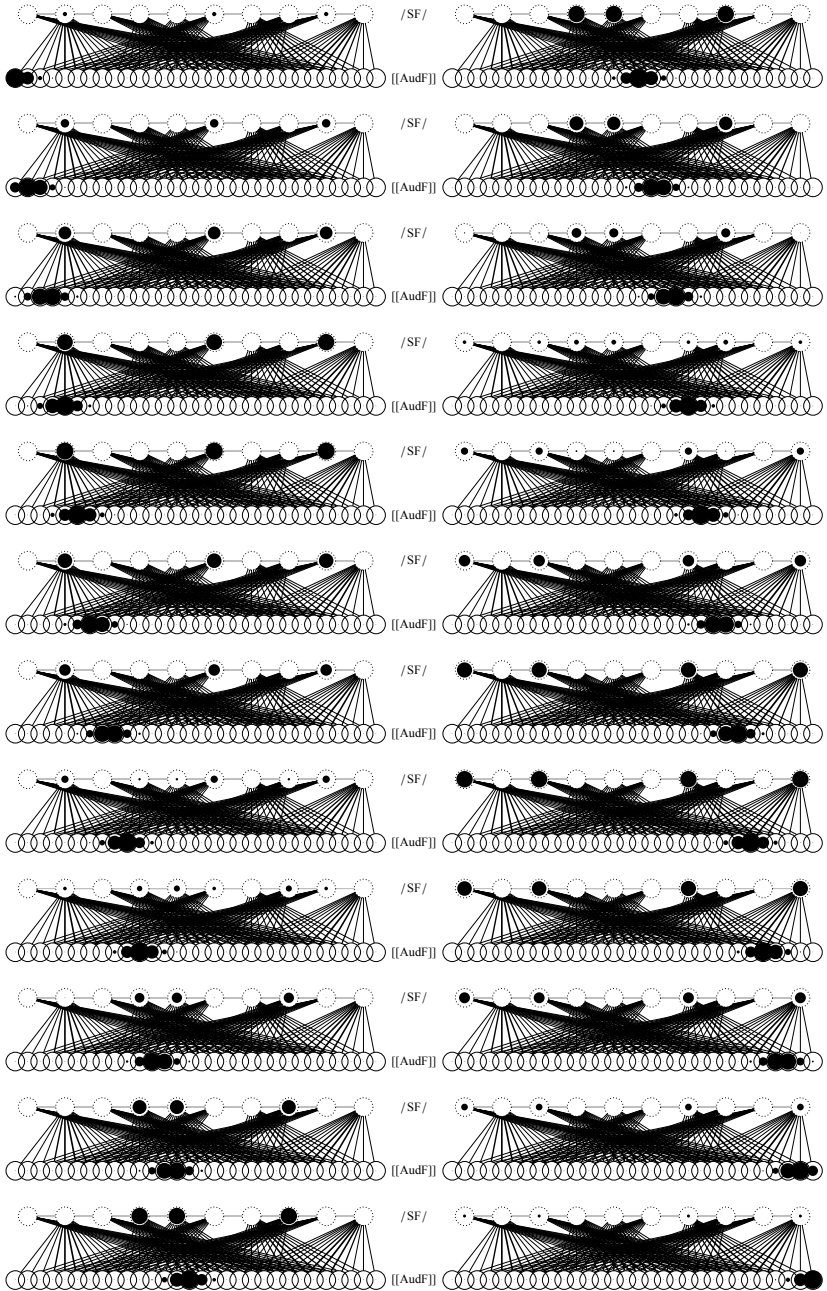
This situation of dedication of SF nodes to AudF areas causes the trained network to *behave categorically* in perception. We can see this by applying a large number of different input patterns to AudF and examining the resulting output patterns at SF. In Figure 16 we pace a local activity pattern through the whole auditory continuum from the lowest values (top-left picture) to the highest values (bottom-right picture). We see that the output at SF favours exactly three patterns of activity. For any low auditory value, only SF nodes 2, 6 and 9 switch

on; for any mid value, only nodes 4, 5 and 8 switch on, and for any high value, only nodes 1, 3, 7 and 10 switch on. Since activity patterns are the brain's way of representing behaviour, the favoured 2–6–9, 4–5–8 and 1–3–7–10 patterns at SF represent favoured (or “attractive”, or “stable”) types of behaviour at SF, or, in other words, three *categories* (when the information proceeds up toward Underlying Form, Morphemes, and perhaps higher semantic areas of the brain, there will still be only three types of behaviour in those higher regions, because according to the adjacency property illustrated in Figure 1, those higher levels of representation cannot “look through” SF toward AudF). We can therefore call the first favoured behaviour at SF the “2–6–9 category”; it replicates the /s/ category of the language of the parents. Likewise, the 4–5–8 category represents the parents' /ç/ and the 1–3–7–10 category represents the parents' /s/.

The final network of Figure 15 differs from the networks we discussed in Sections 2 through 4 in that the network of Figure 15 no longer represents a phonological category as a single node, but represents phonological categories in a *distributed* manner, namely as two or three SF nodes each. The same is true of AudF: every incoming sound activates more than one node at AudF. A biologically desirable property that such a network displays is *redundancy* in the representation of patterns: if a couple of AudF nodes die, and one SF node dies, the network will still perform its classification task quite well. In Figure 15, for example, every incoming sound will still generate one of three stable patterns at SF. For purposes of category creation, it is even more important that having 10 SF nodes allows any number of categories to be created: rather than forcing the existence of 10 categories, as would be the case for the networks in Sections 2 through 4, the 10 nodes are divided roughly equally among the two or three or five categories that the peaky language distribution suggests there are.

We conclude that there come to be three types of stable behaviour at SF, to be interpreted as three phonological categories. This categoricity comes about gradually during learning. On the way to the final state of the network, the categoricity of the behaviour increases from nothing (the random behaviour at SF that the network of Figure 14 exhibits) to almost perfect (the behaviour of the eighth picture in Figure 16, which has the same input). Thus, **categoryhood is gradient** in this model: during development, the patterns gradually grow from

Figure 16:
Pacing
the trained
network
through the
Auditory Form
yields three
types of patterns
in the Surface
Form



being less attractive to being more attractive, without there being a moment at which one can say that a category has just come into existence. During the acquisition period, the behaviour therefore changes from random via slightly categorical toward very categorical.

How does category creation work?

5.5

After seeing *that* category creation works, we would like to understand *why* it works.

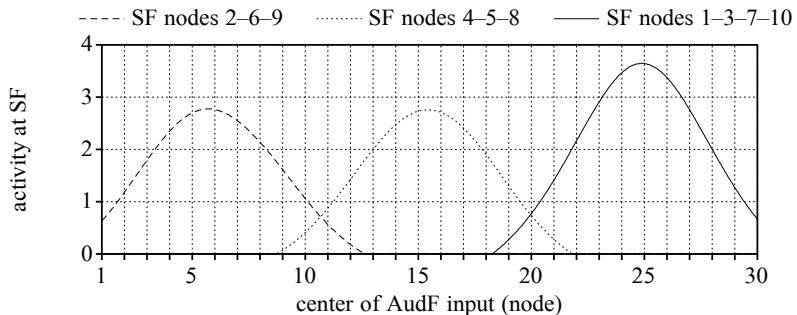
The most crucial aspect of the network is the competition at SF. This is known from competitive learning models (Grossberg 1976, 1987; Rumelhart and Zipser 1985; Guenther and Gjaja 1996), which typically implement competition by “manually” setting the most active output node (the “winner”) to an activity of 1 and all other nodes (the losers) to an activity of 0. This winner-takes-all procedure is an extreme version of what we use in this paper, and could be implemented in our case as follows: if after 100 steps of activity spreading to SF (as in Figure 14) we drastically severed all connections between the SF level and the AudF level, and thereby allowed activity to spread only between the nodes of SF, then the inhibitory connections within SF would reduce the activities of all nodes as long as more than one node were on; one by one, the weakest nodes would drop to zero activity, and this reduction would stop when only a single node were left, which would have some nonzero activity remaining; this node would be the one that had the highest activity to start with. Our exhaustive inhibitory connection scheme, which does not use winner-takes-all, can be seen as a gradual version of the original competitive learning models; it is a more “automatic” version of competition, because no artificial temporary connection severing is necessary; still, the competition is guaranteed by the existence of inhibitory connections within SF.

In the original competitive-learning models, the winner-takes-all step is followed by a learning step in which the weight(s) of the connection(s) between the active input node(s) and the winner are increased and the weights of the connections between the inactive input nodes and the winner are decreased, a procedure identical or similar to instar learning. Our gradual version of competitive learning with

inoutstar learning creates distributed categories by the same cause, which we try to explain now.

First imagine that there is only one node at SF. In Figure 14 this node will be active whenever a part of AudF is switched on. The connections from this node to AudF regions that are often on will strengthen more than the connections to AudF regions that are rarely on. After some time, the connection weights for the various AudF nodes will come to follow a pattern similar to the summed curve in Figure 13. This means that if we pace through AudF as in Figure 16, the activity of the single SF node will go up and down along with the peaks in the summed distribution. Hence, activity in the single SF node will be highest at the three tops of Figure 13. Imagine now that there are 10 nodes at SF, but there is no inhibition between them. Every node at SF will come to be connected to AudF in the same way as the single SF node in the previous imaginary network. Consequently, each node will be activated by AudF according to the summed curve in Figure 13. Imagine finally that an inhibition between all the nodes at SF is introduced. This inhibition militates against different SF nodes being on at the same time. As a result, assuming small random differences in activities between SF nodes (caused by the different random initial weights), different SF nodes will come to specialize in different regions of AudF, so that they can be on at different times (the sum of all activities at SF will still follow Figure 13; see Figure 17). A further question is: why does an SF node specialize in a contiguous *region* of AudF, rather than, say, in the left half of the first peak and the right half of the second peak? This is because of the width of the activity on AudF: the left half of the first peak tends to be active when the

Figure 17:
The degree
of activation
of each
of the three
categories,
as a function
of the auditory
input



right half of the first peak is somewhat active as well. In other words, (spectrally) adjacent nodes at AudF have correlated activities, just as (spatially) adjacent hair cells on the basilar membrane do. If in our simulations we had instead activated only the node nearest to the selected CoG, no categorization of regions would have occurred.

The assignment of each SF node to an AudF region is not random: in fact, the SF nodes tend to become equally divided between the three categories. If each SF node were independently tuned to a region of its choice, we would find that in 5.2% of the experiments an ambient category would be presented by 0 nodes. We never find this; the division 4–3–3 is by far the most common. The cause of this equal division is the inhibition.

Investigating the network's detailed perceptual behaviour

5.6

In Figure 16 we can see that when the incoming sound paces through the auditory continuum, the degree of the activities within a category at SF is not always the same. The activities of the 2–6–9 (/ʒ/) category are much higher if AudF node 6 is on (where the peak of the first category is located, as can be seen in Figure 13) than if AudF nodes 2 or 10 (where the margins of the first peak are located) are on. Thus, the first category is much more strongly activated by the relatively common AudF patterns around node 6 than for the less frequent AudF patterns around nodes 2 and 10.

At the category boundaries, a mixed type of behaviour appears. For AudF nodes around 10 and 11, SF shows a combination of the 2–6–9 (/ʒ/) category and the 4–5–8 (/ç/) category: apparently, both categories are activated to some (small) extent. Observationally, this situation can correspond to an uncertainty in the listener about what the category is; an interpretation of this is that the SF candidates /ʒ/ and /ç/ both move on toward UF, activating in the lexicon words with underlying |ʒ| as well as words with underlying |ç|. Since AudF node 11 can indeed represent either of two categories from the language environment (speakers produce such auditory values sometimes when intending /ʒ/, sometimes when intending /ç/), such uncertainty is adaptive and appropriate (e.g. the Ganong effect mentioned in Section 1.3.2). Something similar happens for AudF nodes around 20 and

21: the listener's reaction at SF is a mixture of the 4–5–8 (/ç/) and 1–3–7–10 (/s/) categories.

Figure 17 shows how strongly every possible location of the Gaussian input bump at AudF activates each of the three categories at SF (after 100 spreading steps, with a spreading rate of 0.01). Thus, a bump centred at AudF node 10 causes activities of approximately 0.37 in nodes 2, 6, and 9, so that the summed activity for category 1 (= nodes 2–6–9) is 1.1, as shown in the figure. Likewise, category 2 (= nodes 4–5–8) has a summed activity of 0.4 in its three nodes, and category 3 has no activity for AudF node 10 in any of its SF nodes 1–3–7–10. In Figure 17 the activity was measured for 581 centre locations, namely for AudF nodes 1 to 30 in steps of 0.05 node.⁵ The peak is higher for category 3 than for the other two categories, because this category is formed by four SF nodes instead of three.

The activity curves follow the input distributions of Figure 13 closely, with the tops at approximately the same locations. A difference with the distributions is that the activities go to zero at a distance of approximately 7 nodes from the tops. This is due to the inhibitory behaviour of the negative connection weights within SF, which e.g. renders the excitation of category 1 negative for all AudF locations above 13. The zero values then follow from the clipping mentioned in Section 5.3.

If we interpret the activities of Figure 17 as relative probabilities of perceiving a certain incoming AudF as any of the three categories (Section 2.5), we can draw the *identification curves* of Figure 18. These curves tell us how likely any incoming AudF is perceived as category 1, 2 or 3. For each category, the curve is computed by dividing the activity curve for that category (Figure 17) by the sum of the three activity curves.

The shapes of the identification curves are similar to those found with human participants in the lab; for this reason, Figure 18 labels the three categories with the language-specific phoneme labels that human participants would have to choose from (a difference with the

⁵The smoothness of the curve shows that there is no major influence of the discretization of the input continuum on the activity curves. This desirable behaviour is caused by the fact that the bumps at AudF have a Gaussian shape. With different input shapes, the activity curves at SF may display ripples.

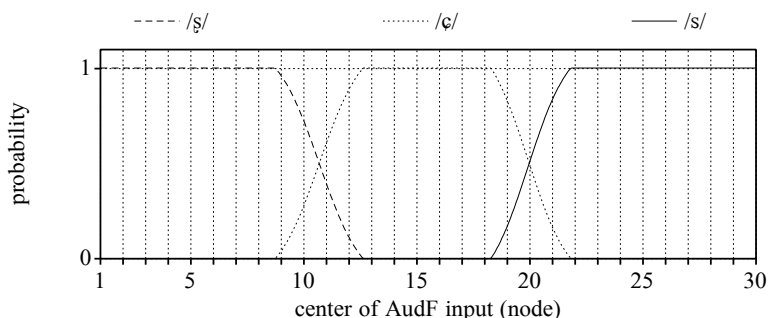


Figure 18:
Identification
curves after
distributional
category
learning

human curves is that the curves in Figure 18 go to their extreme values abruptly; this difference vanishes when we realize that sounds played in the lab are supplied with transmission noise before they are converted to AudF values in the listener; another difference is that the extremes in Figure 18 are exactly 0 and 1, which is because we assumed a perfect reporting mechanism).

In the lab, humans can report not only the category they think they hear, but also how good the sound heard is as a token of that category. Such *goodness judgments* can be thought of as following the curves in Figure 17: if the listener has access to an inspection device that computes the total activity of a category at SF,⁶ she will be able to calculate any activity value in Figure 17 and trivially employ that value as a reportable category goodness between 0 (poor fit to the category) and 1 (perfect fit). Relatedly, since the peaks of the curves in Figure 17 are at or near the most frequent exemplars of the categories (Figure 13), the best exemplars in a *prototype task* will be those same most frequent exemplars (this statement will be amended in Section 6.5).

Investigating the network's behaviour: production

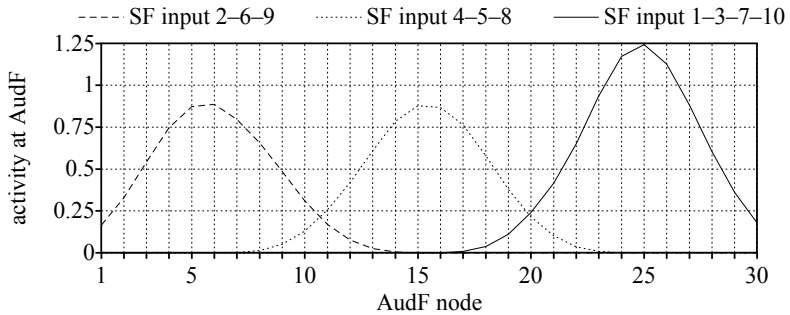
5.7

The network is bidirectional, so it can be used to model not only perception, as in the previous section, but production as well. To measure

⁶A goodness computation for e.g. the 2–6–9 category of Section 5.4 can be performed by a simple network connected to SF, with connection weights of 1 to SF nodes 2, 6 and 9, and connection weights of 0 to the other seven SF nodes. Follow-up simulations by Chládková (2014) have shown that such weights are learnable in a three-level BiPhon model.

the production of a category, we can clamp the SF nodes of that category (i.e. nodes 2–6–9 or 4–5–8 or 1–3–7–10) at an activity of 0.8 and compute what the activity at AudF will be after 100 spreading steps. The three results are in Figure 19.

Figure 19:
The activity at AudF, as a function of a three- or four-node input at SF



The learner turns out to produce the categories in much the same way as her parents, if the activities of Figure 19 are interpreted as relative probabilities. As a result of the inhibition, the standard deviation is somewhat smaller than that of the parents, but this will be counteracted (as it was in the OT model by Boersma and Hamann 2008) by the transmission noise that has to be added to the AudF values drawn from Figure 19 once we want to model multiple generations of learners.

The result in Figure 19 is not realistic. Considerations of articulatory effort will shy the learner’s production away from the edges. We can model this with the network in Figure 20, in which the influence of the sensorimotor knowledge and the knowledge of articulatory effort is summarized (and extremely simplified) as a single clamped ArtF node that has strong inhibitory connections to peripheral AudF nodes and weak inhibitory connections to central AudF nodes. If the inhibitions follow a parabola, with a weight of -0.1 in the centre and -1.6 at the edges, the AudF output of the 2–6–9 category will be that shown in Figure 20.

The AudF activity for all three categories is summarized in Figure 21. The auditory realizations of the two outer categories now avoid the edges: when compared with Figure 19, their peaks slightly moved inward, and their medial tails are much longer than their lateral tails.

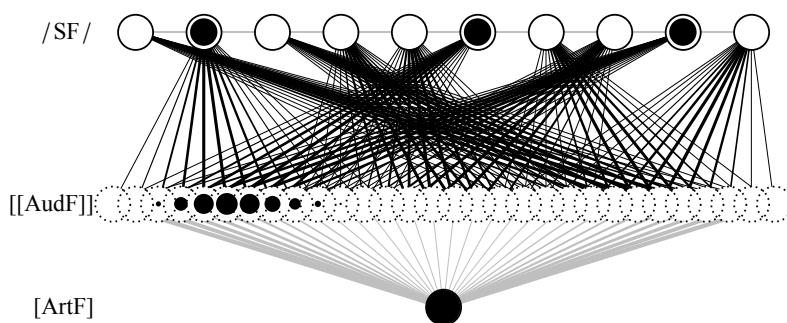


Figure 20:
Network
for production

This means that the learner will on average produce rather more central AudF values than her parents.

If the sound shift of Figure 21 goes on for a number of generations, the three peaks will come so closely together that a new learner cannot create three categories any longer. Inevitably, iterated learning with the procedure of Section 5 must lead to merger. However, information from above SF will come to the rescue, as Section 6 will show.

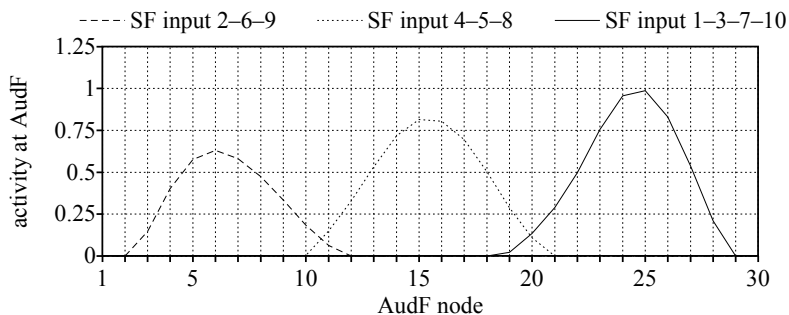


Figure 21:
Production
influenced
by articulatory
effort

It is important to note that the network of Figure 20 is compatible with the results of the two-layer network of Figure 12. That is, the network of Figure 20 works in exactly the same way as that of Figure 12 for the purposes of Sections 5.4–5.6 (and also Sections 5.8–5.10), because adding an articulatory representation below AudF cannot influence the perception process in our simulations, where the auditory representations, which lie in between the higher and the articulatory representations, are clamped (held constant) during activation spreading. An interesting variant of our simulations would appear if we let

the auditory representations settle freely instead (as in e.g. McClelland and Elman 1986), in which case their connection to the articulatory representations (i.e. sensorimotor knowledge) will slowly (during activation spreading) move the auditory representations toward gestures that the listener finds easy to pronounce, which again will influence the higher (e.g. phonological) representations. With this interactive scenario, low-level perception from AudF to SF would partly go *through* articulatory representations, without articulatory representations having to lie *between* AudF and SF. Hence, several phenomena that have been brought forward by proponents of motor theory (Lieberman and Mattingly 1985) or direct realism (Fowler 1986; Best 1995) in favour of articulatory representations mediating between AudF and SF can also be explained when articulatory representations lie outside the direct AudF–SF path, as was pointed out by Boersma (2012). Simulations of such phenomena fall outside the scope of the present paper. The main point we want to make here is that the network of Figure 20, not that of Figure 12, is the complete network that exhibits all the properties discussed in Sections 5 and 6.

5.8 *Replicating experimental data: categorical perception*

It is known that listeners can more easily discriminate two auditory forms that map to different phonological categories than two auditory forms that map to the same category (Lieberman *et al.* 1957). The trained network of Figure 15 can replicate this behaviour, under the assumption that a listener’s report whether two sounds are the same or different rests on her inspecting her SF, not her AudF. That is, when responding to the task of reporting whether two sounds are the same or not, the listener is actually reporting how different she judges the two surface forms instead.

To replicate this with the network of Figure 15, we first compute the average absolute difference between the activities of the SF nodes in the first two pictures in Figure 16. Node 1 (at SF) is activated equally (namely, 0) in both pictures, but node 2 is activated a bit more (by 0.2) in picture 2 than in picture 1. On average, the activity of a node in picture 2 differs from the activity in a node in picture 1 by an amount of 0.03. The difference between picture 3 and picture 4 is even smaller,

the distance between adjacent peaks, the depth of the valleys becomes 0.27, and the network learns as well as before (and on average slightly faster), coming up with three clear categories (4–3–3 or 4–4–2) in all (20 out of 20) replications; the same goes for data with a valley depth of 0.02. This is not surprising: sharper peaks yield better category discriminability, so we expect better learning, if anything. On the other hand, raising the standard deviation of the data to 40% of the peak distance increases the valley depth to over 0.80, and our network no longer learns equally fast: in half of the replications, the situation after 20,000 data is four insecure categories that continuously slide into each other while scanning; however, this is simply a common intermediate learning stage (also often seen after 10,000 data in the simulations of Section 5.4), and correct triple categorization always emerges when we continue to train the network toward 100,000 data (two of the four categories gradually merge). Generalizing, we can say that our network can learn three categories if the distribution shows any visible valley, although learning is faster if the valley is deeper.

To see whether having a valley-depth cut-off is bad, we compare our network to results from the literature with human subjects. Experiments that showed distributional learning have usually been performed with only two peaks, with a valley depth of 0.25 (e.g. Maye *et al.* 2008). For two categories with the same standard deviation as in Figure 13 (i.e. with a much deeper valley of 0.16, because the peaks are spaced 14.5 nodes apart), our network always (in 20 out of 20 replications) succeeds in learning the two categories perfectly, with 5–5, 6–4 or 7–3 divisions of SF. For two categories with a valley depth of 0.65 (i.e. a standard deviation 1.5 times that of the peaks in Figure 13), our network also learns well, although in a minority of replications it does so via temporarily (after 20,000 data) having a small additional category on the shoulder of a big category (when learning continues toward 100,000 data, this shoulder category merges with the main one, so this is just a sign of the expected slower learning). For valley depths used in experiments with human participants, our network therefore performs well.

5.9.2

Number of categories

In Section 5.1 we asserted that our network should be able to learn languages that have between one and four categories along the contin-

uum. In Section 5.4 we saw that our network learns three categories from a three-peaked distribution, and in Section 5.9.1 we saw that it learns two categories from a two-peaked distribution. When confronted with a single broad peak, the network becomes a partly “auditory” listener, with continuously changing output patterns when we scan along the continuum, and with a discrimination curve, rather different from that of Figure 22, that has either a peak in the middle or two peaks around the middle (closer together than in Figure 22). This variation between learners might mirror to some extent the behaviour of participants confronted with monomodal distributions in a distributional experiment, although we can make no numerical comparisons at this point.

Our network works well for four categories with the same valley depth of 0.65 as in Figure 13, i.e. with a standard deviation of 3/4 of that of the three peaks in Figure 13: the learner divides up SF as 3–3–2–2 or sometimes 4–2–2–2, and has three discrimination peaks. With five categories, most of the learners show so much overlap between some adjacent categories that their discrimination curve has only one or two peaks instead of four; this loss of categorizability is OK, because we know of no languages with more than four CoG categories, which is why we designed our network with only 10 nodes at SF (Section 5.1).

Number of nodes at SF

5.9.3

With 10 nodes at SF, our network can learn up to four categories reliably (Section 5.9.2). It is interesting to see whether the limit of four is inherent to our type of network, or whether more categories can be learned if we modify its hyperparameters. And indeed, when we raise the number of SF nodes to 30, we can stably create up to seven categories (four categories do e.g. 7–7–6–4, with 6 nodes unconnected; seven categories do e.g. 5–4–4–4–4–3, with six clear discrimination peaks), under the condition that the valley depth is kept low enough to compensate for the increased effect of the 1.16-node smearing on the basilar membrane (Section 5.3). Learning eight categories usually works perfectly, but slightly fails in a minority of cases (overlapping patterns at SF for adjacent categories; e.g. one of the nodes stays on for two categories, so that one of the seven discrimination peaks is lower than the others), apparently because even with very low ambi-

ent standard deviations, i.e. valley depths close to 0, the pooled distribution of activities at AudF comes to show rather shallow “basilar valley depths”. With a much greater granularity both at AudF and at SF, namely with 100 AudF nodes and 100 SF nodes, up to ten categories can be learned. We conclude that our number of SF nodes (namely, 10) limits the number of categories to 4, and the physical characteristics of our auditory continuum (namely, 0.68 ERB of basilar spreading; see footnote 4) limits the number of categories to 8 or 10.

These numbers do not seem to contradict any known fact about phonological inventories. For instance, a language with four vowel heights will have their F1 values spaced 2.1 ERB apart, if high vowels have an F1 of e.g. 300 Hz (7.3 ERB) and low vowels have an F1 of 800 Hz (13.6 ERB), and the mid vowels are equally spaced between them, i.e. at 9.4 and 11.5 ERB. This 2.1 ERB is approximately how far the peaks are spaced in our simulations with eight categories (namely, a 17-ERB range divided in eight equal steps). This can explain why languages with five vowel heights are very rare. Precise numerical fits with vowel data will have to be relegated to future work.

5.9.4

Inhibition within SF

The simulations of Section 5.4 use an inhibitory weight of 0.1 between nodes at SF. This value of 0.1 works for a great variation of valley depths and numbers of categories in the data, and for quite varying numbers of SF nodes. The value itself is not robust. If we lower this inhibitory weight to 0.01, then all SF nodes are excited equally (no off-and-on pattern as in the simulations above), and this same egalitarian pattern appears in exactly the same way for any AudF, so the number of categories created is zero (or, equivalently, one). If we raise the inhibitory weight to 1 or 2 or 10, then in the great majority of replicated simulations four categories emerge, and each of those categories has only one node at SF; that is, six of the ten nodes will never switch on. An inhibitory weight of 0.05 will generally work well, but may cause, in a sizable minority of learners, a bit of overlap in SF patterns of adjacent categories, and differences in the heights of the two discrimination peaks (in Figure 22 the two peaks are equally high). An inhibitory weight of 0.2 makes the two-category case poorly learnable

(e.g. three discrimination peaks). Thus, the inhibitory weights have to have a value around 0.1, not very well allowing a factor of 2 off in either direction. We speculate that this fine-tuning of excitability of neurons may well correspond to something in biological neural systems, which function best in a state somewhere between anaesthesia and epilepsy.

We can conclude that distributional learning on a single continuum is a bit brittle in our network (in the sense of requiring a notable valley in the distribution of CoG values, and some tuning of the degree of inhibition), an observation that corresponds to what is found in a literature overview on experiments with human subjects (Wanrooij 2015: 35). In real-life acquisition, there will usually be multiple auditory continua, plus contextual information, which could make category learning easier even in cases where distributional valleys are shallow or even non-existent.

The rectifier

5.9.5

The activation function follows (3) or (25), i.e. the activity of a node is always made non-negative. An arguably simpler activation function is the identity function, as in (2), i.e. the activity of a node equals its excitation. It turns out that simulations with such an identity activation do not display stable category creation. To understand this, consider a network with identity activation that is in an initial situation in which the input to SF node 1 is 0.1 (e.g. the weights times activities of all the AudF nodes to SF node 1 sum up to 0.1, which is a possible result of initial weights being uniformly distributed between 0 and 0.1), and the inputs to all other nodes are 0 (which can happen, for instance, if all weights from AudF to these other SF nodes are zero). After the first activation spreading step with a spreading rate of 0.01, the activity of SF node 1 will be 0.001, and the activities of all other SF nodes will still be 0. After the second step, the activity of SF node 1, according to (24), will be 0.00199, but all other SF nodes will be inhibited by SF node 1, i.e. each of their activities will be $-0.1 \cdot 0.01 \cdot 0.001$ (the inhibitory weight times the spreading rate times the activity of SF node 1 after one step) = -0.000001 . After infinitely many steps, the activity of SF node 1 will be $2/19$, and that of each other SF node will

be $-1/171$.⁷ This example serves to illustrate that if we allow negative activities, such negative activities will actually occur. Combined with inhibitory weights, the occurrence of negative activity results in increasing the positive activity of other nodes, defeating the whole idea behind inhibition. This has happened here to SF node 1, which with the rectifying activation function would have ended up with an activity of only 0.1 instead of $2/19$. Also, negative activities defeat the purposes of inoutstar learning, including the idea that weights reflect something close to a conditional probability; with negative activities, weights can easily fall below 0 or rise above 1, and in our simulations with identity activation they indeed tend to do so without bounds, leading to chaotic restructurings of the network upon each learning step. We conclude that no stable learning is possible with an identity activation function, while stable learning is possible with the next simplest activation function, namely the rectifying activation function used throughout Section 5 and 6.

5.9.6 Learning rules

We have seen that category creation works well with the inoutstar rule. It does not work with the simpler outstar learning rule: weights and activities blow up. However, category creation turns out to work well with the equally simpler instar learning rule, if we assume that AudF is the input and SF is the output. This could be expected on the basis of earlier competitive learning studies. The only reason why we use inoutstar instead of instar is that it is symmetric and can therefore handle the cases of Section 6 as well as those of Section 5.

5.10 Plasticity

After learning three categories in her native language environment, the learner might move to an area of the world where four categories are spoken. The network turns out to adapt itself accordingly. If the middle category has four SF nodes, they will split up 2-2. If the middle

⁷One can show that if the excitation of SF after 1 activation spreading step is e_j , the final activity will be $\frac{1}{1-\alpha} \left(e_j - \frac{\alpha}{1+\alpha(N-1)} \sum_i e_i \right) / \text{spreadingRate}$, where α is the inhibitory weight (i.e. 0.1) and N is the number of SF nodes (i.e. 10).

category has three SF nodes, any of three things can happen: the nodes of the middle category split 2–1; the nodes split 2–1 but the second middle category borrows a node from its neighbour; or the category with four nodes splits 2–2.

If, conversely, a learner with four categories moves to a place with three, she will merge two categories, typically the two in the middle.

If all three nodes of the second category (4–5–8) die, the remaining seven nodes will divide themselves up between the three categories. If the whole of the higher-frequency third of AudF dies, its three nodes will be recruited by the first and second categories.

We conclude that the network has a high degree of plasticity, adapting itself to changes in the environment as well as to changes in its own structure.⁸

Comparison with earlier models

5.11

A potential early stage of categorical perception, the *perceptual magnet effect* (Kuhl 1991), has been modelled with neural nets before by Guenther and Gjaja (1996). This work had four aspects that make it difficult to use their model for our purposes. First, the learning rule was instar, which does not work for auditory dispersion (Section 6). Second, the inputs were only four AudF nodes, with a formant value unrealistically represented by the activity levels of two AudF nodes rather than by an array of nodes as here. Third, the state of SF was selected less realistically (i.e. more “manually”) than here, namely by setting all activities that did not exceed a certain threshold to zero (rather than by mutual inhibition). Fourth, the magnet effect was established by computing a “population vector” based on a computation of auditory distance; in our case, a “warped” AudF can be directly computed by clamping an AudF to an incoming CoG value, then computing the output SF, then clamping the SF at this output, then unclamping AudF and having activity spread back to it from SF; this reflection works correctly thanks to the bidirectionality of the connections, which Guenther and Gjaja could not implement.

⁸In human learners, plasticity may well decay with age, so that adaptation to changing environments slows down as the child grows older. Modelling this lies outside the scope of the present paper.

Some aspects of our model are shared with the TRACE model by McClelland and Elman (1986), the most notable being upward activation spreading. The critical difference, however, between our model and TRACE, in the light of Section 5, is that TRACE featured “local” (i.e. non-distributed) representations and therefore had to work with pre-given categories. Even if TRACE had come with a learning algorithm, which McClelland and Elman did not provide, TRACE thus could not have handled the main objective of Section 5, which is the emergence of categories. A phenomenon that TRACE did successfully account for is the *Ganong effect* (Ganong III 1980), by which low-level perception (for us, the mapping from AudF to SF) is influenced by top-down information (for us, from Morphemes down to SF) about the existence of lexical items (for us, Morphemes). Simulating the Ganong effect in our model would equally require a third level of representation that encodes meaning and feeds information back to SF. This should be possible, because the BiPhon model of Figure 2 provides the required level of representation (UF and/or Morphemes), and adding such a level above the SF of the present section would provide the required feedback as a result of the bidirectionality of the connections. While the simulations presented here do not address the exact effects of the top-down feedback from a third level (though see Chládková 2014 for showing that top-down effects do happen when we add a third layer), they do illustrate that our model satisfies another prerequisite for the Ganong effect, namely ambiguity at the middle level. In TRACE, the Ganong effect is critically dependent on ambiguity at the middle level of representation, which is resolved by top-down activation from existing lexical items. In our simulations, ambiguity at SF occurs in the mixed excitations visible in Figure 16 (e.g. the 4th picture from the bottom in the left column, and the 4th picture from the top in the right column) and Figure 17 (around nodes 11 and 20), which occur in response to AudF input that lies in a distributional valley (i.e. near a boundary between two ambient categories). These two-level simulations in our model thus lay the foundation for a full simulation of the Ganong effect, which has to be postponed to future work that investigates three levels of representation.

AUDITORY DISPERSION

6

Auditory dispersion is a phenomenon in sound change whereby the auditory correlates of phonological elements become optimally distributed along one or more auditory dimensions. The emergence of auditory dispersion over the generations was handled successfully in BiPhon-OT (Boersma and Hamann 2008). In this section, we test whether BiPhon-NN is equally capable of doing the job.

Existing work on auditory dispersion

6.1

Languages tend to maximize the auditory contrast between elements in their phonological inventories (e.g. Passy 1890; von der Gabelentz 1901; de Groot 1931; Martinet 1960). In a single auditory dimension, languages favour symmetric inventories whose members lie at equal distances along the auditory continuum, often with a preference for the centre, as in Figure 23. If we take as an example of an auditory

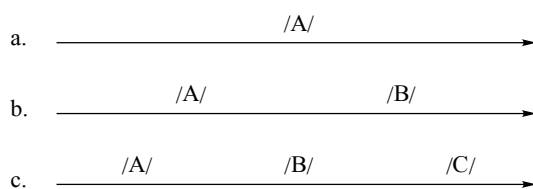


Figure 23:
Typically dispersed
phonological inventories

continuum the voice onset time (VOT) in bilabial plosives, Estonian would be an example of a language with a single category, namely /p/, which is realized with zero VOT (Figure 23a), Swedish exemplifies a language with two categories, namely /b/, realized with negative VOT, and /p^h/, realized with positive VOT (Figure 23b), and Thai serves to illustrate that a language can have the three categories /b/, /p/ and /p^h/ (Figure 23c).

Inventories as in Figure 23 are *optimally dispersed* in the sense that they strike a perfect balance between perceptual clarity and articulatory ease (Lindblom 1986; ten Bosch 1991; Boersma 1998). Practically speaking, optimal auditory dispersion entails that the categories are sufficiently auditorily distinct to minimize confusion in the listener,

and that this distinctivity does not come at too large an articulatory cost for the speaker.

Boersma and Hamann (2008) formalize auditory dispersion within BiPhon-OT as the result of an interaction between cue constraints, whose ranking is a result of optimizing the learner's prelexical perception during acquisition, and articulatory constraints, which aim for articulatory ease. When re-using the perception-optimized cue constraint ranking in production (phonetic implementation), the dispersion effect automatically emerges. With computer simulations, Boersma and Hamann show that optimally dispersed systems are diachronically stable, and that poorly dispersed systems evolve into stable systems within a small number of generations. The BiPhon-OT account is devoid of teleological devices, such as the explicit auditory-distance maximization by Liljencrants and Lindblom (1972), ten Bosch (1991) or Schwartz *et al.* (1997), or such as the OT dispersion constraints proposed by Flemming (1995/2002: MINDIST), Kirchner (1998/2001: DISP), and Padgett (2003: SPACE), whose sole purpose was to preclude categories from approaching each other; in fact, the listener does not have to compute auditory distances at all, as was still the case with some less-teleological methods, such as the agent-based simulations by de Boer (1999) and Oudeyer (2006), and such as Wedel's (2006) exemplar-based account.

6.2

A neural network for auditory dispersion

We will try to replicate Boersma and Hamann's results with BiPhon-NN. We propose that after the unsupervised bottom-up creation of categories of Section 5, the learner creates a lexicon of phonological word forms (at UF), which is capable of "supervising" perceptual learning. That is, once the learner has established a lexicon, the lexicon can provide top-down information, in effect telling the network what phonological category to expect, or what phonological category it should have perceived. To this end, we consider the neural network in Figure 24, which just as the one we used in Section 5.7 has three layers: the phonological surface form (SF), the auditory-phonetic form (AudF), and the articulatory-phonetic form (ArtF).

The network has nine SF nodes for a distributed representation of the categories. As was approximately the case throughout Section 5,

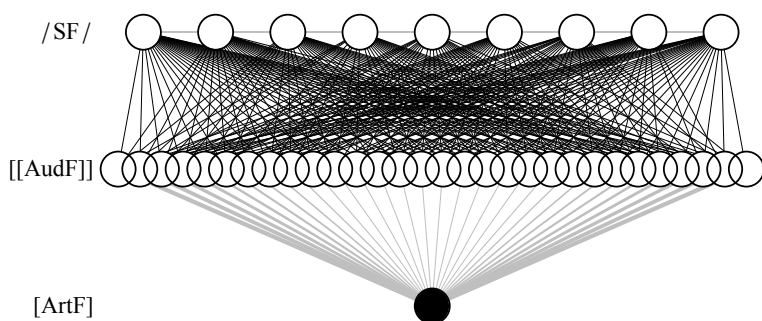


Figure 24:
The initial state
of the neural
network

each discrete phonological category is represented by three SF nodes: category 1 corresponds to SF nodes 1, 4, and 7, category 2 to nodes 2, 5, and 8, and category 3 to nodes 3, 6, and 9. As before, there are inhibitory connections within SF.

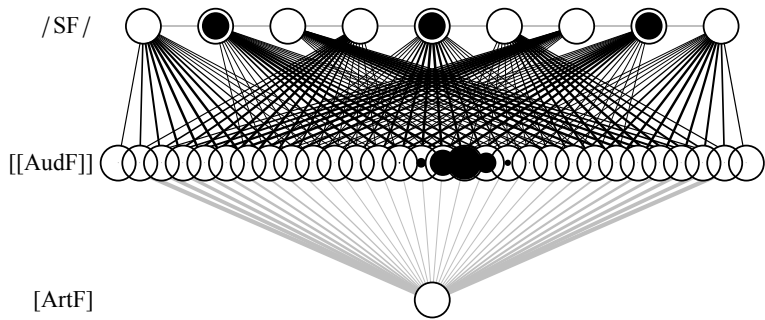
The AudF layer again represents the CoG dimension, sampled again in 30 steps. Each AudF node is connected to all nine SF nodes by excitatory cue connections (drawn in black) whose initial weights have random values between 0 and 0.1. Each AudF node is also connected to the ArtF node by an inhibitory articulatory connection (drawn in light grey); these connections have the same values as in 5.7: they are stronger (i.e. drawn thicker) at the edges of the AudF layer, to represent the idea that the production of a peripheral value requires more articulatory effort than the production of a central value.

Learning to perceive

6.3

The simulated learner will have to establish the appropriate cue connection weights of the ambient language through a process of perceptual learning. Before the learning process begins, we create the initial language: for every category, we determine a normal distribution of input probabilities along the auditory continuum. In each learning step, a combination of a category and an auditory value is selected at random; if a value has a high input probability given the selected category, it is more likely to be drawn. We pair each auditory value with a category because we want the learning process to be supervised by information from “above”, i.e. from the lexicon at and/or above

Figure 25:
The neural
network after
50,000 learning
steps



UF and perhaps also from the phonology of the UF-to-SF mapping: somewhat artificially, we assume that the learner’s lexicon is already in place, i.e. that she knows what category she should have perceived. We switch on the selected AudF nodes as well as the selected category nodes at SF; subsequently, all AudF and SF nodes are clamped, and the weights of the cue connections are updated with the inoustar rule (Section 4.7).

After 50,000 tokens (learning rate = 0.01) from a language with input peaks as in Figure 13, i.e. at 16.667% of the auditory continuum (category 1), at 50% (category 2) and at 83.333% (category 3), the network from Figure 24 comes to look as Figure 25. The left third of the AudF layer is more strongly connected to SF nodes 1, 4 and 7 than to other SF nodes, so the network has learned that low auditory values are most likely to be intended as category 1; likewise, mid auditory values connect to category 2, and high auditory values to category 3, as the language environment dictated.

6.4

Production: the articulatory effect

The network is bidirectional, so it uses the same connections in production as in perception. Figure 26 shows how the network of Figure 25, which has been trained only to perceive, handles production. To see how a category is produced, we switched on its three SF nodes (activity 0.8), as shown by filled disks in the figure, while switching off the other six SF nodes (activity zero), as shown by empty disks; all nine SF nodes are clamped at these values, as shown by solid circles. Now the ArtF node also comes into play, clamped at an activity of 1.0,

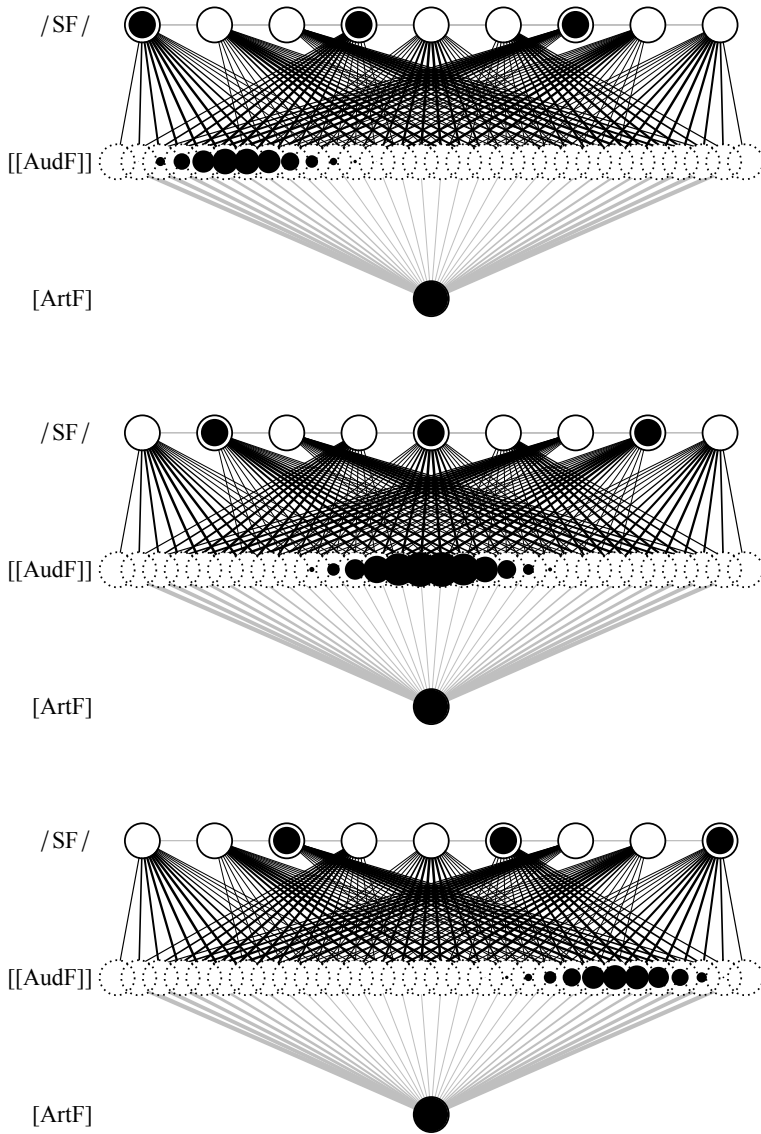


Figure 26:
Output activities
for the three
categories (peaks
in input
distribution
as in Figure 13)

constraining the activities at the unclamped AudF layer. After activity spreads from SF and ArtF to AudF for 500 time steps, Figure 26 shows the resulting activities on the AudF layer (as usual, negative activities are clipped at zero) in the production of each of the three categories. The strongest activities in Figure 26 are between nodes 6 and 7 (i.e. at $5.5/29 = 19.0\%$ of the continuum), between nodes 15 and 16 (50%), and at node 24 ($23/29 = 79.3\%$ of the continuum).

The locations of the strongest activities are important concepts. According to Section 2.5, we can regard these locations as the most probable auditory forms realized in production. When we look at their values, we see that they are different from what the learner has heard in her environment. The learner has shifted category 1 by $19.0\% - 16.7\% = 2.3\%$ toward the centre of the continuum, when compared to her language environment, and she shifted category 3 toward the centre by $83.3\% - 79.3\% = 4.0\%$. These values of 2.3% and 4.0% are typical: if we repeat the experiment, we see that learners will on average shift the two outside categories by 3% toward the centre of the continuum.

It is clear where this shift comes from. As in 5.7, it comes from the articulatory constraints: auditory values around 19% and 79% are just somewhat easier to produce than values around 17% and 83%, so the learner's cue constraints might prefer values around 19% and 79%, but her articulatory constraints move the values away from this effortful periphery.

6.5

Production: the prototype effect

The question is: will learners always shift the categories toward the centre? That would be bad for the future of the language, because a sequence of learners would ultimately make all categories pile up in the very centre of the continuum, where they merge into one.

Fortunately, near the centre of the continuum a different effect counteracts the articulatory effect. Figure 27 shows a network that has learned 50,000 times from a “confusing” language where the distributions of the three categories have peaks at 40%, 50% and 60%. The strongest cue constraints now connect the three categories at SF to much more central auditory values than in Figure 25. The production, however, works as in Figure 28. The strongest activities are at

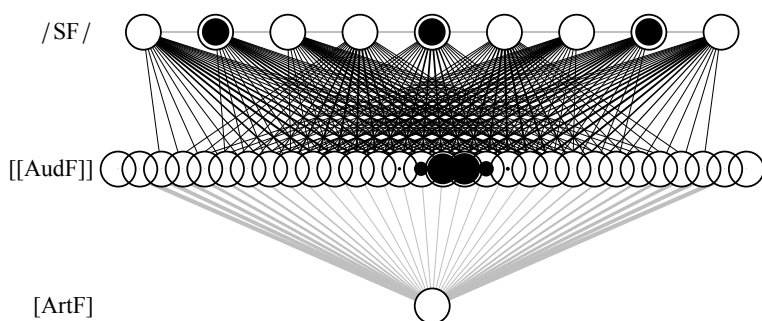


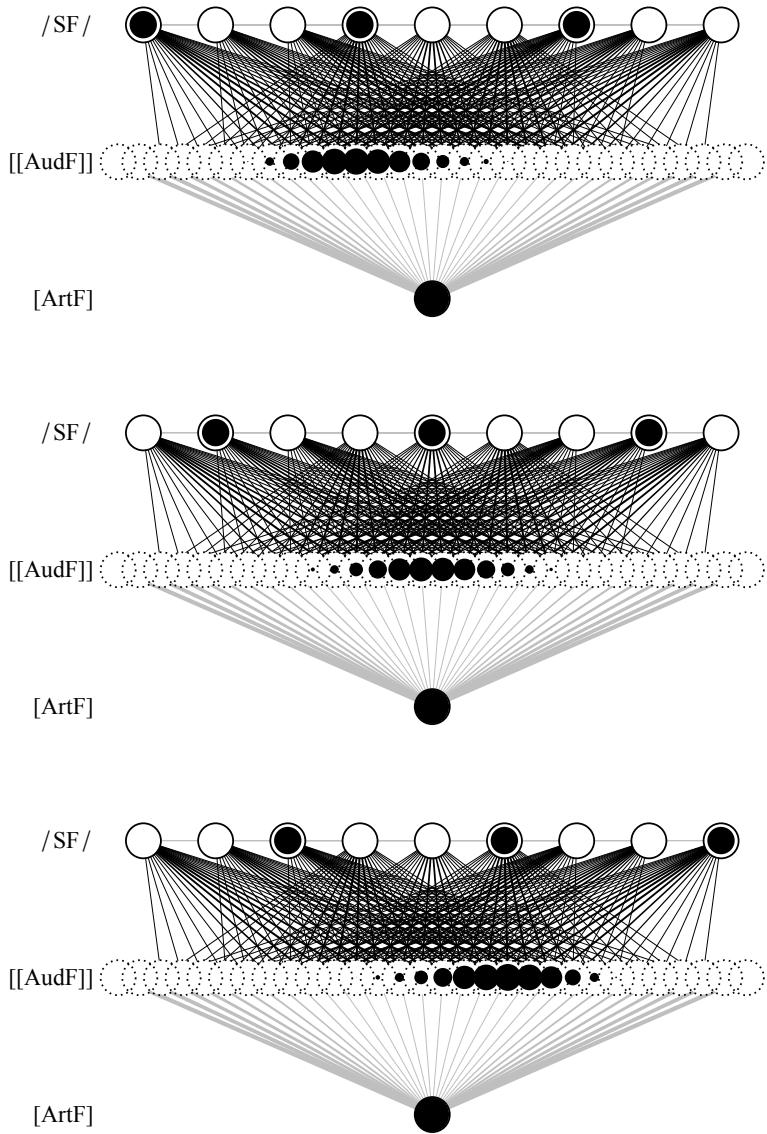
Figure 27:
The neural
network after
50,000 pieces of
confusing data

node 12 (i.e. at $11/29 = 37.9\%$ of the continuum), between nodes 15 and 16 (at 50%), and at node 19.3 or so ($18.3/29 = 63.1\%$ of the continuum). The two outside categories, therefore, have shifted $40\% - 37.9\% = 2.1\%$ and $63.1\% - 60\% = 3.1\%$ toward the periphery of the continuum.

What happened here? The outstar part of the learning algorithm makes stronger connections between AudF and SF if the probability of that SF given that AudF is greater; in fact, the weight moves asymptotically toward the conditional probability of that SF given that AudF. Now, a more peripheral AudF value (say, at 30% of the continuum) is more likely to have been intended as category 1 than a more central AudF value (say, at 40% of the continuum), because around 40% of the continuum we are in a region where the distribution of category 1 overlaps with the distribution of category 2. As a result, the connection between an AudF of 30% and category 1 will be stronger than the connection between an AudF of 40% and category 1. As a result, the production of category 1 will favour an AudF of 30% over an AudF of 40%. This result replicates the observation that listeners choose more peripheral tokens as prototypical than they produce themselves (Johnson *et al.* 1993; explained with BiPhon-OT by Boersma 2006). The inoutstar algorithm employed here does not exhibit this “prototype effect” (Boersma and Hamann 2008) as strongly as the outstar algorithm, but it employs it enough to shift the category by several percent.

Summing up, then, categories whose centres lie near the periphery of the auditory continuum will tend to shift toward the centre, whereas categories that overlap with other categories will tend to

Figure 28:
Output activities
for the three
categories (peaks
in input
distribution as in
Figure 27)



move away from those other categories. Over the generations, an equilibrium will appear where all categories are approximately equally spaced around the centre of the continuum; the distances between the category centres will not depend on where they were in the first generation.

Our simulations show, then, that BiPhon-NN, just as BiPhon-OT, is capable of replicating the emergence of optimal dispersion in phonological inventories. If the network learns the appropriate weights of the cue constraints in comprehension and then “produces” sound using the same connections, any input distribution will evolve into a stable system within a number of generations. It is thus crucial that the neural network is symmetric, as it is in other models that involve both sensory input and production (Kohonen 1984; Wedel 2007).

For more details on the properties of the neural network and learning procedure used here, and for simulations of other inventories, we refer the reader to Seinhorst (2012), who also subjects to closer scrutiny the difference between outstar and inoutstar learning in modelling auditory dispersion.

DISCUSSION

7

One and the same network, with a single learning rule, namely “in-outstar” learning, has turned out to be able to handle both category creation (in a slightly brittle manner) and auditory dispersion (very robustly). While the instar rule would have worked fine for category creation (as Guenther and Gjaja 1996 have shown), and the outstar rule works fine for the emergence of auditory dispersion (as shown by Seinhorst 2012), only the inoutstar rule, which is a combination of the instar and outstar rules, works for both.

On top of the two foci of the present paper (category creation and auditory dispersion), the BiPhon-NN model replicates several realistic behavioural effects, with minimal assumptions and devices. Although the model does not represent or compute auditory distance (as earlier models of both category creation and dispersion did do; see Section 5.11 and Section 6.1), realistic effects of auditory vicinity emerge both in category creation and in dispersion, because the model

automatically learns the correlation between adjacent auditory values in the input (Section 5.5). Although the model employs identical knowledge in the comprehension and production directions, asymmetries between comprehension and production do arise in the realistic prototype effect (Section 6.5). And although the more comprehensive model of Figure 1 includes levels that are non-adjacent and therefore seems to disallow nonlocal interactions, it can achieve realistic effects of interactivity across multiple levels, because activity spreads simultaneously top-down and bottom-up (as in the TRACE model; see Section 5.11); an example of this in Section 5.7 and Section 6.4 is the interactive effect of the “later” articulation on the “earlier” mapping from SF to AudF in production.

On the downside, the model cannot really represent more than one segment yet, and we have not attempted to supply the networks with time-varying input at the auditory or surface level. As a result, no phonological structure beyond single categories can be represented yet in the distributed versions of the network, and interesting issues involving time-varying perception or production, such as contextual cue weighting, dynamic sensorimotor knowledge, or coarticulation, could not be studied yet. Once these sequence restrictions are overcome at all levels of representation, important questions that can be answered are whether anything similar to the within-level restrictions of Figure 1 emerges in these networks, and whether anything emerges that is similar to the many hierarchical structures that have been proposed in the literature. Such issues point toward a large-scale programme for future research.

8

CONCLUSION

The BiPhon-NN model is seen to handle some phenomena that psycholinguists and speech researchers have found in the lab and that have never been modelled within a single framework before. Also, the BiPhon-NN model is biologically one step more plausible than an OT model. One of the main missing areas involves strictly phonological phenomena, which will require the model to come to represent sequential or hierarchical structures at the level of the phonological surface form.

REFERENCES

- David H. ACKLEY, Geoffrey E. HINTON, and Terrence J. SEJNOWSKI (1985), A learning algorithm for Boltzmann machines, *Cognitive Science*, 9:147–169.
- Diana APOUSSIDOU (2007), *The learnability of metrical phonology*, Ph.D. thesis, University of Amsterdam.
- Titia BENDERS (2013), *Nature's distributional-learning experiment: infants' input, infants' perception, and computational modeling*, Ph.D. thesis, University of Amsterdam.
- Iris BERENT, Tracy LENNERTZ, Paul SMOLENSKY, and Vered VAKNIN-NUSBAUM (2009), Listeners' knowledge of phonological universals: evidence from nasal clusters, *Phonology*, 26:75–108.
- Catherine BEST (1995), A direct realist view of cross-language speech perception, in Winifred STRANGE, editor, *Speech perception and linguistic experience: theoretical and methodological issues*, pp. 171–203, York Press, Baltimore.
- Paul BOERSMA (1997), How we learn variation, optionality, and probability, *Proceedings of the Institute of Phonetic Sciences (University of Amsterdam)*, 21:43–58.
- Paul BOERSMA (1998), *Functional phonology: formalizing the interactions between articulatory and perceptual drives*, Ph.D. thesis, University of Amsterdam.
- Paul BOERSMA (2000), The OCP in the perception grammar, Rutgers Optimality Archive 435, <http://roa.rutgers.edu>.
- Paul BOERSMA (2006), Prototypicality judgments as inverted perception, in Gisbert FANSELOW, Caroline FÉRY, Ralf VOGEL, and Matthias SCHLESEWSKY, editors, *Gradience in grammar: generative perspectives*, pp. 167–184, Oxford University Press, Oxford.
- Paul BOERSMA (2007), Some listener-oriented accounts of *h*-aspire in French, *Lingua*, 117:1989–2054.
- Paul BOERSMA (2009), Cue constraints and their interactions in phonological perception and production, in Paul BOERSMA and Silke HAMANN, editors, *Phonology in perception*, pp. 55–110, Mouton De Gruyter, Berlin.
- Paul BOERSMA (2011), A programme for bidirectional phonology and phonetics and their acquisition and evolution, in Anton BENZ and Jason MATTAUSCH, editors, *Bidirectional Optimality Theory*, pp. 33–72, John Benjamins, Amsterdam.
- Paul BOERSMA (2012), Modelling phonological category learning, in Abigail C. COHN, Cécile FOUGERON, and Marie K. HUFFMAN, editors, *The Oxford handbook of laboratory phonology*, pp. 207–218, Oxford University Press, New York.

Paul BOERSMA, Paola ESCUDERO, and Rachel HAYES (2003), Learning abstract phonological from auditory phonetic categories: an integrated model for the acquisition of language-specific sound categories, in Maria-Josep SOLÉ, Daniel RECASENS, and Joaquin ROMERO, editors, *Proceedings of the 15th International Congress of Phonetic Sciences, Barcelona*, pp. 1013–1016, Futurgraphic, Barcelona.

Paul BOERSMA and Silke HAMANN (2008), The evolution of auditory dispersion in bidirectional constraint grammars, *Phonology*, 25:217–270.

Paul BOERSMA and Silke HAMANN (2009), Loanword adaptation as first-language phonological perception, in Andrea CALABRESE and W. Leo WETZELS, editors, *Loanword phonology*, pp. 11–58, John Benjamins, Amsterdam.

Paul BOERSMA and Bruce HAYES (2001), Empirical tests of the Gradual Learning Algorithm, *Linguistic Inquiry*, 32:45–86.

Paul BOERSMA and Jan-Willem VAN LEUSSEN (2017), Efficient evaluation and learning in multi-level parallel constraint grammars, *Linguistic Inquiry*, 48:349–388.

Ludwig BOLTZMANN (1868), Studien über das Gleichgewicht der lebendigen Kraft zwischen bewegten materiellen Punkten, *Wiener Berichte*, 58:517–560.

György BUZSÁKI and Kenji MIZUSEKI (2014), The log-dynamic brain: how skewed distributions affect network operations, *Nature Reviews Neuroscience*, 15:264–278.

Kateřina CHLÁDKOVÁ (2014), *Finding phonological features in perception*, Ph.D. thesis, University of Amsterdam.

Anne CUTLER, Jacques MEHLER, Dennis NORRIS, and Juan SEGUI (1987), Phoneme identification and the lexicon, *Cognitive Psychology*, 19:141–177.

Bart DE BOER (1999), *Self-organisation in vowel systems*, Ph.D. thesis, Vrije Universiteit Brussel.

Willem DE GROOT (1931), Phonologie und Phonetik als Funktionswissenschaften, *Travaux du Cercle Linguistique de Prague*, 4:146–147.

Paola ESCUDERO and Paul BOERSMA (2004), Bridging the gap between L2 speech perception research and phonological theory, *Studies in Second Language Acquisition*, 26:551–585.

Edward FLEMMING (1995/2002), *Auditory representations in phonology*, Ph.D. thesis, University of California at Los Angeles, published in 2002 by Routledge (New York/London).

Carol A. FOWLER (1986), An event approach to the study of speech perception from a direct-realist perspective, *Journal of Phonetics*, 14:3–28.

- William F. GANONG III (1980), Phonetic categorization in auditory word perception, *Journal of Experimental Psychology: Human Perception and Performance*, 6:110–125.
- Stephen D. GOLDINGER (1996), Words and voices: episodic traces in spoken word identification and recognition memory, *Journal of Experimental Psychology: Learning, Memory and Cognition*, 5:1166–1183.
- Stephen GROSSBERG (1969), Embedding fields: a theory of learning with physiological implications, *Journal of Mathematical Psychology*, 6:209–239.
- Stephen GROSSBERG (1976), Adaptive pattern classification and universal recoding: I. Parallel development and coding of neural feature detectors, *Biological Cybernetics*, 23:121–134.
- Stephen GROSSBERG (1987), Competitive learning: from interactive activation to adaptive resonance, *Cognitive Science*, 11:23–63.
- Frank H. GUENTHER and Marin N. GJAJA (1996), The perceptual magnet effect as an emergent property of neural map formation, *Journal of the Acoustical Society of America*, 100:1111–1121.
- Richard H.R. HAHNLOSER, Rahul SARPESHKAR, Misha A. MAHOWALD, Rodney J. DOUGLAS, and H. Sebastian SEUNG (2000), Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit, *Nature*, 405:947–951.
- Mark HALE and Charles REISS (2000), “Substance abuse” and “dysfunctionalism”: current trends in phonology, *Linguistic Inquiry*, 31:157–169.
- Donald O. HEBB (1949), *The organization of behavior*, Wiley, New York.
- John HOPFIELD (1982), Neural networks and physical systems with emergent collective computational abilities, *Proceedings of the National Academy of Sciences of the United States of America*, 79:2554–2558.
- Keith JOHNSON, Edward FLEMMING, and Richard WRIGHT (1993), The hyperspace effect: phonetic targets are hyperarticulated, *Language*, 69:505–528.
- Paul KIPARSKY (1982), From cyclic phonology to lexical phonology, in Harry VAN DER HULST and Norval SMITH, editors, *The structure of phonological representations*, volume I, pp. 131–175, Foris, Dordrecht.
- Robert KIRCHNER (1998/2001), *An effort-based approach to consonant lenition*, Ph.D. thesis, University of California at Los Angeles, published in 2001 by Routledge (New York/London).
- Teuvo KOHONEN (1984), *Self-organization and associative memory*, Springer, Berlin.
- John K. KRUSCHKE (1992), ALCOVE: An exemplar-based connectionist model of category learning, *Psychological Review*, 99:22–44.

Patricia K. KUHL (1991), Human adults and human infants show a “perceptual magnetic effect” for the prototypes of speech categories, monkeys do not, *Perception and Psychophysics*, 50:93–107.

Pierre-Simon LAPLACE (1812), *Théorie analytique des probabilités*, Veuve Courcier, Paris.

Willem LEVELT, Ardi ROELOFS, and Antje MEYER (1999), A theory of lexical access in speech production, *Behavioral and Brain Sciences*, 22:1–75.

Alvin LIBERMAN and Ignatius MATTINGLY (1985), The motor theory of speech perception revised, *Cognition*, 21:1–36.

Alvin M. LIBERMAN, Katherine Safford HARRIS, Howard S. HOFFMAN, and Belder C. GRIFFITH (1957), The discrimination of speech sounds within and across phoneme boundaries, *Journal of Experimental Psychology*, 54:358–368.

Johan LILJENCRANTS and Björn LINDBLOM (1972), Numerical simulation of vowel quality systems: the role of perceptual contrast, *Language*, 48:839–862.

Björn LINDBLOM (1986), Phonetic universals in vowel systems, in John OHALA and Jeri JAEGER, editors, *Experimental phonology*, pp. 13–44, Academic Press, Orlando.

Rafael LORENTE DE NÓ (1938), Cerebral cortex: architecture, intracortical connections, motor projections, in J.F. FULTON, editor, *Physiology of the nervous system*, pp. 291–327, Oxford University Press, London.

R. Duncan LUCE (1959), *Individual choice behavior: a theoretical analysis*, Wiley, New York.

André MARTINET (1960), *Éléments de linguistique générale*, Armand Colin, Paris.

Jessica MAYE, Daniel J. WEISS, and Richard N. ASLIN (2008), Statistical phonetic learning in infants: facilitation and feature generalization, *Developmental Science*, 11:122–134.

James L. MCCLELLAND and Jeffrey L. ELMAN (1986), The TRACE model of speech perception, *Cognitive Psychology*, 18:1–86.

Bob McMURRAY, Jessica S. HORST, Joseph C. TOSCANO, and Larissa K. SAMUELSON (2009), Towards an integration of connectionist learning and dynamical systems processing: case studies in speech and lexical development, in John P. SPENCER, Michael S.C. THOMAS, and James L. MCCLELLAND, editors, *Toward a unified theory of development: connectionism and dynamic systems theory re-considered*, pp. 218–249, Oxford University Press, New York.

Dennis NORRIS (1994), Shortlist: a connectionist model of continuous speech recognition, *Cognition*, 52:189–234.

Dennis NORRIS, James M. MCQUEEN, and Anne CUTLER (2000), Merging information in speech recognition: feedback is never necessary, *Behavioral and Brain Sciences*, 23:299–370.

- Erkki OJA (1982), A simplified neuron model as a principal component analyzer, *Journal of Mathematical Biology*, 15:267–273.
- Pierre-Yves OUDEYER (2006), *Self-organization in the evolution of speech*, Oxford University Press, Oxford.
- Randall C. O'REILLY (1996), *The Leabra model of neural interactions and learning in the neocortex*, Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA.
- Jaye PADGETT (2003), Contrast and post-velar fronting in Russian, *Natural Language and Linguistic Theory*, 21:39–87.
- Paul PASSY (1890), *Étude sur les changements phonétiques et leur caractères généraux*, Firmin-Didot, Paris.
- Joe PATER (2004), Bridging the gap between receptive and productive development with minimally violable constraints, in René KAGER, Joe PATER, and Wim ZONNEVELD, editors, *Constraints in phonological acquisition*, pp. 219–244, Cambridge University Press, Cambridge.
- Donald H. PERKEL and Theodore H. BULLOCK (1969), Neural coding, *Neurosciences Research Symposium Summaries*, 3:405–527.
- Janet PIERREHUMBERT (2001), Exemplar dynamics: word frequency, lenition and contrast, in Joan BYBEE and Paul HOPPER, editors, *Frequency and the emergence of linguistic structure*, pp. 137–157, John Benjamins, Amsterdam.
- Alan PRINCE and Paul SMOLENSKY (1993/2004), Optimality Theory: constraint interaction in generative grammar, Technical Report TR-2, Rutgers University Center for Cognitive Science, published in 2004 by Blackwell (Malden, MA/Oxford).
- Nathaniel ROCHESTER, John H. HOLLAND, Luther H. HAIBT, and William L. DUDA (1956), Tests on a cell assembly theory of the action of the brain, using a large digital computer, *IRE Transactions on Information Theory*, 2:80–93.
- Frank ROSENBLATT (1958), The perceptron: a probabilistic model for information storage and organization in the brain, *Psychological Review*, 65:386–408.
- David E. RUMELHART and David ZIPSER (1985), Feature discovery by competitive learning, *Cognitive Science*, 9:75–112.
- Jean-Luc SCHWARTZ, Louis-Jean BOË, Nathalie VALLÉE, and Christian ABRY (1997), The dispersion–focalization theory of vowel systems, *Journal of Phonetics*, 25:255–286.
- Klaas SEINHORST (2012), *The evolution of auditory dispersion in symmetric neural nets*, Master's thesis, University of Amsterdam.
- Klaas SEINHORST, Paul BOERSMA, and Silke HAMANN (2019), Iterated distributional and lexicon-driven learning in a symmetric neural network explains the emergence of features and dispersion, in Sasha CALHOUN, Paola

- ESCUADERO, Marija TABAIN, and Paul WARREN, editors, *Proceedings of the 19th International Congress of Phonetic Sciences, Melbourne*, pp. 1135–1138, Australasian Speech Science and Technology Association Inc., Canberra.
- Paul SMOLENSKY (1996), On the comprehension/production dilemma in child language, *Linguistic Inquiry*, 27:720–731.
- Louis TEN BOSCH (1991), *On the structure of vowel systems: aspects of an extended vowel model using effort and contrast*, Ph.D. thesis, University of Amsterdam.
- Sophie TER SCHURE, Caroline JUNGE, and Paul BOERSMA (2016), Semantics guide infants' vowel learning: computational and experimental evidence, *Infant Behavior and Development*, 43:44–57.
- Bruce TESAR (1997), An iterative strategy for learning metrical stress in Optimality Theory, in Elizabeth HUGHES, Mary HUGHES, and Annabel GREENHILL, editors, *Proceedings of the 21st Annual Boston University Conference on Language Development*, pp. 615–626, Cascadilla, Somerville, MA.
- Bruce TESAR and Paul SMOLENSKY (1998), Learnability in Optimality Theory, *Linguistic Inquiry*, 29:229–268.
- Bruce TESAR and Paul SMOLENSKY (2000), *Learnability in Optimality Theory*, MIT Press, Cambridge, MA.
- Nikolai TRUBETZKOY (1939), *Grundzüge der Phonologie*, Travaux du Cercle Linguistique de Prague 7.
- Nicolaas VAN WIJK (1936), Positieve en negatieve opmerkingen over de definitie van het phoneem, *Nieuwe Taalgids*, 30:311–326.
- Georg VON DER GABELNTZ (1901), *Die Sprachwissenschaft: ihre Aufgaben, Methoden und bisherigen Ergebnisse*, Tauchnitz, Leipzig.
- Karin WANROOIJ (2015), *Distributional learning of vowel categories in infants and adults*, Ph.D. thesis, University of Amsterdam.
- Richard M. WARREN and Roslyn P. WARREN (1970), Auditory illusions and confusions, *Scientific American*, 223:30–37.
- Andrew WEDEL (2004), *Self-organization and categorical behavior in phonology*, Ph.D. thesis, University of California at Santa Cruz.
- Andrew WEDEL (2006), Exemplar models, evolution and language change, *The Linguistic Review*, 23:247–274.
- Andrew WEDEL (2007), Feedback and regularity in the lexicon, *Phonology*, 24:147–185.
- David WEENINK (2006), *Speaker-adaptive vowel identification*, Ph.D. thesis, University of Amsterdam.
- Bernard WIDROW and Marcian E. HOFF (1960), Adaptive switching circuits, in *IRE WESCON Convention Record, part 4*, pp. 96–104, IRE, New York, reprinted in James E. ANDERSON and Edward ROSENFELD, editors (1988), *Neurocomputing*, pp. 126–134, MIT Press, Cambridge, MA.

Neural network models for phonology and phonetics

Paul Boersma

Ⓘ 0000-0003-4328-3840
paul.boersma@uva.nl

Klaas Seinhorst

Ⓘ 0000-0002-9716-0742
seinhorst@uva.nl

Amsterdam Center
for Language and Communication
University of Amsterdam
Amsterdam, The Netherlands

Titia Benders

Ⓘ 0000-0003-0143-2182
titia.benders@mq.edu.au

Department of Linguistics
and Centre for Language Sciences
Macquarie University
Sydney, Australia

Paul Boersma, Titia Benders, and Klaas Seinhorst (2020), *Neural network models for phonology and phonetics*, *Journal of Language Modelling*, 8(1):103–177

Ⓓ <https://dx.doi.org/10.15398/jlm.v8i1.224>

This work is licensed under the *Creative Commons Attribution 4.0 Public License*.

ⒸⒻ <http://creativecommons.org/licenses/by/4.0/>

Computing and classifying reduplication with 2-way finite-state transducers

Hossep Dolatian and Jeffrey Heinz
Stony Brook University

ABSTRACT

This article describes a novel approach to the computational modeling of reduplication. Reduplication is often treated as a stumbling block within finite-state treatments of morphology because they cannot adequately capture the productivity of *unbounded* copying (total reduplication) and because they cannot describe *bounded* copying (partial reduplication) without a large increase in the number of states. We provide a comprehensive typology of reduplicative processes and show that an understudied type of finite-state machine, 2-way deterministic finite-state transducers (2-way D-FSTs), captures virtually all of them. Furthermore, the 2-way D-FSTs have few states, are in practice easy to design and debug, and are linguistically motivated in terms of the transducer's origin semantics or segment alignment. Most of these processes, and their corresponding 2-way D-FSTs, are available in an online database of reduplication (RedTyp). We classify these 2-way D-FSTs according to the concatenation of known subclasses of regular relations and show that the majority fall into the Concatenated Output Strictly Local (C-OSL) class. Other cases require higher subclasses but are still definable by 2-way D-FSTs.

Keywords:
reduplication,
2-way finite state
transducer, finite
state morphology

INTRODUCTION

Reduplication is a cross-linguistically common word-formation process involving *copying*. Given a word, reduplication can copy either a bounded (1a) or unbounded (1b) number of segments. The symbol \sim marks the boundary between the two copies.

- (1) a. **Partial Reduplication** *Agta* (Moravcsik 1978, 311)
 takki \rightarrow tak \sim takki ‘leg’ \rightarrow ‘legs’
- b. **Total Reduplication** *Indonesian* (Cohn 1989, 308)
 buku \rightarrow buku \sim buku ‘book’ \rightarrow ‘books’

Reduplication is used in the majority of the world’s languages, and total reduplication is more common than partial reduplication. The World Atlas of Language Structures (WALS) database documents that 278 out of 368 (75%) languages have total and partial reduplication (Rubino 2013). 35 additional languages (10%) use only total, not partial, reduplication. The 55 (15%) remaining languages do not have productive reduplication, but this figure is debatable.¹ Therefore, developing analyzable and efficient computational models of reduplication is important.

Although reduplication is well-studied, it is a computationally challenging process (Sproat 1992). In computational linguistics, most morphological and phonological processes can be analyzed with finite-state calculus in terms of rational languages and transductions (Kaplan and Kay 1994; Beesley and Karttunen 2003). However, reduplicative processes cannot be easily modeled with the same finite-state systems. For total reduplication, this is because those finite-state systems cannot express unbounded copying in the first place (Culy 1985). As for partial reduplication, those finite-state systems are often described as unwieldy because of the state explosion caused by partial reduplication (Roark and Sproat 2007, 54). Section 2 of this article explains why reduplication is computationally challenging while reviewing previous computational approaches to this linguistic phenomenon.

¹ Most of the exceptional languages are Indo-European, but some argue that these languages still use total reduplication (Ghosheshi *et al.* 2004; Stolz *et al.* 2011).

In this context, the primary contribution of this article is to show that a specific understudied type of finite-state technology *can* account for virtually all reduplicative processes. This type of transducer is known as a 2-way Finite-State Transducer or 2-way FST (Savitch 1982; Engelfriet and Hoogetboom 2001; Filiot and Reynier 2016).² In theoretical computer science, 2-way FSTs are known to be able to model unbounded copying (Engelfriet and Hoogetboom 2001). To our knowledge, we are the first to apply 2-way FSTs to computational linguistics.³

The FSTs used in most of computational linguistics are more accurately called *1-way* FSTs. They can only read the input once in one direction. 2-way FSTs are more expressive because the read head can move back and forth on the input tape. On the other hand, the write head can only move forward on the output tape. For this reason, they are less expressive than Turing machines. It is this back-and-forth movement of the read head that allows 2-way FSTs to adequately model reduplication without the difficulties faced by 1-way FSTs. This article introduces *deterministic* 2-way finite-state transducers (2-way D-FSTs) in Section 3, along with their formal definition (Section 3.1), illustrative examples of reduplication (Section 3.2), and their computational properties (Section 3.3).

The fact that the 2-way FSTs used in this article are deterministic is significant. It is well known that deterministic 1-way FSTs are less expressive than non-deterministic 1-way FSTs (Elgot and Mezei 1965; Schützenberger 1975; Choffrut 1977; Mohri 1997; Heinz and Lai 2013). Similarly, 2-way D-FSTs are less expressive than non-deterministic 2-way FSTs (Culik and Karhumäki 1986). Consequently, the empirical result that reduplication can be modeled with deterministic 2-way FSTs is in line with work which shows that various phonological and morphological processes can be described with deterministic finite-state technology (Chandlee *et al.* 2012; Gainor *et al.* 2012; Chandlee and Heinz

²This article builds off of our previous work on using 2-way FSTs for reduplication (Dolatian and Heinz 2018a,b, 2019a,b).

³2-way finite-state *acceptors* (2-way FSAs) have been used to model non-concatenative Semitic morphology (Narayanan and Hashem 1993) and to parse dependency grammars (Nelimarkka *et al.* 1984).

2012; Heinz and Lai 2013; Chandlee 2014; Luo 2017; Payne 2014, 2017).

In the later part of this article, we provide a comprehensive cross-linguistic survey of reduplicative processes based on earlier typological studies (Moravcsik 1978; Rubino 2005; Inkelas and Downing 2015a), with reduplication defined as an input-to-output function (McCarthy and Prince 1995). This survey is documented in a database we constructed, which we call The RedTyp database. It contains entries for 138 reduplicative processes from 91 languages and a 2-way D-FST for each entry. Aspects of this survey are presented in Sections 3–4, and discussed in detail in Section 6.

In Section 4, we compare 2-way D-FSTs to 1-way FSTs in terms of empirical coverage (Section 4.1), practical utility (Section 4.2), and linguistic motivation (Section 4.3). We argue that 2-way D-FSTs are linguistically motivated in that they capture the correspondence relations underlying the base and the reduplicant in a linguistically natural way. These correspondence relations are couched in terms of origin semantics (Bojańczyk 2014). We use origin semantics as a diagnostic for the strong generative capacity of reduplicative functions. (We do not claim that origin semantics matches linguistic intuitions exactly in every case, but rather that it approximately does so in many instructive cases.)

The final contribution of this article is an attempt to classify reduplicative processes according to *subclasses* of 2-way D-FSTs in Section 6. The first result we already mentioned: the full typology of reduplicative processes can be modeled with *deterministic* 2-way D-FSTs. These subclasses are defined in terms of concatenations of subclasses of 1-way FSTs. Our next result is that approximately three-quarters of the typology is expressible with the concatenation of Output Strictly Local (OSL) functions (Chandlee *et al.* 2015). The remainder of the typology is expressible with the concatenation of sequential functions, with some arguably requiring sweeping transducers or unrestricted 2-way D-FSTs.

We review these contributions and conclude in Section 7.

BACKGROUND ON COMPUTATION OF REDUPLICATION

2

Within computational linguistics, reduplication has been a challenging process to model (Culy 1985; Sproat 1992; Roark and Sproat 2007; Hulden 2009a; Chandlee 2014, 2017). Finite-state technology, as currently practiced, cannot adequately and elegantly describe many cases of productive reduplication, especially *unbounded* total reduplication. There are three kinds of issues: empirical coverage, practical utility, and matching the intensional description of reduplication. We discuss these challenges in Section 2.1. In response to these problems, some have proposed finite-state approximations for reduplication (Section 2.2) or developing more expressive systems just for total reduplication (Section 2.3). The latter approach however implies that total reduplication is ontologically different from partial reduplication, and thus should be computed differently. In Section 2.4, we discuss this implication and show that the evidence for it is inconclusive. We summarize in Section 2.5.

Why reduplication is challenging

2.1

Reduplication is challenging because segmental copying entails multiple crossing dependencies between the two copies. When the number of copied segments (and thus the number of crossing dependencies) is bound to some maximum number n , the outcome is partial reduplication. When there is no bound, the outcome is total reduplication.

Partial reduplication *can* be modeled with 1-way FSTs (Roark and Sproat 2007; Chandlee and Heinz 2012). However, as we explain in more detail in Section 4.3, these machines are understood as memorizing all finitely-many possible forms of the partial reduplicant. Consequently, the transducers suffer from an explosion of states and become unwieldy. For example, in a language with a medium-sized phonemic inventory of 22 consonants and 5 vowels, partial reduplication with a CVCV template would require at least $22 + 22 \times 5 + 22 \times 5 \times 22 = 2552$ states to memorize the first C (22 states), the first V (22×5 states), and the second C ($22 \times 5 \times 22$ states). 1-way FSTs likewise arguably

do not match the intensional description of reduplication as a *copying* process because the FSTs simply memorize all possible reduplicants in the language (Roark and Sproat 2007, 54). This is discussed in detail in Section 4.3.

On the other hand, total reduplication cannot be modeled at all with 1-way FSTs (Culy 1985). This inability is due to the fact that the output language of total reduplication is not a regular language. Rather, the copying process of total reduplication can create output languages that are identical to the non-context-free $L_{ww} = \{ww \mid w \in \Sigma^*\}$ (Hopcroft and Ullman 1969). Thus the copying function $w \mapsto ww$ (sometimes called the squaring function) is beyond the expressivity of 1-way FSTs. In fact, virtually all attested morphological processes can be described with 1-way finite-state acceptors and transducers, *except* for total reduplication (Langendoen 1981; Gazdar and Pullum 1985; Roark and Sproat 2007). In response to this problem, computational morphologists have often resorted to using either finite-state approximations (Section 2.2) or non-finite-state tools (Section 2.3).

2.2

Finite-state approximations

The literature on finite-state morphology contains many finite-state approximations to reduplication (Walther 2000; Beesley and Karttunen 2003; Cohen-Sygal and Wintner 2006; Hulden and Bischoff 2009). Roark and Sproat (2007, 57) and Cohen-Sygal and Wintner (2006, 52) provide reviews. In general, finite-state approximations are designed to lessen the burden for the developer in designing reduplication rules. They introduce new operations or tools over 1-way FSTs but they do *not* increase their expressivity. In other words, they are designed to improve on practical utility but they don't improve on empirical coverage or intensional description.

Here we briefly review the two main sets of approaches with details following. One set of approaches checks for identity between the two copies (Cohen-Sygal and Wintner 2006; Hulden and Bischoff 2009). Another set of approaches essentially 'postpones' reduplication to a run-time task (Walther 2000; Beesley and Karttunen 2000, 2003). Both try to reduce state complexity either by making a trade-off with time complexity, by implementing reduplication with a unique 1-way transducer for each morpheme in the finite lexicon, or both.

Cohen-Sygal and Wintner (2006) augment 1-way FSAs with finitely many registers (FSRA). These registers keep track of a *bounded* number of segments previously seen in the input. In order to model the total reduplication of a *given* word like *buku* → *buku~buku* (1b), the FSRA has at least as many registers as segments in the base *buku*: four. The registers check that the string *buku~buku* can be broken down into identical copies. Similarly, Hulden and Bischoff (2009) design the EQ function within the foma system (Hulden 2009b) which checks if a string is divided into two identical copies.

As for run-time procedures, these systems are designed on an input by input basis. Given some input word, they create a reduplication FST for it *on the fly*. Given an input *buku*, the compile-replace operation (Beesley and Karttunen 2000, 2003) creates an intermediate representation $\{buku\}^2$ via a 1-way FST. This intermediate representation is then interpreted as a regular expression in run-time, i.e. it is *compiled*. By compiling this regular expression, the word *bukubuku* is outputted.

Within the framework of One-Level Phonology (Bird and Ellison 1994), Walther (2000) models reduplication by representing a potentially-reduplicated morpheme like *buku* as an FSA with augmentations on the types of transition arcs: *content*, *repeat*, and *skip* arcs. These transition arcs turn a linear string *buku* into a multi-linear structure where the read head can ‘move’ around the string. This enriched representation is then intersected with a reduplication FSA that is designed to ‘move’ around this enriched representation and generate *buku~buku*.⁴ Ideally, these operations should be applied in run-time. Otherwise, if these operations are applied to the entire lexicon and stored as a single FST, they then suffer from the state explosion that they were designed to avoid.

As for total reduplication, all four of the above modifications are *approximations*. This is because they impose various restrictions which contradict the linguistic generalization that total reduplication is independent of string length. Most notably, all four approximations permit only a closed finite set of input strings to undergo total reduplication. This restriction fundamentally alters total reduplication from a process

⁴ Walther (2000) does not give a formal analysis. But, we think that these augmented transition arcs are similar to 2-way FSAs and are an independently developed implementation for Precedence-Based Phonology (Raimy 2000).

which in principle applies to infinitely many words to a process which applies to only finitely many. Such approximations thus fall short of capturing how total reduplication is used as a productive process in natural language.

As for partial reduplication, all of the above four approaches have the same expressivity and are able to capture the linguistic generalization that partial reduplication is independent of string length. However, although they are designed to avoid state explosion by one means or another, they can still be said to memorize the partial reduplicant as opposed to copying it (see Section 4.3). In this way they do not intensionally capture the linguistic generalization of copying.

2.3

Extending formal power

Because of the difficulty in modeling reduplication with finite-state machinery, various augmentations and expansions of context-free grammars have been proposed to handle L_{ww} and reduplication. An early augmentation is Reduplication Context-Free Grammars (Manaster-Ramer 1986; Savitch 1989) designed to handle context-free languages and reduplication by using queues instead of stacks. A more recent augmentation is Multiple-Context Free Grammars (MCFGs) which can model L_{ww} (Seki *et al.* 1991, 1993). MCFGs have been used to model reduplication (Albro 2000, 2005). As an extension of MCFGs, Parallel MCFGs have been used to model reduplication and syntactic copying (Kobele 2006; Clark and Yoshinaka 2012, 2014; Clark 2017).⁵ Crysmann (2017) explores the use of HPSG to model total reduplication.

These technologies have had considerably less attention within mainstream computational morphology than finite-state approximations. One shortcoming of these approaches is that they model formal languages, not transformations. They accept well-formed reduplicated words ww , but they do not generate a reduplicated word ww given some input w . Thus, they do not model the squaring function $w \mapsto ww$.

⁵ Kobele (2006) shows that syntactic copying can generate languages of the form a^{2^n} , i.e., exponential copying. This isn't attested in morphological copying. Note the string a^{2^n} is generated as the yield language of a tree transduction over a derivation tree.

The previous sections showed that more expressive mechanisms are needed to model reduplication. Conceptually, the use of more powerful computational formalisms implies that reduplication is ‘different’ from the rest of morpho-phonology which can be modeled using 1-way FSTs (Roark and Sproat 2007, 60). This is especially the case for total reduplication which cannot be exactly modeled with 1-way FSTs, whereas partial reduplication can. This difference has caused debates over whether both types of reduplication *should* be computed with the same formalism or not. However, this debate is inconclusive.

On one hand, Chandlee (2017) suggests that the inadequacy of 1-way FSTs for total reduplication is evidence for total reduplication being ontologically different from partial reduplication. There is some empirical support for this argument. Prosodically, total reduplication resembles more ‘syntactic’ processes like compounding more often than partial reduplication (Downing 2006). The two copies in total reduplication can be stressed separately or have separate tonal contours (Downing 2003).

On the other hand, partial and total reduplication are closely related processes. Typologically, if a language has partial reduplication, then it almost always has total reduplication too (Rubino 2013). Diachronically, both types of reduplication are typically related to each other, but not always (Hurch and Mattes 2009). And in linguistic theory, both are modeled with the same tools (Steriade 1988; Raimy 2000; Inkelas and Zoll 2005; McCarthy *et al.* 2012).

Psycholinguistic work could shed more light on the issue of total reduplication vis a vis partial reduplication. Sadly, there is little to no work on the psycholinguistic processing of reduplication. To our knowledge, existing work focuses on partial reduplication, not total reduplication (Ohala *et al.* 1986; Waksler 1999).

Learnability is another factor which could tease apart these processes. It is an open question whether both partial and total reduplication can be learned in the same way with the same mechanism. In terms of stringsets, the formal language of totally reduplicated words *ww* can be learned with distributional methods for MCFGs (Clark and Yoshinaka 2012, 2014, 2016). There is also a substantial body of work

in cognitive science and connectionism on how to learn reduplicated words (Marcus *et al.* 1999; Berent *et al.* 2014, 2016, 2017; Andan *et al.* 2018; Alhama 2017; Alhama and Zuidema 2019). Here, the task is learning words which have repeated substrings (ABB or ABA) where A and B are syllables.

In contrast, there is little to no work on learning reduplication as a function ($w \rightarrow ww$), whether in machine learning or grammatical inference. To our knowledge, the only algorithm designed specifically for learning reduplication is Nevins (2004) in the principles-and-parameters tradition. There is some recent work on using neural networks to learn copying (Gu *et al.* 2016; Prickett *et al.* 2018; Wilson 2019; Nelson *et al.* 2020). We speculate that one reason for the dearth of learning results is due to the challenges outlined above for finding natural computational models for reduplication.

2.5 *Summary and consequences*

All in all, current finite-state treatments of reduplication have issues regarding their empirical coverage (total reduplication's productivity), practical utility (state space explosion), and intensional descriptions (copying vs. remembering). The present study uses a computational formalism which does not suffer from these three problems: two-way finite-state transducers (2-way FSTs).

3 2-WAY FINITE-STATE TRANSDUCERS: DEFINITION AND APPLICATION TO REDUPLICATION

1-way FSTs read the input *once* from left to right. Most applications use non-deterministic 1-way FSTs (Roark and Sproat 2007), though deterministic 1-way FSTs are largely sufficient (Chandlee 2017). (For all inputs, deterministic FSTs have at most one path through the transducer, whereas non-deterministic ones may have more than one path.) 2-way FSTs can move back and forth on the input (Rabin and Scott 1959; Hopcroft and Ullman 1969). This ability makes them more expressive than 1-way FSTs (Savitch 1982; Engelfriet and Hoogeboom 2001).

It is useful to imagine a 2-way FST as a machine operating on an input tape and writing to an output tape. The symbols on the input tape are drawn from an alphabet Σ and the symbols written to the output tape are drawn from an alphabet Γ . For an input string $w = \sigma_1 \dots \sigma_n$, the initial configuration is that the FST is in some internal state q_0 , its read head on σ_1 , and its write head at the beginning of an empty output tape. After the FST reads the symbol under the read head, three things occur:

- The internal state of the FST may change.
- The FST writes some string, possibly empty, to the output tape.
- The read head moves in one of three ways: moves to the left (-1), moves to the right ($+1$), or stays (0).

This process repeats until the read head “falls off” one of the edges of the input tape. If for some input string w , the FST falls off the right edge of the input tape when the FST is in an accepting state after writing u on the output tape, we say the FST transduces, transforms, or maps, w to u . If for some input string w , the FST falls off the left edge, falls off the right edge while in a non-accepting state, or never falls off either edge, then the FST is undefined at w . The write head cannot move back along the output tape. It can only advance as strings are written.

We formalize the definition and behavior of 2-way FSTs in Section 3.1. They are illustrated for reduplication in Section 3.2. We then describe their generative capacity and computational complexity (Section 3.3).

Preliminaries and formal definition

3.1

Given a finite alphabet Σ , the set of all possible strings of finite length built from Σ is Σ^* . The empty string is represented by λ . The length of a string w is $|w|$, so $|\lambda| = 0$. For the given strings w_1, w_2 , their concatenation is $w_1 w_2$. Below is a formalization of deterministic 2-way FSTs based on Filiot and Reynier (2016) and Shallit (2008). We adopt the convention that inputs to a 2-way D-FST are flanked with the start (\bowtie) and end (\bowtie) boundaries. This larger alphabet is denoted by Σ_{\bowtie} .

(2) **Definition:** A 2-way D-FST is a six-tuple $(Q, \Sigma_{\times}, \Gamma, q_0, F, \delta)$ where:

- Q is a finite set of states,
- $\Sigma_{\times} = \Sigma \cup \{\times, \kappa\}$ is the input alphabet,
- Γ is the output alphabet,
- $q_0 \in Q$ is the initial state,
- $F \subseteq Q$ is the set of final states,
- $\delta : Q \times \Sigma \rightarrow Q \times \Gamma^* \times D$ is the transition function where the direction $D = \{-1, 0, +1\}$.

A *configuration* of a 2-way D-FST T is an element of $\Sigma_{\times}^* Q \Sigma_{\times}^* \times \Gamma^*$. The meaning of the configuration (wqx, u) is that the input to T is wx and the machine is currently in state q with the read head on the first symbol of x (or has fallen off the right edge of the input tape if $x = \lambda$) and that u is currently written on the output tape.

If the current configuration is $(wqax, u)$ and $\delta(q, a) = (r, v, 0)$ then the next configuration is $(wrax, uv)$, in which case we write $(wqax, u) \rightarrow (wrax, uv)$. If the current configuration is $(wqax, u)$ and $\delta(q, a) = (r, v, +1)$ then the next configuration is $(warx, uv)$. In this case, we write $(wqax, u) \rightarrow (warx, uv)$. If the current configuration is $(waqx, u)$ and $\delta(q, a) = (r, v, -1)$ then the next configuration is $(wrax, uv)$. We write $(waqx, u) \rightarrow (wrax, uv)$. Observe that since δ is a function, there is at most one next configuration.

The transitive closure of \rightarrow is denoted with \rightarrow^+ . Thus, if $c \rightarrow^+ c'$ then there exists a finite sequence of configurations $c_1, c_2 \dots c_n$ with $n > 1$ such that $c = c_1 \rightarrow c_2 \rightarrow \dots \rightarrow c_n = c'$.

Next we define the function f_T that a 2-way D-FST T computes. For each string $w \in \Sigma^*$, $f_T(w) = u \in \Gamma^*$ provided there exists $q_f \in F$ such that $(q_0 \times w \kappa, \lambda) \rightarrow^+ (\times w \times q_f, u)$. If $f_T(w) = u$ then u is unique because the sequence of configurations is determined deterministically.

If the configurations of a 2-way D-FST T halt the computation of T on some input w , then we say T is undefined on w . If the configuration is (qax, u) and $\delta(q, a) = (r, -1, v)$ then the derivation crashes and the transduction $f_T(ax)$ is undefined. Likewise, if the configuration is (wq, u) and $q \notin F$ then the transducer crashes and the transduction f_T is undefined on input w . Another way that f_T may be undefined for some input is if the input causes the transducer to go into an infinite

loop.⁶ This occurs for input $wx \in \Sigma_{\times}^*$ whenever there exist $q \in Q$ and $u, v \in \Gamma^*$ such that $(q_0wx, \lambda) \rightarrow^+ (wqx, u) \rightarrow^+ (wqx, uv)$.

Illustration of two-way transducers for reduplication

3.2

Having established what 2-way D-FSTs are, this section illustrates how they can be used to model reduplication. We provide two examples: total reduplication and partial initial-CVC reduplication. Both examples use deterministic 2-way FSTs.

Some useful terms are ‘passes’ and ‘rewinds’. A pass (rewind) is when a 2-way D-FST moves left-to-right (right-to-left) from some position to another over the input.

Total reduplication is cross-linguistically the most common reduplicative process (Rubino 2005), and it is used in an estimated 85% of the world’s languages (Rubino 2013). We illustrate it with data from Indonesian where total reduplication marks plurality (Cohn 1989).

- | | | | | | | | |
|-----|----|--------|---|---------------|----------|---|-----------|
| (3) | a. | buku | → | buku~buku | ‘book’ | → | ‘books’ |
| | b. | wanita | → | wanita~wanita | ‘woman’ | → | ‘women’ |
| | c. | hak | → | hak~hak | ‘right’ | → | ‘rights’ |
| | d. | kəra | → | kəra~kəra | ‘donkey’ | → | ‘donkeys’ |

Figure 1 shows a 2-way D-FST that captures total reduplication. The boundary symbol \sim is a symbol in the output alphabet Γ , and is not necessary. We include it only for illustration. The 2-way D-FST in Figure 1 operates as follows:

1. **First pass:** It reads the input tape from left to right and outputs the first copy.
2. **Rewind:** When it reaches the end boundary \times , it ‘rewinds’ or goes back to the start of the input tape by moving left until the start boundary \times is reached.
3. **Second pass:** It reads the input tape once more from left to right and outputs the second copy.

⁶In practice, infinite loops are not a problem. It can be checked whether an input leads the 2-way D-FST into an infinite loop during run-time, in which case the computation can be halted.

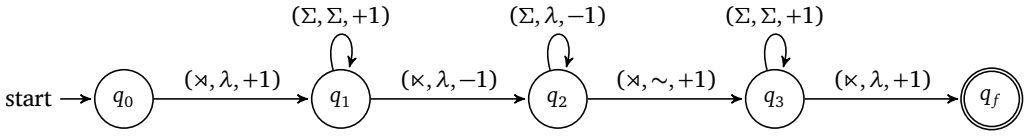


Figure 1: 2-way D-FST for total reduplication

The transition arcs are interpreted as follows. The symbol Σ is a variable representation of any alphabet symbol except for $\{x, \kappa\}$. The arrow from q_1 to itself $(\Sigma, \Sigma, +1)$ means this 2-way D-FST reads a symbol from Σ , writes that same symbol, and advances the read head one step to the right on the input tape.

Table 1 shows an example derivation for $buku \rightarrow buku\sim buku$ using the 2-way D-FST in Figure 1. The derivation shows the step-by-step configurations for the computation. The tuples in Table 1 consist of

Table 1: Derivation of $/buku/ \rightarrow [buku\sim buku]$

Outputting the first copy					
1.	$(q_0 \underline{x} buku \kappa, \lambda,$	N/A)	2.	$(x \underline{q_1} \underline{buku} \kappa, \lambda,$	$q_0 \xrightarrow[\text{+1}]{x:\lambda} q_1)$
3.	$(x \underline{b} \underline{q_1} \underline{u} \underline{k} \kappa, b,$	$q_1 \xrightarrow[\text{+1}]{\Sigma:\Sigma} q_1)$	4.	$(x \underline{bu} \underline{q_1} \underline{k} \underline{u} \kappa, bu,$	$q_1 \xrightarrow[\text{+1}]{\Sigma:\Sigma} q_1)$
5.	$(x \underline{buk} \underline{q_1} \underline{u} \kappa, buk,$	$q_1 \xrightarrow[\text{+1}]{\Sigma:\Sigma} q_1)$	6.	$(x \underline{buku} \underline{q_1} \underline{\kappa}, buku,$	$q_1 \xrightarrow[\text{+1}]{\Sigma:\Sigma} q_1)$
Going back to the start of the tape					
7.	$(x \underline{buk} \underline{q_2} \underline{u} \kappa, buku,$	$q_1 \xrightarrow[-1]{\kappa:\lambda} q_2)$	8.	$(x \underline{bu} \underline{q_2} \underline{k} \underline{u} \kappa, buku,$	$q_2 \xrightarrow[-1]{\Sigma:\lambda} q_2)$
9.	$(x \underline{b} \underline{q_2} \underline{u} \underline{k} \kappa, buku,$	$q_2 \xrightarrow[-1]{\Sigma:\lambda} q_2)$	10.	$(x \underline{q_2} \underline{buku} \kappa, buku,$	$q_2 \xrightarrow[-1]{\Sigma:\lambda} q_2)$
11.	$(q_2 \underline{x} \underline{buku} \kappa, buku,$	$q_2 \xrightarrow[-1]{\Sigma:\lambda} q_2)$			
Outputting the second copy					
12.	$(x \underline{q_3} \underline{buku} \kappa, buku\sim,$	$q_2 \xrightarrow[\text{+1}]{x:\sim} q_3)$	13.	$(x \underline{b} \underline{q_3} \underline{u} \underline{k} \kappa, buku\sim b,$	$q_3 \xrightarrow[\text{+1}]{\Sigma:\Sigma} q_3)$
14.	$(x \underline{bu} \underline{q_3} \underline{k} \underline{u} \kappa, buku\sim bu,$	$q_3 \xrightarrow[\text{+1}]{\Sigma:\Sigma} q_3)$	15.	$(x \underline{buk} \underline{q_3} \underline{u} \kappa, buku\sim buk,$	$q_3 \xrightarrow[\text{+1}]{\Sigma:\Sigma} q_3)$
16.	$(x \underline{buku} \underline{q_3} \underline{\kappa}, buku\sim buku,$	$q_3 \xrightarrow[\text{+1}]{\Sigma:\Sigma} q_3)$	17.	$(x \underline{buku} \underline{\kappa} \underline{q_f}, buku\sim buku,$	$q_3 \xrightarrow[\text{+1}]{\kappa:\kappa} q_f)$

three parts. The first two represent the configuration and the third part shows the transition exercised to reach this configuration from the previous one. The underlined input symbol is what the FST will read next. In the first tuple, there is no transition used (N/A). Transitions in the other tuples are given in the form shown below.

$$\text{input state} \xrightarrow[\text{direction}]{\text{input symbol:output string}} \text{output state}$$

Partial reduplication processes are also very common. A common example is initial-CVC reduplication as in Agta (Moravcsik 1978, 311).

- (4) a. takki → tak~takki ‘leg’ → ‘legs’
 b. uffu → uf~uffu ‘thigh’ → ‘thighs’

The 2-way D-FST in Figure 2 expresses partial initial-CVC reduplication. An example derivation of *takki* → *tak~takki* using our 2-way D-FST is provided in Table 2. For illustrative purposes, we assume that the function is undefined for V-initial inputs.

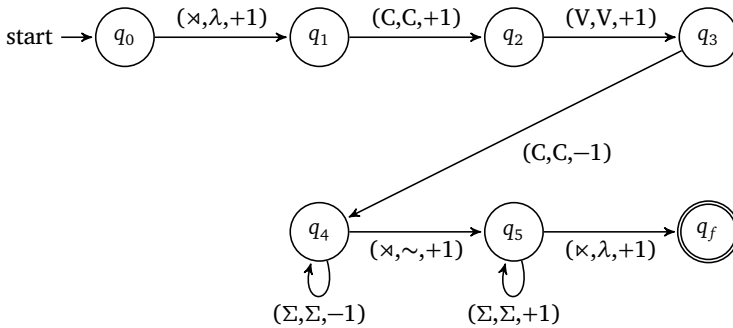


Figure 2:
2-way D-FST
for initial-CVC
reduplication

Generative capacity and computational complexity

3.3

With respect to acceptors, 1-way and 2-way finite-state acceptors are equivalent in expressive power. Both define the regular languages (Hopcroft and Ullman 1969; Shallit 2008). However, with respect to transducers, 1-way FSTs are strictly less expressive than 2-way D-FSTs (Savitch 1982; Aho *et al.* 1969; Filiot and Reynier 2016). For a 1-way

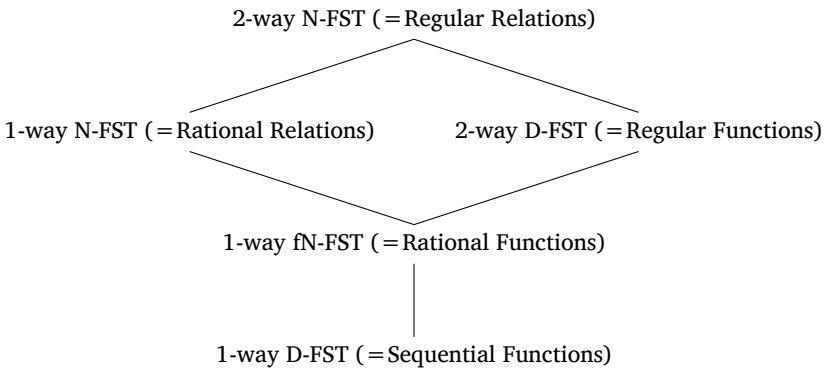
Table 2: Derivation of /takki/ → [tak~takki]

Outputting reduplicant	Outputting the base
1. ($q_0 \times \text{takki} \times$, λ , N/A)	8. ($\times q_5 \text{takki} \times$, tak~, $q_4 \xrightarrow[+1]{\times:\sim} q_5$)
2. ($\times q_1 \text{takki} \times$, λ , $q_0 \xrightarrow[+1]{\times:\lambda} q_1$)	9. ($\times t q_5 \text{akki} \times$, tak~t, $q_5 \xrightarrow[+1]{\Sigma:\Sigma} q_5$)
3. ($\times t q_2 \text{akki} \times$, t, $q_1 \xrightarrow[+1]{C:C} q_2$)	10. ($\times t a q_5 \text{kki} \times$, tak~ta, $q_5 \xrightarrow[+1]{\Sigma:\Sigma} q_5$)
4. ($\times t a q_3 \text{kki} \times$, ta, $q_2 \xrightarrow[+1]{V:V} q_3$)	11. ($\times t a k q_5 \text{ki} \times$, tak~tak, $q_5 \xrightarrow[+1]{\Sigma:\Sigma} q_5$)
5. ($\times t q_4 \text{akki} \times$, tak, $q_3 \xrightarrow[-1]{C:C} q_4$)	12. ($\times t a k k q_5 \text{i} \times$, tak~takk, $q_5 \xrightarrow[+1]{\Sigma:\Sigma} q_5$)
Going back to the start of the tape	
6. ($\times q_4 \text{takki} \times$, tak, $q_4 \xrightarrow[-1]{\Sigma:\lambda} q_4$)	13. ($\times t a k k i q_5 \times$, tak~takki, $q_5 \xrightarrow[+1]{\Sigma:\Sigma} q_5$)
7. ($q_4 \times \text{takki} \times$, tak, $q_4 \xrightarrow[-1]{\Sigma:\lambda} q_4$)	14. ($\times t a k k i \times q_f$, tak~takki, $q_5 \xrightarrow[+1]{\times:\lambda} q_f$)

FST, both the input language and the output language must be regular languages. Therefore a 1-way FST cannot have its output language be the non-regular copy language $L_{ww} = \{ww|w \in \Sigma^*\}$. In contrast, the output language of a 2-way D-FST can be a non-regular language such as L_{ww} .

Figure 3 shows the hierarchy of FSTs, adapted from Filiot and Reynier (2016, p.8). Different FSTs have different generative capacity, based on whether the FST is deterministic (D-FST), non-deterministic (N-FST), 1-way, 2-way, and/or functional (f-FST).

Figure 3:
Hierarchy
of FSTs



2-way D-FSTs are equivalent in expressivity to string transductions that are defined in Monadic Second Order logic (Engelfriet and Hoogetboom 2001) and to streaming string transducers (Alur 2010).⁷ 2-way D-FSTs are less powerful than Turing machines because they cannot move back and forth on the output tape. They are closed under composition (Chytil and Jákl 1977) and some important classes are closed under inverse (Courcelle and Engelfriet 2012, 526).

Because of the difference in expressivity between 1-way and 2-way D-FSTs, it makes sense to give different names to the classes of functions that they compute. We follow Filiot and Reynier (2016) who identify the class of functions describable with 1-way deterministic FSTs as ‘sequential functions’, with 1-way functional non-deterministic FSTs as ‘rational functions’, and with 2-way deterministic FSTs as ‘regular functions’. The non-deterministic counterparts for 1-way and 2-way D-FSTs are respectively the ‘rational relations’ and ‘regular relations’.

1-way D-FSTs run in time linear to the length of the input string. As for 2-way D-FSTs, one useful metric for measuring their complexity is in terms of the number of times the 2-way D-FST passes through the input (Baschenis *et al.* 2016). In the case of the reduplication examples in Section 3.2, the 2-way D-FSTs used only two passes through the input, one for each copy. Thus, the run time for those 2-way D-FSTs is at most $2n \cdot m$ where n is the number of passes and m is the length of the input. Since n here is fixed at 2, the run time is still linear in the size of the input string. To our knowledge existing applications of regular functions have been efficient (Alur and Černý 2011; Alur *et al.* 2014).

CONTRASTING 2-WAY D-FSTs WITH 1-WAY FSTs

4

Having illustrated how 2-way D-FSTs can model reduplication, here we contrast 2-way FSTs with 1-way FSTs on three criteria: empirical coverage, practical utility, and intensional description.

⁷ A streaming-string transducer (SST) is a 1-way FST that uses finitely many registers of unbounded size. These registers allow the SST to keep track of previous information on the input tape, thus simulating 2-way D-FSTs.

4.1 *Empirical coverage of the typology and productivity*

In terms of empirical coverage, 2-way D-FSTs can effectively model *virtually* the entire typology of reduplication as described by Moravcsik (1978), Hurch (2005), Inkelas and Zoll (2005), Rubino (2005), and Samuels (2010). We review part of this typology in Section 6. This stands in stark contrast to 1-way FSTs discussed in Section 2. We say *virtually* because there are two cases in the literature which require further discussion. These are discussed in Section 6.6.2.

4.2 *Practical utility and the RedTyp database*

To showcase the empirical coverage of 2-way D-FSTs and their practical utility, we have constructed the RedTyp database (Dolatian and Heinz 2019a).⁸ It contains entries for 138 reduplicative processes from 91 languages. These were gleaned from various surveys (Rubino 2005; Inkelas and Downing 2015a). 50 of these processes were from Moravcsik (1978), an early survey which is representative of the cross-linguistically most common reduplicative patterns.

RedTyp contains 57 distinct 2-way D-FSTs that model the 138 processes. Each 2-way D-FST was designed manually, implemented in Python, and checked for correctness. On average, these 2-way D-FSTs had 8.8 states. This shows that 2-way D-FSTs are concise and convenient computational descriptions and models for reduplicative morphology. This is in contrast to 1-way FSTs which suffer from an explosion of states when modeling partial reduplication.⁹

To our knowledge, the only other database on reduplication is the Graz Database on Reduplication (Hurch 2005 ff.). However, RedTyp differs from the Graz Database because the latter does not include computational representations or implementations of its entries.

⁸It can be found on the first author's GitHub page <https://github.com/jhdeov/RedTyp>.

⁹The largest 2-way D-FST in RedTyp is for verbal reduplication in Kinande (Downing 2000) with 29 states. This pattern depends on the size of the root and the number and type of suffixes and prefixes around it. In contrast, we estimate a deterministic 1-way D-FST would require over 1,000 states for this pattern of partial reduplication.

A comparison between the two databases is provided in Dolatian and Heinz (2019a).

RedTyp offers a useful corpus of reduplicative patterns for research. For example as described in Section 6, we have used this database to identify subclasses of 2-way D-FSTs for classifying the typology of reduplication. This corpus could be used to test for other universal computational properties of reduplication. Since it contains 2-way D-FSTs, it can also be used to generate reduplicated forms. Such data sets can be used to test morphological learning algorithms.

One shortcoming is that RedTyp under-represents cases of opacity in reduplication because our main source, Moravcsik (1978), did not list opaque cases. As discussed further in Sections 6.4.3–6.5, opacity can be said to occur when phonological processes exceptionally apply either across both copies or across neither copy because of a drive to maintain identity between the two copies (McCarthy and Prince 1995). Only 5% of RedTyp displays opacity. Furthermore, RedTyp focuses on morphological copying, not syntactic copying (cf. Koble 2006).

Linguistic motivation with origin semantics

4.3

Importantly, using 2-way D-FSTs for reduplication is linguistically motivated and matches the intensional descriptions behind the linguistic generalizations on reduplication.

2-way D-FSTs do not approximate reduplication like 1-way FSTs do. 2-way D-FSTs do not copy by remembering strings of segments (see Section 2). Instead they *actively and literally copy*.

This contrast between copying and remembering can be formalized with the notion of the *origin semantics* of a transduction (Bojańczyk 2014).¹⁰ Given a string-to-string function, the origin semantics of a function is the origin information of each symbol o_n in the output string. This is the position i_m of the read head on the input tape when the transducer had outputted o_n . To illustrate, consider a partial string-to-string function f_{ab} which maps ab to itself:

$$f(x) = \{(ab, ab)\}$$

¹⁰For an application of origin semantics to MCFGs and potentially to machine translation, see Nederhof and Vogler (2019).

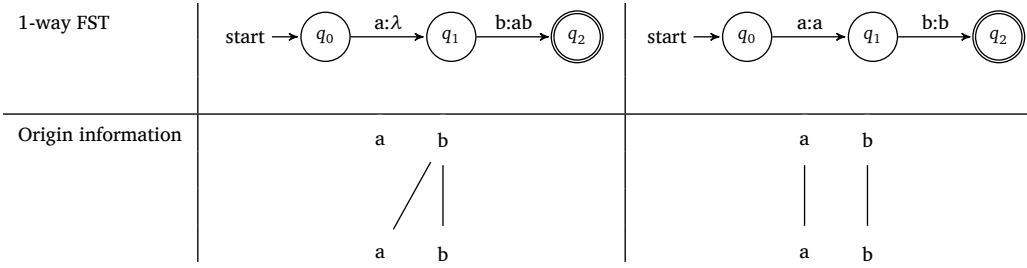


Figure 4: Pair of 1-way FSTs for the function f_{ab} and the origin information created by them for the mapping $ab \rightarrow ab$

As shown in the top row of Figure 4, this function can be modeled with at least two different 1-way FSTs which differ in *when* they output the output symbols a , b . In the bottom row of Figure 4, we visualize the origin information created by the two FSTs for the mapping (ab, ab) as graphs called *origin graphs* (Bojańczyk *et al.* 2017). The FSTs model the same function and are equivalent in their general semantics of what they output; however, they are not equivalent in their origin semantics because they use different *origin information* for their outputs.

This notion of origin semantics can be used to contrast how 1-way FSTs and 2-way FSTs model reduplication. Consider the toy example of initial-CV reduplication with a small alphabet $\Sigma = \{p, a, t\}$. This function can be modeled by either a 1-way or 2-way FST as in Figure 5. The two transducers in Figures 5 are equivalent in their general semantics because they can output the same string. For example, given the input pat , both FSTs will output $pa\sim pat$. However, the two FSTs differ in their origin semantics for the mapping $pat \rightarrow pa\sim pat$. Setting aside boundary symbols \times, \times, \sim , the 1-way FST associates the second pa string of the output with the vowel a of the input as in the bottom middle column of Figure 5. This is because the second pa was outputted when the 1-way FST was reading the a in the input. In contrast, the 2-way FST associates each segment in the output with an identical segment in the input as in the bottom right column of Figure 5.

The origin information created by the 2-way FST matches theoretical treatments of how the reduplicant’s segments are individually associated with identical segments in the input (Marantz 1982; Inke-

Reduplication with 2-way FSTs

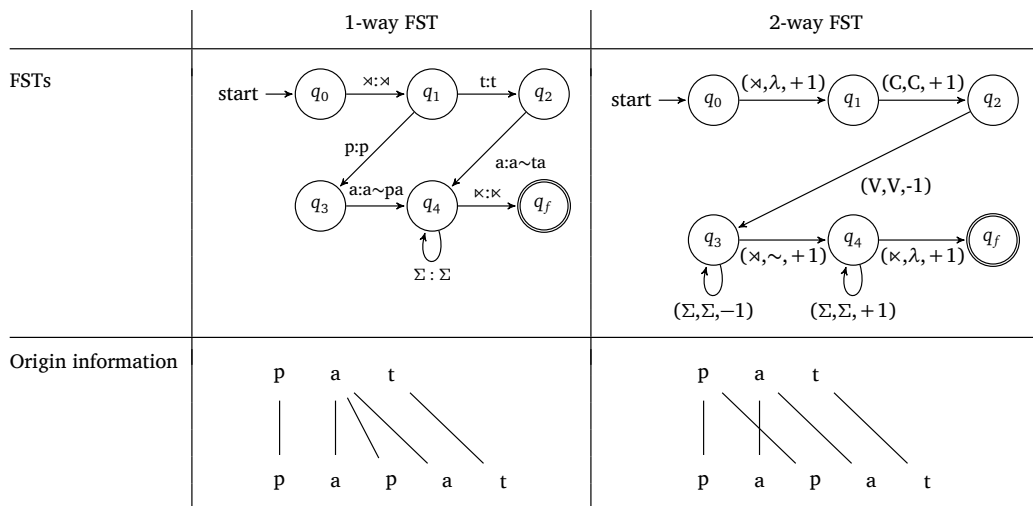


Figure 5: 1-way and 2-way FSTs for initial-CV reduplication and the origin information created by them for the mapping $pat \rightarrow pa\sim pat$

las and Zoll 2005).¹¹ In contrast, the origin information created by the 1-way FST does not match linguistic intuitions of reduplication because non-identical segments are associated. This difference in the origin semantics of the 1-way FST and 2-way FST formalizes their behavior: the 1-way FST simply *remembers* what strings of segments to output twice (Roark and Sproat 2007, 54), while the 2-way FST actively copies.

LINGUISTIC MOTIVATIONS FOR SUBCLASSES OF TRANSDUCERS

5

Having shown the utility of 2-way D-FSTs for reduplication, the next two sections show that reduplication does not require the full power

¹¹ In Base-Reduplicant correspondence theory or BRCT (McCarthy and Prince 1995), what matters for reduplication is not the relationship or correspondence between the input and output segments, but between the two copies in the output. Origin semantics might be able to formalize the intuition behind BRCT with finite-state technology, e.g. output symbols with the same origin are in correspondence. The only computational implementation of BRCT to our knowledge (Albro 2000, 2005) uses MCFGs to do so. Note however that the empirical validity of BRCT is questionable (Inkelas and Zoll 2005; McCarthy *et al.* 2012).

of 2-way D-FSTs but falls within certain subclasses. This means that reduplication has a demarcable generative capacity or complexity. In this section, we discuss the subclasses of 1-way FSTs that have been proposed to model segmental phonology (Section 5.1), specifically the Output-Strictly Local (OSL) functions and Sequential (Seq) functions. In Section 5.2, we discuss subclasses of 2-way FSTs and design new subclasses based on the concatenation of OSL and Seq functions. We explain the intuition behind using concatenation-based subclasses for reduplication (Section 5.3). The next Section 6 goes over the typology of reduplication and shows how it fits into these subclasses.

5.1 *Computational typology of phonology and 1-way transducers*

It is known that 1-way finite-state machines can model all attested phonological processes (Johnson 1972; Kaplan and Kay 1994; Mohri 1997). However, phonological processes do not require the full power of 1-way finite-state machines (Heinz 2007; Chandlee 2014). Subclass hierarchies have been discovered for 1-way FSAs (McNaughton and Papert 1971; Rogers and Pullum 2011; Heinz and Idsardi 2013) and 1-way FSTs (Garcia *et al.* 1990; Gainor *et al.* 2012; Heinz and Lai 2013; Chandlee *et al.* 2014). Some of these subclasses have been argued to characterize different types of phonological well-formedness conditions and transformations (Heinz 2018; Chandlee and Heinz 2018; Chandlee *et al.* 2018). We give a brief and informal overview.

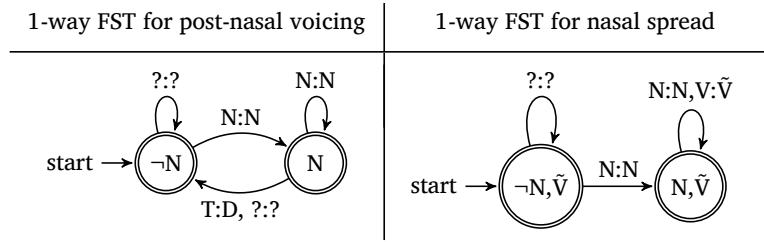
A common intuition in linguistic theory is that phonological processes are local or subject to adjacency constraints (Odden 1994). For example, a common phonological process is post-nasal voicing (5a) whereby voiceless stops are voiced after nasals. This process is local in the sense that the trigger for voicing (the nasal) is within a finite bound from the target of voicing (the stop). The symbols N, T, D represent nasals, voiceless stops, and voiced stops. Another local process is nasal spread whereby a vowel becomes nasalized after a nasal or nasalized vowel (5b). Nasal spread is iterative in that when a nasal triggers nasalization on a subsequent vowel, the newly nasalized vowel can then nasalize its subsequent vowel (5b-iii). The symbols V, \tilde{V} represent vowels and nasalized vowels.

- (5) a. **Post-nasal voicing**
 [+stop, -voice] → [+voice]/[+nasal] _ or T → D/N_
 i. /ata/ → [ata]
 ii. /anta/ → [anda]
- b. **Nasal spread**
 [+vowel] → [+nasal]/[+nasal] _ or V → $\tilde{V}/\{N, \tilde{V}\}_$
 i. /atapa/ → [atapa]
 ii. /anapa/ → [anãpa]
 iii. /anaapa/ → [anããpa]

Both processes are intuitively local. This intuition corresponds to Strict Locality in formal language theory (McNaughton and Papert 1971; Vaysse 1986; Rogers and Pullum 2011; Chandlee 2014). Formal definitions can be found in Chandlee *et al.* (2014, 2015). Informally, given an input string w , a function is Input-Strictly Local for a natural number k (k -ISL) if generating the output correspondent of some input symbol w_i relies on information about the current input symbol w_i and the $k - 1$ most recently seen input symbols. Post-nasal voicing is a 2-ISL function and it is computed by the 1-way FST in Figure 6. The symbol ? marks any other segment which isn't in an existing transition arc (Beesley and Karttunen 2003). The state labels are interpreted as keeping track of the last seen input symbol. The state labeled as N is where the post-nasal consonant is generated as voiced. The state label is interpreted as saying that a nasal was recently seen.

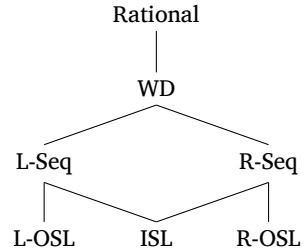
Output-Strictly Local functions for a number k (k -OSL) are analogously understood. A function is k -OSL if generating the output correspondent of w_i relies on information about the current input symbol w_i and the $k - 1$ most recently seen output symbols. An OSL function is L-OSL (R-OSL) if we read the input from the left (right), and write the output from the left (right). Nasal spread is a 2-L-OSL function and is computed by the 1-way FST in Figure 6. The state labels are interpreted as keeping track of the last recently generated output symbol. The state labeled as N, \tilde{V} is where a nasalized vowel is generated; the state label is interpreted as saying that the most recently outputted symbol was a nasal or nasalized vowel.

Figure 6:
1-way FSTs
for post-nasal
voicing and
nasal spread



ISL and OSL functions are part of a larger hierarchy of functions which are computed by 1-way FSTs. They are shown in Figure 7. The Sequential functions correspond to 1-way deterministic FSTs (1-way D-FST), while rational functions correspond to 1-way functional non-deterministic FSTs (1-way fN-FST) from Figure 3. WD stands for Weakly Deterministic functions which can compute some patterns in vowel harmony (Heinz and Lai 2013) and tone (Jardine 2016).

Figure 7:
Hierarchy
of 1-way FSTs



An example of an L-OSL function that will prove useful to model reduplication is truncation, such as nickname formation in English. This truncation process will output the first (C)VC of the input but delete everything after.¹²

(6) **English truncation**

- a. dʒɛfɹɪ → dʒɛf ‘Jeffrey’ → ‘Jeff’
- b. dɛrɪvɪd → dɛrɪv ‘David’ → ‘Dave’
- c. ælən → æl ‘Alan’ → ‘Al’

English nickname formation is a 3-L-OSL function because it requires a window of size three in the output tape. The window keeps

¹²We have simplified the analysis by not considering cases of complex onsets in the input, e.g. stɪvɪn → stɪv (‘Steven’ → ‘Steve’).

track of the last 2 symbols on the output tape and the current input symbol. The 3-OSL 1-way FST function in Figure 8 outputs up until the first VC of the input; it then stops outputting anything after that.¹³

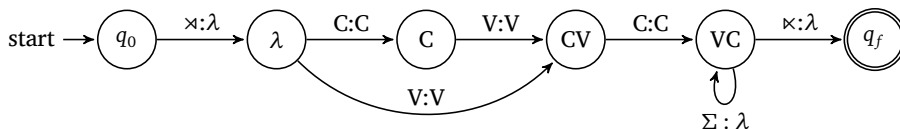


Figure 8: OSL 1-way FST for English nickname formation

A significant proportion of segmental phonology can be modeled with ISL and OSL functions (Chandlee 2014; Chandlee and Heinz 2018; Chandlee *et al.* 2018). Long-distance processes in phonology are however neither ISL or nor OSL. For example, Kikongo nasal harmony (7) requires the higher subclass of Sequential functions (Gainor *et al.* 2012). In Kikongo, alveolar stops like *d* or *l* surface as *n* if a nasal precedes them anywhere in the input. There can be any number of vowels and consonants intervening between the triggering nasal and the target alveolar (7c). This long-distance information means that the 1-way FST must keep track of whether a nasal consonant was seen *anywhere* in the input stem before it will output the alveolar.

(7) **Kikongo nasal harmony**

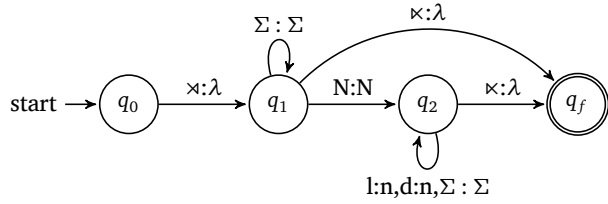
- | | | |
|----|---------------------------|-----------------------|
| a. | /sakid-ila/ → [sakid-ila] | ‘to congratulate for’ |
| b. | /mant-ila/ → [mant-ina] | ‘to climb for’ |
| c. | /tunik-idi/ → [tunik-ini] | ‘we ground’ |

The above pattern cannot be modeled by an ISL or OSL function but requires a Sequential 1-way FST as in Figure 9. A Seq 1-way FST is a deterministic 1-way FST that will read the input in only one direction (here left-to-right) and can use any information that it had found in the input string when processing the next input symbol.

To summarize, different types of phonological processes are computed by different subclasses of rational functions and with different subclasses of 1-way FSTs. The relatively low complexity of this subclasses has opened doors to understanding the cognitive limitations and learnability of phonological processes (Heinz 2018). In the next

¹³ See Chandlee (2017) on why this function is necessarily OSL and not ISL.

Figure 9:
Sequential 1-way FST
for Kikongo nasal harmony



section, we show that extending OSL and Sequential functions into 2-way FSTs opens similar doors for the typology of reduplication.

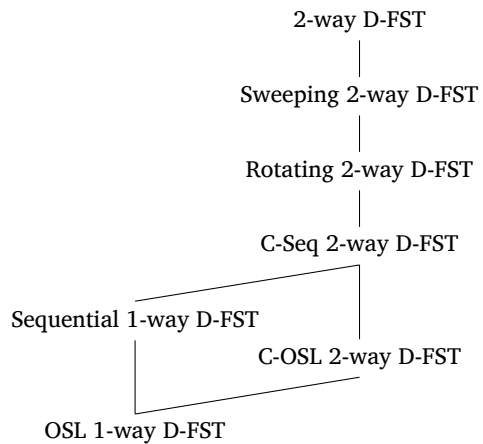
5.2 Subclasses of 2-way finite-state transducers

Unlike for 1-way FSTs, there is much less work on subclasses for 2-way FSTs. Some intuitive subclasses have been proposed in the literature. The typology of reduplication inspired us to devise additional subclasses. All of these subclasses, shown in Figure 10, restrict:

1. where the 2-way FST can rewind in the input,
2. what it can output while it is rewinding, and
3. what information can be transferred across multiple passes, i.e., if a later pass depends on an earlier pass.

At the top of the hierarchy are 2-way D-FSTs which correspond to regular functions. In regards to the first restriction, a 2-way FST is a **sweeping** transducer if the read head can change direction only at

Figure 10:
Hierarchy of subclasses
for 2-way FSTs



the ends of the input (Baschenis *et al.* 2015, 2016, 2018). A sweeping transducer is a generalization of similarly defined sweeping automata (sweeping 2-way FSAs) (Sipser 1980). For example, the 2-way FST for total reduplication in Figure 1 is a sweeping transducer. The only time the FST moves right-to-left is going from the end boundary \times to the start boundary \times . In contrast, the 2-way FST for initial-CVC partial reduplication in Figure 2 is not a sweeping transducer. It rewinds from the third input segment C to the beginning \times . However, the partial reduplication function computed by this 2-way FST can be computed by a sweeping transducer, which we show in the next section.

2-way D-FSTs are more expressive than deterministic sweeping 2-way D-FSTs. Consider the function $u_1\#u_2\#\dots\#u_n \mapsto u_n\dots u_2u_1$ where the input is a sequence of strings u_i separated by the special symbol $\#$. The output is formed by reversing these strings and deleting the $\#$'s. This function can be computed by a deterministic 2-way D-FST but not by a sweeping transducer. See Baschenis *et al.* (2016) for discussion.

In regards to the second restriction, a sweeping transducer is a **rotating** transducer if it does not output anything while it's moving right-to-left (Baschenis *et al.* 2017). The 2-way FST for total reduplication is a rotating transducer because it outputted nothing while moving right-to-left. Sweeping transducers are more expressive than rotating transducers. A sweeping transducer can compute the mirror function $w \rightarrow ww^r$, but a rotating transducer cannot.

As for the third restriction, we develop a set of concatenated-based subclasses of functions.

(8) *Subclasses of Regular Functions*

- a. **C-Seq function:** A Concatenated-Sequential function f is the concatenation of n Sequential functions s_i , e.g. $f(x) = s_1(x) \cdot s_2(x) \cdot \dots \cdot s_n(x)$. f is C-L-Seq (C-R-Seq) if the component Seq functions read the input left-to-right (right-to-left).¹⁴

¹⁴In terms of function combinatorics for regular string transformations (Alur *et al.* 2014; Dave *et al.* 2018), the class of C-Seq functions involves the use of a 'sum combinator' \otimes that concatenates the output of two or more Seq functions: $f(x) = s_1(x) \otimes s_2(x) \otimes \dots \otimes s_n(x)$ where s_i is a Seq function. This is similar to the use of product automata. See Alur *et al.* (2014) for details.

- b. **C-OSL function:** A Concatenated-OSL function f is the concatenation of n Output-Strictly Local functions o_i , e.g. $f(x) = o_1(x) \cdot o_2(x) \cdot \dots \cdot o_n(x)$. f is C-L-OSL (C-R-OSL) if the component OSL functions read the input left-to-right (right-to-left).

Rotating transducers are more expressive than C-Seq transducers, which are more expressive than C-OSL transducers. Examples witnessing these separations are drawn from the typology of reduplication in Section 6. C-Seq functions are more expressive than sequential functions (= 1-way D-FSTs) which are more expressive than OSL functions. A set of definitions is provided below for easier reference.

(9) *Subclasses of 2-way D-FSTs*

- a. **Sweeping 2-way FST:** A 2-way FST which can change direction only at the ends of the input
- b. **Rotating 2-way FST:** A sweeping 2-way FST which outputs nothing while moving right-to-left
- c. **C-Seq 2-way FST:** A rotating 2-way FST that computes a Concatenated Sequential function
- d. **C-OSL 2-way FST:** A rotating 2-way FST that computes a Concatenated Output-Strictly Local function

We have found that virtually the entire typology of reduplication can be modeled with deterministic rotating 2-way FSTs. Further, the bulk of the typology can be modeled with C-OSL functions. A few minor cases require C-Seq functions; these mostly involve infixal or internal reduplication. A smaller set of cases require the full power of rotating transducers; though these cases are not clear-cut. Before going through the typology in detail in Section 6, we illustrate the insight behind C-OSL functions and how they compute reduplicative processes.

5.3

Illustrating C-OSL

Intuitively, a C-OSL function is a function that takes as input the string x , gives x to n many separate 1-way FSTs which are OSL, and concatenates their output. To illustrate the insight behind C-OSL functions,

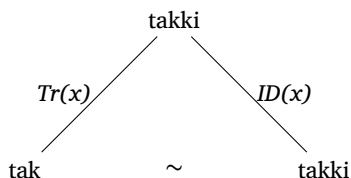


Figure 11:
Initial-CVC reduplication
as a concatenation of functions

consider initial-CVC partial reduplication from Agta again (Moravcsik 1978, 311) from (1a) repeated in (10a).

- (10) a. takki \rightarrow tak~takki 'leg' \rightarrow 'legs'
b. takki \rightarrow takki~takki

As an input-to-output function, reduplication may be viewed as submitting the same input to two separate functions in parallel and concatenating their output as in Figure 11. The first function, here labeled $Tr(x)$, truncates the input to the first CVC while the second function, $ID(x)$, is the identity function. The outputs of these two functions, *tak* and *takki*, are concatenated to form the reduplicated output: *tak~takki*. In (10b), we explicitly show how initial-CVC reduplication can be seen as truncating the first copy: *takki* \rightarrow ~~*takki*~~~*takki* where truncated material is shown in strike-through.

The truncation function $Tr(x)$ is a 3-L-OSL function because it outputs a truncation of the input to just the first CVC. This is similar to English nickname formation from Section 5.1. The identity function $ID(x)$ is both 1-L-OSL and 1-R-OSL. Thus both $Tr(x)$ and $ID(x)$ are L-OSL and hence their concatenation is C-OSL. Figure 12 illustrates a 2-way D-FST for initial-CVC reduplication which is formulated as a concatenation of these two OSL functions. Contrast this model of initial-CVC reduplication (shown in Figure 12) with the non-rotating 2-way D-FST shown in Figure 2. (It is the the additional state CV_1 and its transition arcs in Figure 12 which make this D-FST rotating.)

To summarize this section, understanding reduplicative processes as C-OSL and C-Seq functions is intuitive. This analysis echoes Steriade (1988)'s treatment of partial reduplication as total reduplication followed by truncation and Inkelas and Zoll (2005)'s treatment of total reduplication as morphological doubling.

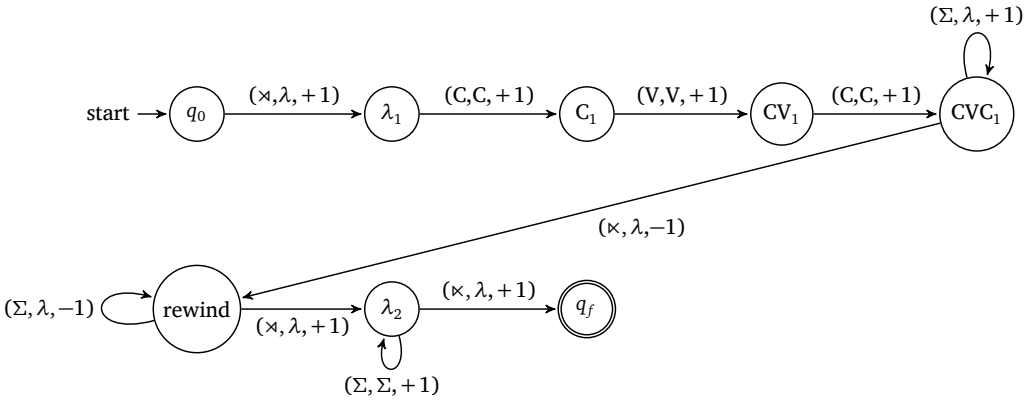


Figure 12: C-OSL 2-way D-FST for initial-CVC partial reduplication

6 COMPUTATIONAL TYPOLOGY OF REDUPLICATION

This section provides a detailed, comprehensive review of the RedTyp typology and classifies RedTyp’s reduplicative processes according to the computational classification introduced in the last section. The main finding is that most processes are C-OSL (Section 6.2) and most of the ones that are not are C-Seq (Section 6.3). There are few (and questionable) cases where reduplication needs the full power of 2-way D-FSTs (Section 6.4). We give an overview in Section 6.5.

Note that all partial reduplicative processes can be computed by 1-way FSTs. However, in order to get the linguistically-motivated origin semantics right (Section 4.3), we need the additional power of 2-way FSTs. Thus in this section, when we discuss how partial reduplicative cases fit into the subclasses of 2-way FSTs, we mean in terms of them generating the right origin semantics.

6.1 Preliminaries to the typology

Although reduplication is cross-linguistically ubiquitous, there is a wide cross-linguistic variation in a) what substring or subsequence gets repeated, b) where the copied substring or subsequence is placed

in the output, and c) whether and how phonological processes interact with copying. This section provides a brief but representative typology of reduplication compiled from various surveys (Moravcsik 1978; Rubino 2005; Inkelas and Downing 2015a).

We emphasize that our reported typology is descriptive and not theoretical. Various theoretical frameworks have been developed to account for the range of variation on reduplication (Marantz 1982; McCarthy and Prince 1995; Spaelti 1997; Raimy 2000; Inkelas and Zoll 2005; Frampton 2009; Samuels 2010; McCarthy *et al.* 2012; Saba Kirchner 2010, 2013). The reader is referred to elsewhere for theoretical overviews (Raimy 2011; Urbanczyk 2007, 2011; Inkelas and Downing 2015a,b).

We define the following *descriptive* terms which will be useful in categorizing different reduplicative processes:

(11) *Terminology for categorizing the typology:*

- **reduplicant:** the substring in the output which was created via copying
- **base:** the substring in the output which was not created via copying
- **target:** the substring in the input which will be copied or duplicated
- **anchor point:** the position in the input where the target starts or ends
- **source:** the morphological or phonological constituent in the input which contains the target

The output-based terms *base* and *reduplicant* are common in the literature on reduplication (McCarthy and Prince 1995) though their definition is problematic (Shaw 2005; Haugen 2009). Anchor points have been proposed for reduplication (Fitzpatrick 2006; Raimy 2009) and other non-concatenative processes (Yu 2007; Samuels 2010). We introduce the input-based terms *source* and *target* in order to better fully describe reduplication as an input-to-output function. This section goes through the typology of reduplication, organized in terms of how they vary in the source, target, and/or reduplicant. These variations align with what type of 2-way FST is needed to compute them.

6.2

Most reduplication is C-OSL

Most reduplicative processes are C-OSL. We go through common and some uncommon reduplicative processes and show they are C-OSL. For a function to be C-OSL, the two copies must be independent of each other, and the two passes over the input must likewise be independent of each other. Informally, some criteria for a C-OSL function are that each of the component functions:

1. reference only a finite and bounded number of the most recently generated output symbols, meaning that each of the functions,
2. do not depend on any long-distance information in the input,
3. do not use any finite lookahead or finite lookback on the input,
4. do not rely on deleted material, and
5. do not rely on any information from the other function.

6.2.1

Total and word-initial partial reduplication

Total reduplication and word-initial partial reduplication are C-L-OSL, which means they are the concatenation of two L-OSL functions.

Consider total reduplication first.

- (12) a. **Total reduplication**
Indonesian (Cohn 1989)
 buku → buku~buku ‘book’ → ‘books’
- b. **Initial-CVC reduplication**
Agta (Moravcsik 1978, 311)
 takki → tak~takki ‘leg’ → ‘legs’

For total reduplication, the two OSL functions are identity $ID(x)$. (Total reduplication is also C-R-OSL.)

For partial reduplication, there is limited variety in the shape of the copied material, the *reduplicant*. Some languages have a partial reduplicative process that copies the first *C* or consonant of the word (13a), first CV or consonant-vowel sequence (CV) of the word (13b), first CVC sequence (13c), or first CV(C)CV sequence (13d). In general, the copied material has to fit into some template of a particular size.

- (13) *Common prefixal partial reduplicative patterns*
- a. **Initial-C reduplication**
Shilh (Moravcsik 1978, 308)
gen → g~gen ‘to sleep’ → ‘to be sleeping’
 - b. **Initial-CV reduplication**
Sundanese (Moravcsik 1978, 319)
guyon → gu~guyon ‘to jest’ → ‘to jest repeatedly’
 - c. **Initial-CVC reduplication**
Panganisan (Rubino 2005, 11)
baley → bal~baley ‘town’ → ‘towns’
 - d. **Initial-CV(C)CV reduplication**
Dyirbal (McCarthy *et al.* 2012, 187)
 - a. baniju → bani~baniju ‘come’
 - b. balgan → balga~balgan ‘laugh’

As for the partial reduplication functions in (13), they are all C-L-OSL just like initial-CVC reduplication from Section 5.3. They involve the concatenation of a truncation $Tr(x)$ and identity function $ID(x)$. The truncation function varies in terms of how much word-initial material is faithfully outputted.

Table 3 illustrates these examples where the truncated material is shown in strike-through. The outputs of the two component OSL function are separated by ~ for illustration.

Table 3: C-OSL treatment for total and initial partial reduplication

	Total (12a)	Initial-C (13a)	Initial-CV (13b)	Initial-CVC (13c)	Initial-CV(C)CV (13d)
Input x	buku	gen	guyon	baley	balgan
Components	$ID(x) \cdot ID(x)$	$Tr(x) \cdot ID(x)$	$Tr(x) \cdot ID(x)$	$Tr(x) \cdot ID(x)$	$Tr(x) \cdot ID(x)$
Output	buku~buku	gen~gen	guyon~guyon	baley~baley	balgan~balgan
Subclass	C-L-OSL C-R-OSL	C-L-OSL	C-L-OSL	C-L-OSL	C-L-OSL

Variation in the number and placement of copies

6.2.2

The typology is larger than the above examples. Some languages create three copies of the input (triplication) instead of just two (14a).

Some reduplicative processes are suffixal; they specify that the location of the target be a word-final substring (14b) instead of word-initial substring (13d). Some reduplicative process are wrong-sided by making the target and the reduplicant not adjacent in the output, i.e., copying the final CVC and placing it at the beginning of the output (14c vs. 13c). There are likewise cases where both the base and the reduplicant are shortened or truncated in the output, e.g., truncating both copies to CV (14d).

(14) *Variation in number and reduplicant placement*

a. **Total triplication**

Mokilese (Moravcsik 1978, 301)

roar → roar~roar~roar

‘give a shudder’ → ‘continue to shudder’

b. **Final-CVCV reduplication**

Siriono (Moravcsik 1978, 308)

erasi → erasi~rasi

‘he is sick’ → ‘he continues being sick’

c. **Initial-CVC reduplication and opposite-edge or wrong-sided placement**

Koryat (Riggle 2004, 3)

qanga → qanga~qan ‘fire’ → ‘fire (ABS)’

d. **Abbreviated reduplication (*Kager-Hamilton Problem*)**

Guarijio (Caballero 2006)¹⁵

toni → to~to ‘to boil’ → ‘to start boiling’

muhiba → mu~mu ‘to throw’ → ‘to start throwing’

All these processes are still C-OSL, however. They differ in the number and order of concatenated functions, the direction in which the input is read, and whether all or none of the functions are identity. Their computation is visualized in Table 4. Triplication is C-L-OSL and C-R-OSL; it involves concatenating three identity functions. Suffixal reduplication like final-CVCV reduplication is C-R-OSL because

¹⁵Such reduplication is often argued to be unattested and is called the *Kager-Hamilton Problem* (Idsardi and Raimy 2008). See Caballero (2006) for discussion on what prosodic and morphological factors condition this rare type of reduplication.

Table 4: C-OSL treatment for less common reduplication patterns

	Triplication (14a)	Final-CVCV (14b)	Wrong-sided (14c)	Abbreviated (14d)
Input x	roar	erasi	qanga	toni
Components	$ID(x) \cdot ID(x) \cdot ID(x)$	$ID(x) \cdot Tr(x)$	$ID(x) \cdot Tr(x)$	$Tr(x) \cdot Tr(x)$
Output	roar~roar~roar	erasi~erasi	qanga~qanga	toni~toni
Subclass	C-L-OSL C-R-OSL	C-R-OSL	C-L-OSL	C-L-OSL

it is the concatenation of an identity function and an R-OSL truncation function. The truncation function reads the input right-to-left and deletes everything to the left of the final CVCV. Wrong-sided initial-CVC reduplication is C-L-OSL. It differs from initial-CVC reduplication by ordering identity before truncation. Abbreviated reduplication is C-L-OSL. Unlike initial-CV copying, it is composed of two L-OSL truncation functions instead of just one.

Copying a morphological subconstituent

6.2.3

In the above examples, the *source* was the entire input. But unlike concatenative morphology, reduplication is often sensitive to word-internal morphological constituents, contra Bracket Erasure (Kiparsky 1982). In these cases, the semantic function of reduplication builds on the meaning of the entire input while the location of the reduplicant is word-internal (cf. Aronoff 1988). For example, some languages have reduplication target a morphological subconstituent within the input as the source, such as a root/stem (15a, 15b) or affix (15c), and whether for total reduplication (15a, 15c) or partial reduplication (15b). The source and reduplicant are usually adjacent; though there are some cases where the two copies are non-adjacent in the output, e.g., Madurese copies the root-final CVC and places it at the beginning of the output (15d).

(15) *Copying from a morphological subconstituent*a. **Total reduplication of the stem***KiHehe* (Aronoff 1988, 8)

ku-haata → ku-haata~haata

'to ferment' → 'to start fermenting'

Table 5: C-OSL treatment for copying morphological subconstituents

	KiHehe	Bikol	Hungarian	Madurese
	(15a)	(15b)	(15c)	(15d)
Input x	$ku\{r,haata\}_r$	$na\{r,murak\}_r$	$_p\{el\}_p\ megy$	$pa\{r,jalan\}_r\ an$
Components	$L(x) \cdot R(x)$	$L(x) \cdot R(x)$	$L(x) \cdot R(x)$	$L(x) \cdot R(x)$
Output	$ku\{r,haata\}_r \sim$ $\text{ku} \{r,haata\}_r$	$na\{r,murak\}_r \sim$ $\text{na} \{r,murak\}_r$	$\{p,el\}_p\ megy \sim$ $\{p,el\}_p\ megy$	$pa\{r,jalan\}_r\ an \sim$ $pa\{r,jalan\}_r\ an$
Subclass	C-L-OSL C-R-OSL	C-L-OSL	C-L-OSL C-R-OSL	C-R-OSL

b. Initial CV reduplication of the stem

Bikol (Mattes 2007, 84)

$na\text{-}murak \rightarrow na\text{-}mu\sim murak$

‘to flower’ \rightarrow ‘decorating with flowers’

c. Total reduplication of an affix (prefix)

Hungarian (Inkelas and Downing 2015a, 505)

a. $el\text{-}megy \rightarrow el\sim el\text{-}megy$

‘he goes there’ \rightarrow ‘he occasionally goes there’

b. $bele\text{-}nez \rightarrow bele\sim bele\text{-}nez$

‘he looks into it’ \rightarrow ‘he occasionally looks into it’

d. Root-final CVC reduplication and word-initial placement

Madurese (Brown 2017, 964)

$pa\text{-}jalan\text{-}an \rightarrow lan\sim pa\text{-}jalan\text{-}an$

‘pedestrian’ \rightarrow ‘pedestrians’

More cases of reduplication targeting a morphological subconstituent are well-attested (Shaw 2005; Inkelas and Zoll 2005; Haugen 2009; Hyman 2009; Inkelas 2014). The above cases are C-OSL if the relevant morphological boundaries are present in the input. Their computation is visualized in Table 5. Each process uses two functions $L(x), R(x)$ which generate the two copies, reference the morphological boundaries, and they crucially output these boundaries.

For total copying in KiHehe and Hungarian, the function is C-L-OSL and C-R-OSL. For total stem copying in KiHehe, the first function $L(x)$ outputs everything up until the root right-boundary ‘ $\}_r$ ’. The sec-

ond function $R(x)$ outputs nothing until it sees the root left-boundary $\{_r$; it outputs this and everything after it. Both functions are both L-OSL and R-OSL, thus KiHehe is C-L-OSL and C-R-OSL. Prefix copying in Hungarian is similarly defined but for the prefix boundaries $\{_p$ and $\}_p$. Partial stem copying in Bikol is only C-L-OSL. The function $L(x)$ outputs everything up until it outputs the string $\{_r CV$; it deletes everything after that. The function $R(x)$ outputs nothing up until it sees the root left-boundary $\{_r$; it outputs this and everything after it. Non-local copying in Madurese is C-R-OSL. The function $L(x)$ reads the input right-to-left; it outputs nothing until it sees the root right-boundary $\}_r$; it outputs this and the first CVC that it sees. After that, it deletes everything. The function $R(x)$ is the identity function.

Even though some have argued against the use of morpheme boundaries in morpho-phonological representations (Anderson 1992; Stump 2001), morphological boundaries must be part of the input for a finite-state systems like ours (e.g. Karttunen 1983; Koskeniemi 1984; Roark and Sproat 2007).¹⁶ Consider partial stem copying in Bikol: $na\{_r murak\}_r \rightarrow na\{_r mu\{_r murak\}_r$. Without the root boundaries, we could not distinguish the prefixed input $na\{_r murak\}_r$ from a hypothetical mono-morphemic input $namurak$. Without some way to encode the relevant morphological constituents in the input, we simply cannot define this reduplication function with any type of 1-way or 2-way FSTs. The use of such boundaries in finite-state morphology is standard practice.

Copying a prosodic subconstituent

6.2.4

Besides morphological subconstituents, the source can also be a metrical or prosodic subconstituent such as the stressed syllable (16a), the first syllable (16b), or the first foot (16c). The source can also

¹⁶It should be noted though that HPSG-based approaches to computational morphology (Bonami and Crysmann 2013, 2016; Crysmann and Bonami 2016) do not need morpheme boundaries as symbols in their alphabet. One reason is because they can essentially directly capture hidden morphological structure or constituency. Another strategy is to temporally apply reduplication to the subconstituent and then later add the other affixes, e.g., $haata \rightarrow haata \sim haata \rightarrow kuhaata \sim haata$. This is a common strategy in handling morphology-semantics bracketing paradoxes (cf. Stump 1995, 2001).

be a morphophonological constituent, e.g. a prosodic stem (16d). In Chumash, the prosodic stem (underlined) consists of all the segments in the morphological stem alongside any prefixal consonants that are syllabified with the morphological stem.

- (16) *Copying from a metrical or prosodic subconstituent*
- a. **CV-reduplication of the stressed syllable**
Chamorro (Inkelas and Downing 2015a, 507)
 hu.gán.do → hu.gá~gan.do ‘play’ → ‘playing’
 - b. **Total reduplication of the initial syllable**
Hiaki (Haugen 2009)
 vu.sa → vu~vu.sa ‘awaken’
 vam.se → vam~vam.se ‘hurry’
 - c. **Total reduplication of the initial foot**
Yidiny (Marantz 1982, 453)
 (gindal)ba → gindal~gindalba ‘lizard sp.’ → ‘lizards’
 - d. **Initial-CVC reduplication of the prosodic stem**
Chumash (Downing 1998, 101)
 s + tʃeq → s-tʃeq~tʃeq ‘it is very torn’
 s + ikuk → s + ik~s-ikuk ‘he is chopping’

If the relevant prosodic boundaries are in the input, the computation is C-OSL. The computation proceeds the same as for copying a morphological constituent. Table 6 shows this with syllable boundaries $(_s)_s$, foot boundaries $(_f)_f$, stressed syllable boundaries $(_s)_s$, and prosodic stem boundaries $(_{PS})_{PS}$.¹⁷

Given an unsyllabified input, these prosodic boundaries can be generated via a 1-way FST (Hulden 2006; Yu 2017) which can be ISL because it uses finite lookahead on the input (see also Strother-Garcia 2018, 2019).

However, if the input to reduplication lacks boundaries, then reduplication is C-Seq because we need finite lookahead to know if some consonant is part of the relevant prosodic constituent. Consider

¹⁷The second function $R(x)$ in stressed syllable CV-copying must change stressed \acute{a} to unstressed a . Stressed syllable copying is also C-R-OSL if the first function $L(x)$ is R-OSL and outputs the right-boundary $)_s$. Generating the prosodic stem in Chumash requires reference to morphological boundaries too.

Table 6: C-OSL treatment for copying prosodic subconstituents

	Chamorro (16a)	Hiaki (16b)		Yidiny (16c)	Chumash (16d)
Input x	hugándo	vusa	vamse	gindalba	s + ikuk
Syllabify x	hu _(s) gán _(s) do	(_(s) vu _(s) (_(s) sa) _(s)	(_(s) vam) _(s) (_(s) se) _(s)	(_(f) gindal) _(f) ba	(_(p_S) s + ikuk) _(p_S)
Components	$L(x) \cdot R(x)$	$L(x) \cdot R(x)$	$L(x) \cdot R(x)$	$L(x) \cdot R(x)$	$L(x) \cdot R(x)$
Output	hu _(s) gá } _(s) ndo ~ h _(s) gán _(s) do	(_(s) vu _(s) } _(s) sa) _(s) ~ (_(s) vu) _(s) (_(s) sa) _(s)	(_(s) vam) _(s) } _(s) se) _(s) ~ (_(s) vam) _(s) (_(s) se) _(s)	(_(f) gindal) _(f) } _(f) ba ~ (_(f) gindal) _(f) ba	(_(p_S) sik } _(p_S) uk) _(p_S) ~ (_(p_S) sikuk) _(p_S)
Subclass	C-L-OSL C-R-OSL	C-L-OSL	C-L-OSL	C-L-OSL	C-L-OSL

initial syllable copying in Hiaki $vusa \rightarrow vu \sim vusa$. In the first function $L(x)$, the consonant s is not generated because it is part of the next syllable. We know because it precedes a vowel. In contrast for $vamse \rightarrow vam \sim vamse$, the consonant m is copied because it precedes a consonant. The use of such information from finite lookahead on the input cannot be computed by an OSL function.

This section presented cases in RedTyp which are C-OSL. They comprise the bulk of reduplicative typology. Of the 138 reduplicative processes in RedTyp (Section 4.2), 121 (87%) were C-OSL.

Some reduplication is C-Seq

6.3

This section goes through some types of reduplication which are not C-OSL but are instead C-Seq. Informally, a reduplicative function is C-Seq if its component functions do not rely on any information from the other function or its output. A component function can use finite lookback, finite lookahead, or even long-distance information in the input.

Internal reduplication and gray areas between C-OSL or C-Seq

6.3.1

One problematic area for C-OSL are internal reduplication functions which seem infixal (Broselow and McCarthy 1983; Gafos 1998; Spaelti 1997). Some of these are C-OSL, some are not. These functions are C-OSL if the truncation functions can uniquely determine what segments to delete based on only what was outputted.

In the previous sections, the reduplication's target can be thought of as a contiguous substring that is determined by scanning either the left or right edge of the source. In those cases, the target was at the edge of the source and the reduplicant was placed at the left or right edge of the base. However, there are cases of *internal* or *infixal* reduplication where the target is a substring *inside* the source such that the substring is not strictly adjacent to the source's edges (17a, 17b) and the reduplicant is placed inside the base in the output (17c). In (17c), the word-initial C is copied and placed after the first vowel.

- (17) *Internal reduplication cases which are arguably not C-OSL*
- a. **Leftmost-VCC* reduplication**
Mangarayi (Raimy 2000, 135)
gabuji → *g-ab~abuji* 'old person' → 'old persons'
 - b. **Rightmost-CV reduplication**
Chamorro (Inkelas and Zoll 2005, 107)
nalaŋ → *nala~laŋ* 'hungry' → 'very hungry'
 - c. **Initial-C reduplication and internal placement**
Quileute (Broselow and McCarthy 1983, 44)
t^siko → *t^si~tko* 'he failed sp.' → 'he failed (freq.)'

Table 7 visualizes these processes. A traditional analysis is that the reduplicant is infixal (Broselow and McCarthy 1983) where < > marks infixation, e.g., Mangarayi *gabuji* → *gab* < *ab* > *buji*. However, their treatment as C-OSL is somewhat counter-intuitive because a C-OSL function models these processes as concatenating two truncation functions: *gabuji*~*gabuji*. The first function *L(x)* outputs the first C*VC* substring and deletes everything after that. The second function *R(x)* deletes all word-initial strings of consonants C*; once it sees a vowel V, it outputs it and everything after it.

6.3.2 Internal or non-contiguous reduplication which is C-Seq

The main reason why Mangarayi and the other functions in Table 7 are C-OSL is because the target and deleted materials do not have the same shape. Knowing what to delete or generate doesn't need any finite lookahead or lookback over the input, just over the output. However, other cases of internal and non-contiguous reduplication do require such finite lookback/lookahead over the input. This makes them

Reduplication with 2-way FSTs

Case	Mangarayi (17a)	Chamorro (17b)	Quileute (17c)
Input x	gabuji	nalaŋ	t ^s iko
Output	gababuji	nalalaŋ	t ^s itko
Infixed treatment	gab<ab>uji	nala<la>ŋ	t ^s i<t>ko
C-OSL treatment	gab <u>u</u> ji~gabuji	nala <u>ŋ</u> ~nalaŋ	t ^s ik <u>ø</u> ~tiko
Subclass	C-L-OSL	C-R-OSL	C-L-OSL

Table 7:
Infixed vs. C-OSL
treatment of
internal
reduplication

C-Seq. In (18a), the penultimate syllable is reduplicated. In (18b), the word-initial CV is copied and placed before the final C. In most of these cases, the target is a contiguous substring in the input. In some cases, the target is not contiguous (18c). In (18c), the input's first CV and final C are copied and placed together at the beginning of the output.

(18) *Internal reduplication which are C-Seq*

a. **Penultimate syllable reduplication**

Samoan (Moravcsik 1978, 301,310)

a.lo.fa → a.lo~.lo.fa

'he loves' → 'they love'

ta.o.to → to.o~o.to

'he lies' → 'they lie'

b. **Initial-CV reduplication and internal placement**

Creek (Riggle 2004, 3)

fayatk + i: → fayat~fa-k + i:

'crooked' → 'crooked (pl.)'

c. **Double-sided reduplication**

Nisgha (Urbanczyk 2007, 474)¹⁸

lút'tux^w → lúx^w~lút'tux^w

'to value' → 'to value (pl.)'

These C-Seq processes are visualized in Table 8. Consider penultimate syllable copying in Samoan with two truncation functions *a.lo.fa* → *a.lo.fã~ã.lo.fa*. If the input is read left-to-right, the first function must output everything up until the penultimate syllable: *a.lo.fã*. This is not OSL because knowledge about whether some syllable is penultimate or not requires finite lookahead on the input. If the input is instead read right-to-left, the first truncation function is still not OSL. The function would delete the last vowel and the last consonant; but once it sees the penultimate vowel, the function cannot

¹⁸We set aside issues in predicting the quality of the vowel (Shaw 2005).

Table 8:
Non-C-OSL
patterns of
internal or
non-contiguous
reduplication

Case	Samoan (16a)	Creek (18b)	Nisgha (18c)
Input x	a.lo.fa	fayatk	lút'tux ^w
Components	$L(x) \cdot R(x)$	$L(x) \cdot R(x)$	$L(x) \cdot R(x)$
Infixal treatment	a.lo < lo > fa	fayat < fa > k	lút ^w ~ lút'tux ^w
C-Seq treatment	a.lo-fa ~ a.lo.fa	fayatk ~ fayatk	lút'tux ^w ~ lút'tux ^w
Subclass	C-Seq	C-Seq	C-Seq

determine if this vowel is penultimate or not based on only what it has outputted. The function would need lookback access to the input. In both left-to-right and right-to-left cases, the truncation functions are Seq. The other processes in Table 8 are not C-OSL for similar reasons.

Although penultimate syllable copying is not C-OSL, Samoan has penultimate stress (Zuraw *et al.* 2014). Stressed syllable copying is C-OSL. This is an argument for reanalyzing Samoan as instead copying the stressed syllable. This relates to Nelson’s (2003, 117) hypothesis that any references to the penultimate position in reduplication must be prosodic. Note that for Creek and Nisgha, the component functions are Seq; however they do not generate the right origin semantics because of unbounded word-internal deletion. The first function for Nisgha deletes everything except for the first CV and last C $lut'tux^w$. If read left-to-right, in order to generate the final C, we need to move to the next symbol and check if it is \times ; thus the final C is generated as an output correspondent for the end-boundary \times . If read left-to-right, in order to generate the initial CV, we move on to the preceding symbol and check if it is \times ; thus the first CV are generated as output correspondents to the start-boundary \times . This is because Seq functions are deterministic. In contrast, the right origin semantics would be generated if the component functions were non-deterministic or if the function was computed by a full 2-way FST.¹⁹

¹⁹To exactly capture the right origin semantics, it is possible that a subclass of *Streaming-string transducers* (1-way FSTs with registers) (Alur and Černý 2011; Alur and Deshmukh 2011) are a suitable alternative for modeling infixal reduplication. Discovering subclasses of SSTs and their relations to subclasses of 2-way D-FSTs is a worthwhile open question.

The above cases with infixation showed that capturing the right origin semantics might require classes which are more expressive than C-OSL and C-Seq. In this section, we go through more cases. Some are ambiguously C-Seq depending on the analysis; others must use rotating or even unrestricted 2-way D-FSTs in order to capture the right origin semantics.

Reduplication with syllable-count

6.4.1

Reduplication that is sensitive to syllables may involve iteration (19a) or minimality requirements on what is reduplicated (19b). Both examples are from Mandarin for different reduplicative processes. In (19a), reduplication is iterative because each syllable undergoes total reduplication: an input of the form A.B has A.A.B.B as the output. In (19b), a word undergoes total reduplication if it is monosyllabic, otherwise the morpheme *meei* is added.

(19) *Reduplication and syllable number*a. **Iterative reduplication of syllables**

Mandarin (Moravcsik 1978, 314)

huang.jang → huang~huang-jang~jang

‘flustered’ → ‘flustered (vivid form)’

b. **Minimality in reduplication**

Mandarin (Moravcsik 1978, 305-6).

jang → jang~jang ‘sheet’ → ‘every sheet’

jia.luen → mee-i-jia.luen ‘gallon’ → ‘every gallon’

Iterative copying (19a) is C-OSL if the number of iterations (= number of syllables) is bounded. The individual syllables must also be uniquely identifiable in the input. For Mandarin, the function is made up of four concatenated OSL truncation functions. The first two functions output everything up until the medial syllable boundary ‘.’ while the latter two delete everything up until the ‘.’ boundary.

(20) *Mandarin iterative reduplication as C-OSL:*

$$L1(x) \cdot L2(x) \cdot R1(x) \cdot R2(x)$$

huang.jang~huang.jang~huang.jang~huang.jang.

We are unaware of any examples showing reduplicative processes which iterate over inputs that have at least three syllables. So while Mandarin provides examples of $A \rightarrow A\sim A$ and $A.B \rightarrow A.A\sim B.B$, we have no examples of $*A.B.C \rightarrow A.A\sim B.B\sim C.C$. We likewise have not seen cases of trisyllabic iterative copying in other languages. This is computationally significant. If Mandarin allowed iterative copying over trisyllabic words, then generating the right origin semantics would need as many passes over the input as there are syllables in the input. The function would either need the full power of a 2-way D-FST in order to generate the right origin semantics, otherwise we could use a 1-way FST that has the linguistically-unmotivated origin semantics.

As for minimality requirements (19b), this cannot be computed by a C-Seq transducer with the right origin semantics. In order to reduplicate a monosyllabic input: $jang \rightarrow jang\sim jang$, we use two concatenated identity functions. But to block reduplication in a bisyllabic input $jia.luen \rightarrow meei-jia.luen$, we need to check that the input does not contain any medial syllable boundaries. The first function would need to use to finite lookahead before choosing to output the first segment j or the prefix $meei$. As with the infixation cases in Section 6.3.2, a C-Seq 2-way FST can do so but it then generates the wrong origin semantics because it associates the output segment j with the input syllable boundary ‘.’. Generating the right origin semantics needs a rotating 2-way D-FST that involves three passes. The first pass reads the input and checks if it is monosyllabic or not. If yes, the second and third passes apply the identity function: $jang\sim jang$. If no, the second pass outputs the prefix and the base $meei-jia.luen$; there is no third pass.

6.4.2

Phonological changes to the reduplicant

The previous section illustrated how C-Seq 2-way FSTs are distinct from rotating 2-way FSTs. In the latter, a pass can transfer information (e.g., *is the input monosyllabic*) to a later pass. Similar information transfer is required in certain cases where phonological processes interact with reduplication. We first go over cases where we arguably do not need such information transfer.

Reduplicative patterns do not only involve copying. In addition to copying segments, a reduplicative process may involve a host of other

L-OSL, their composition is L-OSL, and the concatenation with $ID(x)$ is C-L-OSL. Complex onset reduction in Tagalog and echo reduplication in Turkish are likewise C-L-OSL and consist of the concatenation of a composed L-OSL function with some other L-OSL function.

However, this does not mean that all hypothetical cases of reduplicant modifications are C-OSL. Such processes can be C-Seq or higher if the composition of a truncation or identity function with the modification function is Seq or higher. For example, if complex onset reduction in the reduplicant deleted the first consonant: $mag\text{-}trabaho \rightarrow mag\text{-}ra\sim trabaha$, this process would be C-Seq and not C-OSL. In the first copy, the truncation function would generate $mag\{,trabah\}_r$, while the modification function would generate $mag\{,tra$. Deleting only the root-initial consonant t if it precedes a consonant is not OSL because we need finite lookahead on the input. Interestingly, this type of cluster reduction is argued to be unattested in reduplication (Zukoff 2017, 25). This may either be an accidental gap or evidence that reduplication modification must be C-OSL. To our knowledge, there is no typological survey of attested reduplicant modifications to settle this.

6.4.3 Phonological changes to or across both copies

Phonological changes may likewise affect both copies or apply across the boundary between the copies (22). Some involve a phonological process which is productive in the language (22a), others involve a phonological process which is not found anywhere else in the language outside of reduplication (22b). The former set of cases are often called

Table 9: C-OSL treatment for phonological changes to the reduplicant

	Papago (21a)	Tagalog (21b)	Turkish (21c)
Input x	bana	$mag\{,trabaho\}_r$	kitap
Components	$M(Tr(x)) \cdot ID(x)$	$M(Tr(x)) \cdot R(x)$	$ID(x) \cdot M(ID(x))$
Innermost function	ba na	$mag\{,tra bah\}_r$	kitap
Composition	baa	$mag\{,t \# a$	mitap
Concatenation	baa \sim bana	$mag\text{-}ta \sim mag\{,trabaho\}_r$	kitap \sim mitap
Subclass	C-L-OSL	C-L-OSL	C-L-OSL

normal application of phonological rules, while the latter are *juncture effects* which are morpheme-specific phonological processes.

(22) *Phonological modifications across the two copies*

a. **Normal application of nasal substitution**

Balangao (McCarthy and Prince 1995, 85)

- i. /maN + tagtag/ → [ma + nagtag] ‘running’
- ii. /maN-RED + tagtag/ → [ma + nagta~tagtag], ‘running’
* [ma + nagta~nagtag] everywhere’

b. **Phonology across the boundary (*juncture effects*)**

Dakota (Inkelas and Zoll 2005, 101)

- i. /skokpá → o-skókpa~kpa ‘to be scooped out’
- ii. /čap/ → čap~čap-a ‘trot’
- iii. /žat/ → žag~žat-a ‘curved’

In (22a), the prefix *maN-* can trigger reduplication of the root/stem. Nasal substitution combines the prefix’s nasal with an adjacent voiceless consonant into a single nasal that has the place of articulation of the consonant. Nasal substitution applies only to the segment next to the prefix, regardless of whether that consonant is part of the reduplicant or not. In (22b), the final syllable of the root is copied and placed at the left edge of the input. If there are two coronals across the reduplicative boundary (b), then the first coronal becomes dorsal. The final /a/ is epenthesized.

We likewise find phonological processes or rules interacting *differently* in the context of reduplication. For example in *Madurese*, there is a phonological process of nasal spread in which nasality is spread from nasals onto sequences of glides and vowels (23a). Reduplication copies the final CVC and places it at the beginning of the output (23b). If a vowel in the base is nasalized by a nasal, its nasality will transfer to the reduplicant as well. Because the reduplicant does not contain any nasals to trigger nasal spread, nasal spread in the reduplicant is treated as an over-application of the phonological process of nasal spread.

(23) **Over-application of nasal spread**

Madurese (McCarthy and Prince 1995, 30; Cohn 1993, 358)

- a. /neyat/ → [nẽỹāt] ‘intention
- b. [ỹāt~nẽỹāt] ‘intentions’

Traditionally, these cases can be thought as a composition of a morphological rule of reduplication (C-OSL) and a phonological rule (that is independently ISL, OSL, or Sequential) (Raimy 2000; Inkelas and Zoll 2005). If the morphological rule precedes the phonological rule, then we have normal application. If the morphological rule outputs reduplicant boundaries and precedes the phonological rule, then we have juncture effects. And, if the phonological rule precedes the morphological rule then we have over-application. Table 10 visualizes these three types of interactions as rule or function composition.

Table 10: Order of compositions for different reduplication-phonology interactions

	Normal application	Juncture effect	Over-application
Language	Balangao (22a)	Dakota (22b)	Madurese (23)
Input x	$\text{maN}\{\text{,tagtag}\}_r$	žat	neyat
Order of composition	1. Copy 2. Modify	1. Copy 2. Modify	1. Modify 2. Copy
Components	$M(L(x)) \cdot R(x)$	$M(L(x)) \cdot R(x)$	$L(M(x)) \cdot R(M(x))$
Innermost functions	$\text{maN}\{\text{,tagtag}\}_r \sim \text{maŋ}\{\text{,tagtag}\}_r$	žat \sim žata	něyāt
Outer function	$\text{maN}\{\text{,ŋagtag}\}_r \sim \{\text{,tagtag}\}_r$	žag \sim žata	něyāt \sim něyāt

Computationally, we can treat all these cases as composition of a C-OSL/C-Seq function for reduplication with an OSL/Seq function for phonology in either order. We conjecture C-OSL/C-Seq functions are not closed under composition, but we do not prove it. This means that composition may create a rotating 2-way FST.

Whether a case of normal application, juncture effect, or over-application is C-OSL, C-Seq, or higher depends on the complexity of the individual functions. In fact, the above three examples can be done with a C-Seq transducer. To illustrate, consider over-application in Madurese. Nasal spread is an L-OSL function $M(x)$. Reduplication is the concatenation of an R-OSL truncation function $L(x)$ and an L/R-OSL identity function $R(x)$. To generate overapplication, reduplication is instead the concatenation of two *modified* functions $L(M(x)) \cdot R(M(x))$. The first function is the composition of truncation over nasal spread. The composition of these two OSL rules of different directions is L-Seq because nasalization relies on deleted information

from the input.²¹ The second function is the composition of identity over nasal spread; this is L-OSL. Together, Madurese overapplication is C-L-Seq.

It is an open question if there are cases of normal application, juncture effects, and over-application which *cannot* be treated with a C-Seq formalization but require an unrestricted rotating 2-way FST. Solving this requires an in-depth knowledge of both the morphology and phonology of any such example (Inkelas and Zoll 2005).

Under-application and Back-copying In contrast to the over-application of phonological processes in reduplication, we likewise find cases of *under-application*. For example in Akan, velar consonants become palatalized before nonlow front vowels: /k,g/ → [tɕ, dʒ]/ _ /i,e/ as in (24a). Akan likewise has a process of initial-CV reduplication where the reduplicant V is a pre-specified non-low front vowel /ɪ/ (24b).²² However if the reduplicant C is a velar, it will not be palatalized before the reduplicant's non-low front vowel /ɪ/ (24c). Thus the rule under-applies. The velar will only palatalize if both copies of the velar in the reduplicant and base are preceded by a non-low front vowel (24d).

(24) **Under-application of palatalization in reduplication**

Akan (McCarthy and Prince 1995, 83-93)

(Schachter and Fromkin 1968, 89))

- | | |
|---------------------------------|-----------|
| a. /ke/ → [tɕe] | 'divide' |
| b. /si/ → [si~siʔ] | 'stand' |
| c. /kaʔ/ → [kɪ~kaʔ], *[tɕɪ~kaʔ] | 'bite' |
| d. /ge/ → [dʒɪ~dʒe] | 'receive' |

Cases of apparent under-application or over-application in reduplication are termed *opacity effects* (cf. the transparency of normal application (22a)). They are often understood as being caused by a need to maintain identity between the two copies that reduplication cre-

²¹ As with infixal reduplication (Section 6.3.2), the C-Seq transducer needs finite look-ahead into the end boundary \times and this makes it not have the exact origin semantics that we want.

²² The reduplicant V in Akan gets its front/back features from vowel harmony. For illustration, we represent it simply as /ɪ/.

ated (Wilbur 1973; McCarthy and Prince 1995). A more drastic version of identity is back-copying whereby the reduplicant undergoes some phonological rule, and then the effects of this rule are transferred onto the base. It is reported that in Malay, nasality spreads from a nasal consonant onto a sequence of vowels. Nasality can spread over glides and /h/. Plurality is marked by total reduplication. If nasal spread applies across the two copies, nasality will transfer onto both copies.

(25) **Back-copying of nasal spread**

Malay (McCarthy and Prince 1995, 85)

/hamə/ → [hãmẽ~hãmẽ], *[hamẽ~hãmẽ] ‘germ’ → ‘germs’

These opacity effects are controversial both theoretically and empirically (Inkelas and Zoll 2005; Samuels 2010; Kiparsky 2010; McCarthy *et al.* 2012). Many cases of under-application have been re-analyzed as either unproductive (McCarthy *et al.* 2012) or due to morpheme-specific rules (Inkelas and Zoll 2005).²³ In fact, Akan palatalization (24) is the classical case study on under-application but it is likely a synchronically unproductive and fossilized rule (Silverman 2002; Adomako 2018). Empirically, there have been little if any convincing cases of back-copying (Bruening 1997) and some are arguably due to morphological factors outside of reduplication (McLaughlin 2005). The Malay data itself has not been successively reproduced (Kiparsky 2010).

Because under-application and back-copying have weak empirical backing, there is a limited attested typology of these processes. It is thus unclear whether we can make any computational generalizations about them. But putting aside these empirical problems, Akan under-application can be modeled with a C-Seq function which uses finite lookahead on the input. It is the concatenation of a modified truncation function and a modified identity function. The first function truncates the input $C_1V_2\Sigma^*$ to C_1i and applies palatalization if V_2 is /i,e/. The second function applies palatalization to the input.

Malay back-copying is not C-Seq. This is because nasalization requires unbounded lookahead on the input. The function requires an

²³ An exception is Tonkawa (Gouskova 2007) which is arguably a bona fide case of under-application.

unrestricted rotating transducer with three passes over the input.²⁴ In the first pass, we output nothing but we check if the input ends in a $N(V+G)^*$ sequence where G stands for glides and $/h/$: $hamə$. If yes, the second pass applies nasal spread starting from the first segment: $hamə \sim \tilde{h}āmə$. The third pass does the same: $hamə \sim \tilde{h}āmə \sim \tilde{\tilde{h}}āmə$.

Overview of the typology summary

6.5

To summarize, we cataloged a wide variety of attested reduplicative patterns. All of it can be computed with deterministic 2-way FSTs. Most common and uncommon types of reduplication can be computed with the subclass of C-OSL functions, including total reduplication (12a), common partial reduplication patterns (13), triplication (14a), suffixal reduplication (14b), non-local reduplication (14c), and abbreviated reduplication (14d). Subconstituent reduplication is likewise C-OSL if the relevant morphological (15) or prosodic boundaries (16) are present in the input. In fact, of the 138 reduplicative processes in RedTyp (§5.2), 121 (87%) are C-OSL.²⁵

We analyzed the typology in terms of generating the right origin semantics. To do so, some less common types of reduplication are C-Seq or higher. This is largely because of the need for finite lookahead on the input. Some but not all types of infixal or non-contiguous reduplication are C-OSL (Section 6.3.1) and some are C-Seq (Section 6.3.2). In the latter case, generating the right origin semantics can require full 2-way FSTs because of the need for finite lookahead. Some cases like iterative reduplication (19a) are C-OSL if the input is at most bisyllabic; otherwise generating the right origin semantics needs an unrestricted 2-way D-FST. Minimality requirements (19b) likewise require

²⁴ Malay back-copying can likewise be treated as the composition of a C-OSL function for triplication $hamə \sim hamə \sim hamə$, followed by an OSL function for nasal spread $hamə \sim \tilde{h}āmə \sim \tilde{\tilde{h}}āmə$, followed by an OSL function that deletes everything before the first \sim boundary $\tilde{h}āmə \sim \tilde{\tilde{h}}āmə$. This analysis is inspired by Reiss and Simpson (2009).

²⁵ Although the cross-linguistic typology on reduplication is overwhelmingly C-OSL, our numbers from RedTyp do not mean that we estimate that 13% of the cross-linguistic typology of reduplicative processes is not C-OSL. RedTyp likely under-represents cases of opacity. Such cases can be non-C-OSL.

full 2-way D-FSTs. When reduplication interacts with phonological processes, the computation can range anywhere from C-OSL to full 2-way D-FSTs depending on the individual phonological process and the order of function composition. We suspect the finite lookahead in these cases may be resolved with more sophisticated representations and logical transductions (Dolatian 2020).

This concludes the section on how various subclasses of 2-way D-FSTs map to certain divisions in the reduplicative typology. The above cases are representative of common and uncommon reduplicative processes. There are other subtle variations for reduplication in natural language, such as cases of allomorphy (Spaelti 1997), or multiple reduplicants (Urbanczyk 1999, 2001; Fitzpatrick and Nevins 2004; Fitzpatrick 2006), among others. We will not discuss these cases because a full typology is beyond the scope of this paper. However, virtually all attested reduplicative processes can be modeled with 2-way FSTs. Fitting the *entire* attested typology into the right subclasses is a fruitful research direction. The next section looks at cases where 2-way D-FSTs arguably over-generate or under-generate the typology, even with these well-defined subclasses.

6.6 *Issues in over- and under-generation*

Here, we address the questions whether and how 2-way D-FSTs under- and over-generate reduplicative processes.

6.6.1 Over-generation with 2-way D-FSTs

One way to interpret the contribution we have made is that we are advocating the following hypothesis:

- (26) **2-way Hypothesis:** Reduplication is anything that can be computed with 2-way D-FSTs.

This is not, in fact, a position we advocate. We think this hypothesis is false because it overgenerates in ways we consider linguistically bizarre. For example, 2-way D-FSTs can map words to their reverse ($w \mapsto w^r$) and to a copy of itself and its mirror image ($w \mapsto ww^r$). None

of these transformations are attested morphologically. Some overgeneration can be avoided by hypothesizing stronger computational properties; that is, focusing on *subclasses* of 2-way D-FSTs which cannot generate the above unattested patterns.

This leads to another hypothesis.

- (27) **C-OSL Hypothesis:** Reduplication is anything that can be computed with C-OSL 2-way D-FSTs.

The C-OSL hypothesis is well supported because it covers the *bulk* of reduplicative typology as shown in Section 6. Rarer reduplicative patterns require more powerful subclasses of 2-way D-FSTs.

Even this hypothesis can be said to suffer from overgeneration. For example, while this excludes the reversal and mirror image processes above, it permits total reduplication of a word up to some large natural number n ($w \mapsto w^n$), or partial reduplication up to some natural number n of segments.

Nonetheless, not all issues in overgeneration can be reduced to computation or computability. Some are certainly due to external factors.²⁶ To illustrate, total reduplication in most *spoken* languages creates at most two copies. The creation of three copies (= triplication) is relatively rare in spoken languages, e.g. Thao (Blust 2001). In sign languages, we find the reverse situation: creating two copies is rare but triplication is common, e.g. ASL (Wilbur 2005). The difference between sign and spoken reduplication is more likely due to modality and not to the computation.²⁷

Under-generation with 2-way D-FSTs

6.6.2

Non-computational factors can also help us understand apparent cases of under-generation. There are two cases we discuss here: abstract morphemic copying and reduplication with haplology. Both of these

²⁶ Like Potts and Pullum (2002, 375), “we are extremely sceptical of the idea that formalisms exist that correspond exactly to what linguists wish to say.”

²⁷ A similar point can be made for the role of pivot or anchor points in reduplication. Cross-linguistically, most reduplicative processes target specific positions in the word which are perceptually or psycho-linguistically more salient (Samuels 2010; Raimy 2009; Idsardi and Raimy 2008), e.g. the first syllable and not the third syllable. The choice of these pivots is likely functional, not computational.

can be explained as involving interactions between reduplication and other linguistic modules (the lexicon) or processes (filters).

Undergeneration of abstract morphemic copying Abstract morphemic copying is when the input to the copying mechanism is not a string of phonological segments but a more abstract morphological entity, i.e. a morpheme or morpho-syntactic feature bundle (Inkelas and Zoll 2005). This is in contrast to examples reviewed earlier, where the source of reduplication was a string of segments which may contain morpheme boundaries. Such a case occurs in Sye in Table 11.

In Sye, a stem may have multiple suppletive allomorphs used in different morphological contexts. For example, the abstract morpheme $\sqrt{\text{FALL}}$ in Table 11a has two allomorphs *amol* and *omol*, such that *amol* is used after future morpheme and certain other tense morphemes while *omol* is used elsewhere. As for reduplication, total reduplication is used to mark intensification (Table 11b). When total reduplication applies in a context that requires using one of the allomorphs, we have an allomorph mismatch between the two copies (Table 11c).

Table 11:
Abstract
morphemic
copying in Sye

Morphemes	a. / $\sqrt{\text{FALL}}$ /	b. / $\sqrt{\text{FALL}} + \text{RED}$ /	c. / $\text{FUT} - \sqrt{\text{FALL}} - \text{RED}$ /
Output	omol	omol~omol	<i>cw-amol~omol</i> * <i>cw-omol~omol</i> * <i>cw-amol~amol</i>
Gloss	'fall'	'fall all over'	'they will fall all over'

Inkelas and Zoll (2005) analyze Sye as involving morphological copying. The copies are not in phonological correspondence because they are different allomorphs of the same morpheme. What was copied was an abstract morpheme $\sqrt{\text{FALL}}$. Its two copies were later spelled-out as two different allomorphs. Inkelas and Zoll's (2005) analysis for Sye is controversial (Frampton 2009); but there are a few other languages which show that the reduplicant is copying an abstract morphological entity (Inkelas and Downing 2015a,b; Hyman *et al.* 2009).

Cases of morphological copying for *suppletive* roots can be modeled with a 2-way D-FST that copies an abstract pre-spelled-out morphological entity, e.g. a root morpheme $\sqrt{\text{FALL}}$ or a root index (Harley 2014) which can be represented as a finite string of symbols. This is followed by a 1-way FST that models spell-out such that it is equipped

with knowledge over what all the *finite* pairs of morphemes and their suppletive allomorphs are. We make the safe assumption that the number of morphemes in a language that show suppletion is finite.²⁸

Undergeneration of reduplication with haplology Another case of potential under-generation is when reduplication is affected by anti-homophony constraints or haplology, i.e. when reduplication is blocked because it would create a sequence of identical syllables or feet that is dis-preferred by speakers (Yip 1995; Nevins 2012).

For example in Kanuri (Moravcsik 1978, 313), total reduplication is used to form glossonyms (28b). However reduplication is blocked if it creates sequence of identical syllables/feet (28b).

(28) Reduplication and haplology in Kanuri

- | | | |
|----|--------------------|----------------------------------|
| a. | kanəmbu | ‘Kanembu tribe’ |
| | kanəmbu~kanəmbu | ‘language of the Kanembu tribe’ |
| b. | karekare | ‘Karekare tribe’ |
| | *karekare~karekare | ‘language of the Karekare tribe’ |

A 2-way D-FST can encode this requirement that the input must not itself be a sequence of identical syllables or feet. However that would require the 2-way D-FST to know what all the finitely possible sequences of syllables and feet are in the language.

Two alternatives to this solution are possible. One is using a copy-and-filter mechanism (Golston 1995). The 2-way D-FST would handle the copying. The output of the copying process would be fed to a phonological system which would filter out any homophonous sequences of syllables or feet. The other alternative is to argue that the Kanuri input stems which contain a repeated sequence of symbols /karekare/ are underlyingly already reduplicated via lexical reduplication /kare + RED/. Such arguments have been brought up for superficially similar haplology effects in Manam (Buckley 1997). From this approach, there is then no haplology problem for 2-way D-FSTs.

In sum, although virtually the entire typology of reduplication can be modeled with 2-way D-FSTs, there are complications if one

²⁸ An FST which handles spell-out would resemble the lexical transducer used in the `xfst` finite-state package (Beesley and Karttunen 2003).

wishes to model the full interface of reduplicative morphology with other systems. However, as a reviewer point out, it may not be desirable, from a linguistic perspective at least, to model the interfaces in this way. It is disputable whether complications occurring at the interface of morphology with syntax or phonology should be addressed within an FST that is intended to account for the computational complexity of the morphology itself. On the other side of the coin, some conceptual problems arise with over-generation of 2-way D-FSTs because of the power that they require to handle copying in the first place.

CONCLUSION

The present study has taken a step in formalizing the wide typology of reduplicative processes in formal-language theoretic terms. We showed that 2-way D-FSTs, which are an understudied type of finite-state transducer, can easily model reduplication because they can reread their input multiple times in multiple directions. Computationally, this means that *recognizing* whether strings belong to the copy language $\{ww \mid w \in \Sigma^*\}$ (so for any $w \in \Sigma^*$ determining whether there is a $v \in \Sigma^*$ such that $w = vv$) is a harder problem than the one that takes any $w \in \Sigma^*$ as input and returns ww as output (*copying*). Reduplication studied as recognition is computationally more complex than reduplication studied as copying.

In addition to modeling reduplicative morphology as copying, 2-way D-FSTs do not suffer from state explosion nor do they assume finite bounds on the input, unlike 1-way FSTs. In terms of strong generative capacity, 2-way FSTs actively copy segments instead of memorizing segments. A diagnostic for copying vs. remembering is the origin semantics of the function. This article also presented the RedTyp database, which provides concrete examples of 2-way DFSTs modeling a range of cross-linguistic reduplicative morphemes.

Furthermore, we showed that the typology of reduplication can be modeled with subclasses of 2-way FSTs that are essentially defined as concatenations of simple subclasses of 1-way FSTs. Thus, our work

showed the role of computational subclasses in carving out the generative capacity of morphological processes, whether reduplicative or not. To give more context, most morphological processes can be computed by 1-way finite state automata and transducers (Koskenniemi 1983; Beesley and Karttunen 2003). In fact, substantially less expressive subregular classes are capable of computing most of these morphological processes (Aksënova *et al.* 2016; Chandlee 2017). So far, these subclasses have been identified based on considerations of locality (ISL, OSL) and determinism (Seq, sequentiality). At first, reduplication looks like an outlier in that it requires the more expressive generative capacity of 2-way transducers. However, even within this larger class of 2-way FSTs, we argued that reduplication only needs certain subclasses which are also based on the same considerations of locality (C-OSL) and sequentiality (C-Seq). These subclasses reinforce the role of locality and determinism as general constraints in linguistic processes (cf. Heinz 2018).

Having showcased the utility of 2-way D-FSTs for modeling reduplication, we conclude with three avenues of future research.

First, we have approached reduplication from the perspective of morphological generation. Given an input *buku*, a 2-way D-FST can generate the output *buku~buku* easily. On the other hand, it is an open question as to how to do morphological *analysis* with 2-way FSTs to get the inverse relation of *buku~buku* \rightarrow *buku*. As a class, deterministic 2-way FSTs are not invertible. We are currently developing algorithms for inverting the subclasses (C-OSL, C-Seq) that we have set up.

A second area of research is the integration of 2-way FSTs into natural language processing. This obviously has many aspects. A first step may be the integration of 2-way FSTs into existing platforms such as *xfst* (Beesley and Karttunen 2003), *open-fst* (Allauzen *et al.* 2007), *foma* (Hulden 2009b), and *pynini* (Gorman 2016).²⁹

A third promising area of research is developing learning models based on the computational models that we proposed here. One approach builds on Chandlee *et al.*'s 2015 learning results of OSL func-

²⁹In fact, the team behind Thrax (Tai *et al.* 2011) have recently been exploring the use of multi-pushdown transducers (MPDT) to generate reduplication (Richard Sproat, p.c.). An open question is comparing the generative capacity of MPDTs and 2-way FSTs.

tions (Dolatian and Heinz 2018a). Another approach probes the learnability of reduplicative patterns with neural networks (Nelson *et al.* 2020).

REFERENCES

- Kwasi ADOMAKO (2018), Velar palatalization in Akan: A reconsideration, *Journal of West African Languages*, 45(2).
- Alfred V. AHO, John E. HOPCROFT, and Jeffrey D. ULLMAN (1969), A general theory of translation, *Mathematical Systems Theory*, 3(3):193–221.
- Alëna AKSËNOVA, Thomas GRAF, and Sedigheh MORADI (2016), Morphotactics as tier-based strictly local dependencies, in *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pp. 121–130.
- Daniel M. ALBRO (2000), Taking Primitive Optimality Theory beyond the finite state, in Jason EISNER, Lauri KARTTUNEN, and Alain THÉRIAULT, editors, *Finite-State Phonology: Proceedings of the 5th Workshop of SIGHON*, pp. 57–67, Luxembourg, <http://aclanthology.coli.uni-saarland.de/pdf/W/W00/W00-1806.pdf>.
- Daniel M. ALBRO (2005), *Studies in Computational Optimality Theory, with Special Reference to the Phonological System of Malagasy*, Ph.D. thesis, University of California, Los Angeles.
- Raquel G. ALHAMA (2017), *Computational Modelling of Artificial Language Learning: Retention, Recognition & Recurrence*, Ph.D. thesis, Universiteit van Amsterdam.
- Raquel G. ALHAMA and Willem ZUIDEMA (2019), A review of computational models of basic rule learning: The neural-symbolic debate and beyond, *Psychonomic Bulletin & Review*, 26(4):1–21.
- Cyril ALLAUZEN, Michael RILEY, Johan SCHALKWYK, Wojciech SKUT, and Mehryar MOHRI (2007), OpenFst: A general and efficient weighted finite-state transducer library, in Jan HOLUB and Jan ŽĎÁREK, editors, *Implementation and Application of Automata*, pp. 11–23, Springer, Berlin, Heidelberg.
- Rajeev ALUR (2010), Expressiveness of streaming string transducers, in *Proceedings of the 30th Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 8, p. 1–12, doi:10.4230/LIPIcs.FSTTCS.2010.1.

Rajeev ALUR and Jyotirmoy V. DESHMUKH (2011), Nondeterministic streaming string transducers, in Luca ACETO, Monika HENZINGER, and Jiří SGALL, editors, *Automata, Languages and Programming*, pp. 1–20, Springer, Berlin, Heidelberg, ISBN 978-3-642-22012-8.

Rajeev ALUR, Adam FREILICH, and Mukund RAGHOTHAMAN (2014), Regular combinators for string transformations, in *Proceedings of the Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), Vienna, Austria, CSL-LICS '14*, pp. 9:1–9:10, Association for Computing Machinery, New York, NY, USA, ISBN 978-1-4503-2886-9, doi:10.1145/2603088.2603151.

Rajeev ALUR and Pavol ČERNÝ (2011), Streaming transducers for algorithmic verification of single-pass list-processing programs, in *Proceedings of the 38th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, Austin, Texas, USA, POPL '11*, pp. 599–610, Association for Computing Machinery, New York, NY, USA, ISBN 978-1-4503-0490-0, doi:10.1145/1926385.1926454.

Qatherine ANDAN, Outi BAT-EL, Diane BRENTARI, and Iris BERENT (2018), ANCHORING is amodal: Evidence from a signed language, *Cognition*, 180:279–283.

Stephen R. ANDERSON (1992), *A-morphous morphology*, volume 62 of *Cambridge Studies in Linguistics*, Cambridge University Press, Cambridge.

Mark ARONOFF (1988), Head operations and strata in reduplication: A linear treatment, in Geert BOOIJ and Jaap VAN MARLE, editors, *Yearbook of Morphology*, volume 1, pp. 1–15, Foris, Dordrecht.

Félix BASCHENIS, Olivier GAUWIN, Anca MUSCHOLL, and Gabriele PUPPIS (2015), One-way definability of sweeping transducers, in *35th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'15)*, Bangalore, India, <https://hal.archives-ouvertes.fr/hal-01219509>.

Félix BASCHENIS, Olivier GAUWIN, Anca MUSCHOLL, and Gabriele PUPPIS (2016), Minimizing resources of sweeping and streaming string transducers, in Ioannis CHATZIGIANNAKIS, Michael MITZENMACHER, Yuval RABANI, and Davide SANGIORGI, editors, *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 114:1–114:14, Schloss Dagstuhl-Leibniz-Zentrum für Informatik, Dagstuhl, Germany, ISBN 978-3-95977-013-2, ISSN 1868-8969, doi:10.4230/LIPIcs.ICALP.2016.114.

Félix BASCHENIS, Olivier GAUWIN, Anca MUSCHOLL, and Gabriele PUPPIS (2017), Untwisting two-way transducers in elementary time, in *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik*,

Iceland, June 20-23, 2017, pp. 1–12, ISBN 978-1-5090-3018-7,
doi:10.1109/LICS.2017.8005138.

Félix BASCHENIS, Olivier GAUWIN, Anca MUSCHOLL, and Gabriele PUPPIS (2018), One-way definability of two-way word transducers, *Logical Methods in Computer Science*, 14.

Kenneth BEESLEY and Lauri KARTTUNEN (2000), Finite-state non-concatenative morphotactics, in *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics, Hong Kong, ACL '00*, pp. 191–198, Association for Computational Linguistics, Stroudsburg, PA, USA,
doi:10.3115/1075218.1075243.

Kenneth BEESLEY and Lauri KARTTUNEN (2003), *Finite-State Morphology: Xerox Tools and Techniques*, CSLI Publications, Stanford, CA.

Iris BERENT, Outi BAT-EL, Diane BRENTARI, Amanda DUPUIS, and Vered VAKNIN-NUSBAUM (2016), The double identity of linguistic doubling, *Proceedings of the National Academy of Sciences*, 113(48):13702–13707.

Iris BERENT, Outi BAT-EL, and Vered VAKNIN-NUSBAUM (2017), The double identity of doubling: Evidence for the phonology-morphology split, *Cognition*, 161:117–128.

Iris BERENT, Amanda DUPUIS, and Diane BRENTARI (2014), Phonological reduplication in sign language: Rules rule, *Frontiers in Psychology*, 5:560.

Steven BIRD and T. Mark ELLISON (1994), One-level phonology: Autosegmental representations and rules as finite automata, *Computational Linguistics*, 20(1):55–90.

Robert A. BLUST (2001), Thao triplication, *Oceanic Linguistics*, 40(2):324–335.

Mikołaj BOJAŃCZYK (2014), Transducers with origin information, in Javier ESPARZA, Pierre FRAIGNIAUD, Thore HUSFELDT, and Elias KOUTSOPIAS, editors, *Automata, Languages, and Programming*, pp. 26–37, Springer, Berlin, Heidelberg.

Mikołaj BOJAŃCZYK, Laure DAVIAUD, Bruno GUILLON, and Vincent PENELLE (2017), Which classes of origin graphs are generated by transducers, in Ioannis CHATZIGIANNAKIS, Piotr INDYK, Fabian KUHN, and Anca MUSCHOLL, editors, *44th International Colloquium on Automata, Languages, and Programming (ICALP 2017)*, volume 80 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 114:1–114:13, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany, ISBN 978-3-95977-041-5, ISSN 1868-8969,
doi:10.4230/LIPIcs.ICALP.2017.114.

Olivier BONAMI and Berthold CRYSMANN (2013), Morphotactics in an information-based model of realisational morphology, in Stefan MÜLLER, editor, *Proceedings of HPSG 2013*, pp. 27–47, CSLI Publications, Stanford, CA.

- Olivier BONAMI and Berthold CRYSMANN (2016), The role of morphology in constraint-based lexicalist grammars, in Andrew HIPPLISLEY and Gregory T. STUMP, editors, *The Cambridge Handbook of Morphology*, p. 609–656, Cambridge University Press, Cambridge.
- Ellen BROSELOW and John MCCARTHY (1983), A theory of internal reduplication, *The Linguistic Review*, 3(1):25–88.
- Jason BROWN (2017), Non-adjacent reduplication requires spellout in parallel, *Natural Language & Linguistic Theory*, 35(4):1–23.
- Benjamin BRUENING (1997), Abkhaz mabkhaz: M-reduplication in Abkhaz, weightless syllables, and base-reduplicant correspondence, in Benjamin BRUENING, Yoonjung KANG, and Martha MCGINNIS, editors, *PF: Papers at the Interface*, volume 30, MIT Working Papers in Linguistics, Cambridge, MA.
- Eugene BUCKLEY (1997), Integrity and correspondence in Manam double reduplication, in *Proceedings of NELS*, volume 28, pp. 59–67.
- Gabriela CABALLERO (2006), “Templatic backcopying” in Guarijio abbreviated reduplication, *Morphology*, 16(2):273–289.
- Jane CHANDLEE (2014), *Strictly local phonological processes*, Ph.D. thesis, University of Delaware, Newark, DE.
- Jane CHANDLEE (2017), Computational locality in morphological maps, *Morphology*, 27(4):1–43.
- Jane CHANDLEE, Angeliki ATHANASOPOULOU, and Jeffrey HEINZ (2012), Evidence for classifying metathesis patterns as subsequential, in Jaehoon CHOI, E. Alan HOGUE, Jeffrey PUNSKE, Deniz TAT, Jessamyn SCHERTZ, and Alex TRUEMAN, editors, *The Proceedings of the 29th West Coast Conference on Formal Linguistics*, pp. 303–309, Cascillida Press, Somerville, MA.
- Jane CHANDLEE, Rémi EYRAUD, and Jeffrey HEINZ (2014), Learning strictly local subsequential functions, *Transactions of the Association for Computational Linguistics*, 2:491–503, <http://aclweb.org/anthology/Q14-1038>.
- Jane CHANDLEE, Rémi EYRAUD, and Jeffrey HEINZ (2015), Output strictly local functions, in *14th Meeting on the Mathematics of Language*, pp. 112–125.
- Jane CHANDLEE and Jeffrey HEINZ (2012), Bounded copying is subsequential: Implications for metathesis and reduplication, in *Proceedings of the 12th Meeting of the ACL Special Interest Group on Computational Morphology and Phonology*, SIGMORPHON '12, pp. 42–51, Association for Computational Linguistics, Montreal, Canada.
- Jane CHANDLEE and Jeffrey HEINZ (2018), Strict locality and phonological maps, *Linguistic Inquiry*, 49(1):23–60.
- Jane CHANDLEE, Jeffrey HEINZ, and Adam JARDINE (2018), Input strictly local opaque maps, *Phonology*, 35(2):171–205.

Christian CHOFFRUT (1977), Une caractérisation des fonctions séquentielles et des fonctions sous-séquentielles en tant que relations rationnelles, *Theoretical Computer Science*, 5(3):325–337, ISSN 0304-3975, doi:[https://doi.org/10.1016/0304-3975\(77\)90049-4](https://doi.org/10.1016/0304-3975(77)90049-4).

Michal P. CHYTIL and Vojtěch JÁKL (1977), Serial composition of 2-way finite-state transducers and simple programs on strings, in Arto SALOMAA and Magnus STEINBY, editors, *Automata, Languages and Programming*, pp. 135–147, Springer, Berlin, Heidelberg, ISBN 978-3-540-37305-6.

Alexander CLARK (2017), Computational learning of syntax, *Annual Review of Linguistics*, 3:107–123.

Alexander CLARK and Ryo YOSHINAKA (2012), Beyond semilinearity: Distributional learning of parallel multiple context-free grammars, in *International Conference on Grammatical Inference*, pp. 84–96.

Alexander CLARK and Ryo YOSHINAKA (2014), Distributional learning of parallel multiple context-free grammars, *Machine Learning*, 96(1-2):5–31.

Alexander CLARK and Ryo YOSHINAKA (2016), Distributional learning of context-free and multiple context-free grammars, in Jeffrey HEINZ and José M. SEMPERE, editors, *Topics in Grammatical Inference*, pp. 143–172, Springer, Berlin, Heidelberg.

Yael COHEN-SYGAL and Shuly WINTNER (2006), Finite-state registered automata for non-concatenative morphology, *Computational Linguistics*, 32(1):49–82.

Abigail C. COHN (1989), Stress in Indonesian and bracketing paradoxes, *Natural language & linguistic theory*, 7(2):167–216.

Abigail C. COHN (1993), The status of nasalized continuants, in Marie K. HUFFMAN and Rena A. KRAKOW, editors, *Nasals, Nasalization, and the Velum*, volume 5 of *Phonetics and Phonology*, pp. 329–367, Academic Press, Inc., San Diego, CA.

Bruno COURCELLE and Joost ENGELFRIET (2012), *Graph Structure and Monadic Second-Order Logic, a Language Theoretic Approach*, Cambridge University Press, Cambridge.

Berthold CRYSMANN (2017), Reduplication in a computational HPSG of Hausa, *Morphology*, 27(4):527–561.

Berthold CRYSMANN and Olivier BONAMI (2016), Variable morphotactics in information-based morphology, *Journal of Linguistics*, 52(2):311–374.

Karel CULIK and Juhani KARHUMÄKI (1986), The equivalence of finite valued transducers (on HDTOL languages) is decidable, *Theoretical Computer Science*, 47:71–84, ISSN 0304-3975, doi:[https://doi.org/10.1016/0304-3975\(86\)90134-9](https://doi.org/10.1016/0304-3975(86)90134-9).

- Christopher CULY (1985), The complexity of the vocabulary of Bambara, *Linguistics and Philosophy*, 8:345–351.
- Vrunda DAVE, Paul GASTIN, and Shankara Narayanan KRISHNA (2018), Regular transducer expressions for regular transformations, in *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '18*, pp. 315–324, Association for Computing Machinery, New York, NY, USA, ISBN 978-1-4503-5583-4, doi:10.1145/3209108.3209182.
- Hossep DOLATIAN (2020), *Computational Locality of Cyclic Phonology in Armenian*, Ph.D. thesis, Stony Brook University.
- Hossep DOLATIAN and Jeffrey HEINZ (2018a), Learning reduplication with 2-way finite-state transducers, in Olgierd UNOLD, Witold DYRKA, , and Wojciech WIECZOREK, editors, *Proceedings of Machine Learning Research: International Conference on Grammatical Inference*, volume 93 of *Proceedings of Machine Learning Research*, pp. 67–80, Wrocław, Poland.
- Hossep DOLATIAN and Jeffrey HEINZ (2018b), Modeling reduplication with 2-way finite-state transducers, in *Proceedings of the 15th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, Association for Computational Linguistics, Brussels, Belgium.
- Hossep DOLATIAN and Jeffrey HEINZ (2019a), RedTyp: a database of reduplication with computational models, in *Proceedings of the Society for Computation in Linguistics*, volume 2, article 3.
- Hossep DOLATIAN and Jeffrey HEINZ (2019b), Reduplication with finite-state technology, in *Proceedings of the 53rd Annual Meeting of the Chicago Linguistics Society*, Chicago Linguistics Society, Chicago.
- Laura J. DOWNING (1998), Prosodic misalignment and reduplication, in Geert BOOIJ and Jaap VAN MARLE, editors, *Yearbook of Morphology 1997*, pp. 83–120, Kluwer Academic Publishers, Dordrecht.
- Laura J. DOWNING (2000), Morphological and prosodic constraints on Kinande verbal reduplication, *Phonology*, 17(01):1–38.
- Laura J. DOWNING (2003), Compounding and tonal non-transfer in Bantu languages, *Phonology*, 20(1):1–42.
- Laura J. DOWNING (2006), *Canonical Forms in Prosodic Morphology*, number 12 in Oxford studies in Theoretical Linguistics, Oxford University Press, Oxford.
- Calvin C. ELGOT and Jorge E. MEZEI (1965), On relations defined by generalized finite automata, *IBM Journal of Research and development*, 9(1):47–68.
- Joost ENGELFRIET and Hendrik Jan HOOGEBOOM (2001), MSO definable string transductions and two-way finite-state transducers, *Transactions of the Association for Computational Linguistics*, 2(2):216–254, ISSN 1529-3785, doi:10.1145/371316.371512.

- Emmanuel FILIOT and Pierre-Alain REYNIER (2016), Transducers, logic and algebra for functions of finite words, *ACM SIGLOG News*, 3(3):4–19, ISSN 2372-3491, doi:10.1145/2984450.2984453.
- Justin FITZPATRICK (2006), Sources of Multiple Reduplication in Salish and Beyond, *Studies in Salishan 7*, pp. 211–240.
- Justin FITZPATRICK and Andrew NEVINS (2004), Linearizing nested and overlapping precedence in multiple reduplication, in *University of Pennsylvania Working Papers in Linguistics*, pp. 75–88.
- Jennifer FITZPATRICK-COLE (1994), *The Prosodic Domain Hierarchy in Reduplication*, Ph.D. thesis, Stanford University, Stanford, CA.
- John FRAMPTON (2009), *Distributed Reduplication*, MIT Press, Cambridge.
- Diamandis GAFOS (1998), A-templatic reduplication, *Linguistic Inquiry*, 29(3):515–527.
- Brian GAINOR, Regine LAI, and Jeffrey HEINZ (2012), Computational characterizations of vowel harmony patterns and pathologies, in Jaehoon CHOI, E. Alan HOGUE, Jeffrey PUNSKE, Deniz TAT, Jessamyn SCHERTZ, and Alex TRUEMAN, editors, *The Proceedings of the 29th West Coast Conference on Formal Linguistics*, pp. 63–71, Cascillida Press, Somerville, MA.
- Pedro GARCIA, Enrique VIDAL, and José ONCINA (1990), Learning locally testable languages in the strict sense, in *Proceedings of the Workshop on Algorithmic Learning Theory*, pp. 325–338.
- Gerald GAZDAR and Geoffrey K PULLUM (1985), Computationally relevant properties of natural languages and their grammars, *New generation computing*, 3:273–306.
- Jila GHOMESHI, Ray JACKENDOFF, Nicole ROSEN, and Kevin RUSSELL (2004), Contrastive focus reduplication in English (the salad-salad paper), *Natural Language & Linguistic Theory*, 22(2):307–357.
- Chris GOLSTON (1995), Syntax outranks phonology: Evidence from Ancient Greek, *Phonology*, 12(3):343–368.
- Kyle GORMAN (2016), Pynini: A Python library for weighted finite-state grammar compilation, in *Proceedings of the SIGFSM Workshop on Statistical NLP and Weighted Automata*, pp. 75–80, Association for Computational Linguistics, Berlin, Germany, doi:10.18653/v1/W16-2409, <http://www.aclweb.org/anthology/W16-2409>.
- Maria GOUSKOVA (2007), The reduplicative template in Tonkawa, *Phonology*, 24(3):367–396.
- Jiatao GU, Zhengdong LU, Hang LI, and Victor O.K. LI (2016), Incorporating copying mechanism in sequence-to-sequence learning, in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1631–1640, Association for Computational Linguistics, Berlin, Germany.

- Heidi HARLEY (2014), On the identity of roots, *Theoretical linguistics*, 40(3/4):225–276.
- Jason D. HAUGEN (2009), What is the base for reduplication?, *Linguistic Inquiry*, 40(3):505–514.
- Jeffrey HEINZ (2007), *The Inductive Learning of Phonotactic Patterns*, Ph.D. thesis, University of California, Los Angeles.
- Jeffrey HEINZ (2018), The computational nature of phonological generalizations, in Larry HYMAN and Frans PLANK, editors, *Phonological Typology*, Phonetics and Phonology, chapter 5, pp. 126–195, Mouton de Gruyter, Berlin.
- Jeffrey HEINZ and William IDSARDI (2013), What complexity differences reveal about domains in language, *Topics in Cognitive Science*, 5(1):111–131.
- Jeffrey HEINZ and Regine LAI (2013), Vowel harmony and subsequentiality, in Andras KORNAI and Marco KUHLMANN, editors, *Proceedings of the 13th Meeting on the Mathematics of Language (MoL 13)*, pp. 52–63, Association for Computational Linguistics, Sofia, Bulgaria, <http://www.aclweb.org/anthology/W13-3006>.
- John E. HOPCROFT and Jeffrey D. ULLMAN (1969), *Formal Languages and their Relation to Automata*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA.
- Mans HULDEN (2006), Finite-state syllabification, in Anssi YLI-JYRÄ, Lauri KARTTUNEN, and Juhani KARHUMÄKI, editors, *Finite-State Methods and Natural Language Processing. FSMNLP 2005. Lecture Notes in Computer Science*, volume 4002, Springer, Berlin/Heidelberg.
- Mans HULDEN (2009a), *Finite-State Machine Construction Methods and Algorithms for Phonology and Morphology*, Ph.D. thesis, University of Arizona, Tucson, AZ.
- Mans HULDEN (2009b), Foma: a finite-state compiler and library, in *Proceedings of the Demonstrations Session at EAACL 2009*, pp. 29–32, Association for Computational Linguistics, Athens, Greece, <http://www.aclweb.org/anthology/E09-2008>.
- Mans HULDEN and Shannon T. BISCHOFF (2009), A simple formalism for capturing reduplication in finite-state morphology, in Jakub PISKORSKI, Bruce WATSON, and Anssi YLI-JYRÄ, editors, *Proceedings of the 2009 conference on Finite-State Methods and Natural Language Processing: Post-proceedings of the 7th International Workshop FSMNLP 2008*, pp. 207–214, IOS Press, Amsterdam, ISBN 978-1-58603-975-2, <http://dl.acm.org/citation.cfm?id=1564035.1564059>.
- Bernhard HURCH, editor (2005), *Studies on Reduplication*, number 28 in Empirical Approaches to Language Typology, Walter de Gruyter, Berlin.
- Bernhard HURCH (2005 ff.), Graz database on reduplication, last accessed 10-26-2017 from <http://reduplication.uni-graz.at/redup/>.

- Bernhard HURCH and Veronika MATTES (2009), Introduction: diachrony and productivity of reduplication, *Morphology*, 19(2):107–112.
- Larry M. HYMAN (2009), The natural history of verb-stem reduplication in Bantu, *Morphology*, 19(2):177–206.
- Larry M. HYMAN, Sharon INKELAS, and Galen SIBANDA (2009), Morphosyntactic correspondence in Bantu reduplication, in Kristin HANSON and Sharon INKELAS, editors, *The Nature of the Word: Studies in Honor of Paul Kiparsky*, Current Studies in Linguistics, pp. 273–309, The MIT Press, Cambridge, MA.
- William IDSARDI and Eric RAIMY (2008), Reduplicative economy, in Bert VAUX and Andrew NEVINS, editors, *Rules, constraints, and phonological phenomena*, chapter 5, pp. 149–184, Oxford University Press, Oxford.
- Sharon INKELAS (2014), *The Interplay of Morphology and Phonology*, Oxford University Press, Oxford.
- Sharon INKELAS and Laura J. DOWNING (2015a), What is reduplication? Typology and analysis part 1/2: The typology of reduplication, *Language and Linguistics Compass*, 9(12):502–515.
- Sharon INKELAS and Laura J. DOWNING (2015b), What is Reduplication? Typology and analysis Part 2/2: The analysis of reduplication, *Language and Linguistics Compass*, 9(12):516–528.
- Sharon INKELAS and Cheryl ZOLL (2005), *Reduplication: Doubling in Morphology*, Cambridge University Press, Cambridge.
- Adam JARDINE (2016), Computationally, tone is different, *Phonology*, 33(2):247–283.
- C. Douglas JOHNSON (1972), *Formal Aspects of Phonological Description*, Mouton, The Hague.
- Ronald M. KAPLAN and Martin KAY (1994), Regular models of phonological rule systems, *Computational linguistics*, 20(3):331–378.
- Lauri KARTTUNEN (1983), KIMMO: A general morphological processor, in *Texas Linguistic Forum*, volume 22, pp. 163–186.
- Paul KIPARSKY (1982), Lexical morphology and phonology, in I.-S. YANG, editor, *Linguistics in the morning calm: Selected papers from SICOL-1981*, pp. 3–91, Hansin, Seoul.
- Paul KIPARSKY (2010), Reduplication in stratal OT, in Linda UYECHI and Lian Hee WEE, editors, *Reality Exploration and Discovery: Pattern Interaction in Language & Life*, pp. 125–142, CSLI Press, Stanford.
- Gregory Michael KOBELE (2006), *Generating Copies: An Investigation into Structural Identity in Language and Grammar*, Ph.D. thesis, University of California, Los Angeles.

- Kimmo KOSKENNIEMI (1983), *Two-level morphology: A general computational model for word-form recognition and production*, Ph.D. thesis, University of Helsinki.
- Kimmo KOSKENNIEMI (1984), A general computational model for word-form recognition and production, in *Proceedings of the 10th international conference on Computational Linguistics*, pp. 178–181, Association for Computational Linguistics.
- D. Terence LANGENDOEN (1981), The generative capacity of word-formation components, *Linguistic Inquiry*, 12(2):320–322.
- Jeffrey LIDZ (2001), Echo reduplication in Kannada and the theory of word-formation, *Linguistic review*, 18(4):375–394.
- Huan LUO (2017), Long-distance consonant agreement and subsequentiality, *Glossa: A journal of General Linguistics*, 2(1):1–25, doi:<http://doi.org/10.5334/gjgl.42>.
- Alexis MANASTER-RAMER (1986), Copying in natural languages, context-freeness, and queue grammars, in *Proceedings of the 24th annual meeting on Association for Computational Linguistics*, pp. 85–89, Association for Computational Linguistics.
- Alec MARANTZ (1982), Re reduplication, *Linguistic Inquiry*, 13(3):435–482.
- Gary F. MARCUS, Sugumaran VIJAYAN, S. Bandi RAO, and Peter M. VISHTON (1999), Rule learning by seven-month-old infants, *Science*, 283(5398):77–80.
- Veronika MATTES (2007), *Reduplication in Bikol*, Ph.D. thesis, University of Graz, Graz, Austria.
- John J. MCCARTHY, Wendell KIMPER, and Kevin MULLIN (2012), Reduplication in Harmonic Serialism, *Morphology*, 22(2):173–232.
- John J. MCCARTHY and Alan PRINCE (1994), The emergence of the unmarked: Optimality in prosodic morphology, in Mercé GONZÁLEZ, editor, *Proceedings of the North East Linguistic Society 24*, p. 333–79, Graduate Linguistic Student Association, University of Massachusetts, Amherst, MA.
- John J. MCCARTHY and Alan PRINCE (1995), Faithfulness and reduplicative identity, in Jill N. BECKMAN, Laura Walsh DICKEY, and Suzanne URBANCZYK, editors, *Papers in Optimality Theory*, Graduate Linguistic Student Association, University of Massachusetts, Amherst, MA.
- Fiona MCLAUGHLIN (2005), Reduplication and consonant mutation in the Northern Atlantic languages, in Hurch (2005), pp. 111–134.
- Robert MCNAUGHTON and Seymour A. PAPERT (1971), *Counter-free automata*, MIT Press, Cambridge, MA.
- Mehryar MOHRI (1997), Finite-state transducers in language and speech processing, *Computational Linguistics*, 23(2):269–311.

Edith MORAVCSIK (1978), Reduplicative constructions, in Joseph GREENBERG, editor, *Universals of Human Language*, volume 1, pp. 297–334, Stanford University Press, Stanford, California.

Ajit NARAYANAN and Lama HASHEM (1993), On abstract finite-state morphology, in *Proceedings of the Sixth Conference on European Chapter of the Association for Computational Linguistics, Utrecht, The Netherlands, EAACL '93*, pp. 297–304, Association for Computational Linguistics, Stroudsburg, PA, USA, ISBN 90-5434-014-2, doi:10.3115/976744.976779.

Mark-Jan NEDERHOF and Heiko VOGLER (2019), Regular transductions with MCFG input syntax, in *Proceedings of the 14th International Conference on Finite-State Methods and Natural Language Processing*, pp. 56–64, Association for Computational Linguistics, Dresden, Germany, <https://www.aclweb.org/anthology/W19-3109>.

Esa NELIMARKKA, Harri JÄPPINEN, and Aarno LEHTOLA (1984), Two-way finite automata and dependency grammar: A parsing method for inflectional free word order languages, in *Proceedings of the 10th international conference on Computational linguistics*, pp. 389–392, Association for Computational Linguistics.

Max NELSON, Hossep DOLATIAN, Jonathan RAWSKI, and Brandon PRICKETT (2020), Probing RNN encoder-decoder generalization of subregular functions using reduplication, in *Proceedings of the Society for Computation in Linguistics*, volume 3.

Nicole Alice NELSON (2003), *Asymmetric Anchoring*, Ph.D. thesis, Rutgers University, New Brunswick, NJ.

Andrew NEVINS (2004), What UG can and can't do to help the reduplication learner, in Aniko CSLRMAZ, Andrea GUALMINI, and Andrew NEVINS, editors, *MIT Working Papers in Linguistics 48*, pp. 113–126, MIT Department of Linguistics and Philosophy, Cambridge, MA.

Andrew NEVINS (2012), Haplological dissimilation at distinct stages of exponence, in Jochen TROMMER, editor, *The Morphology and Phonology of Exponence*, pp. 84–116, Oxford University Press, Oxford.

Andrew NEVINS and Bert VAUX (2003), Metalinguistic, shmetalinguistic: The phonology of shmreduplication, in *Proceedings from the Annual Meeting of the Chicago Linguistic Society*, volume 39, pp. 702–721, Chicago Linguistic Society, Chicago.

David ODDEN (1994), Adjacency parameters in phonology, *Language*, 70(2):289–330.

John J. OHALA, Joseph Paul STEMBERGER, and Marshall LEWIS (1986), Reduplication in Ewe: Morphological accommodation to phonological errors, *Phonology*, 3:151–160.

Amanda PAYNE (2014), Dissimilation as a subsequential process, in Jyoti IYER and Leland KUSMER, editors, *NELS 44: Proceedings of the 44th Meeting of the North East Linguistic Society*, volume 2, pp. 79–90, Graduate Linguistic Student Association, University of Massachusetts, Amherst, MA.

Amanda PAYNE (2017), All dissimilation is computationally subsequential, *Language: Phonological Analysis*, 93(4):e353–e371, doi:doi:10.1353/lan.2017.0076.

Christopher POTTS and Geoffrey K. PULLUM (2002), Model theory and the content of OT constraints, *Phonology*, 19(3):361–393.

Brandon PRICKETT, Aaron TRAYLOR, and Joe PATER (2018), Seq2Seq models with dropout can learn generalizable reduplication, in *Proceedings of the Fifteenth Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pp. 93–100.

Michael O. RABIN and Dana SCOTT (1959), Finite automata and their decision problems, *IBM Journal of Research and Development*, 3(2):114–125.

Eric RAIMY (2000), *The Phonology and Morphology of Reduplication*, Mouton de Gruyter, Berlin.

Eric RAIMY (2009), Deriving reduplicative templates in a modular fashion, in Eric RAIMY and Charles E. CAIRNS, editors, *Contemporary views on architecture and representations in phonology*, number 48 in Current Studies in Linguistics, pp. 383–404, MIT Press, Cambridge, MA.

Eric RAIMY (2011), Reduplication, in Marc VAN OOSTENDORP, Colin EWEN, Elizabeth HUME, and Keren RICE, editors, *The Blackwell Companion to Phonology*, volume 4, pp. 2383–2413, Wiley-Blackwell, Malden, MA.

Charles REISS and Marc SIMPSON (2009), Reduplication as projection, unpublished manuscript, Concordia University, Montréal.

Jason RIGGLE (2004), Nonlocal reduplication, in Kier MOULTON and Matthew WOLF, editors, *Proceedings of the 34th meeting of the North Eastern Linguistics Society*, Graduate Linguistic Student Association, University of Massachusetts, Amherst, MA.

Brian ROARK and Richard SPROAT (2007), *Computational Approaches to Morphology and Syntax*, Oxford University Press, Oxford.

James ROGERS and Geoffrey PULLUM (2011), Aural pattern recognition experiments and the subregular hierarchy, *Journal of Logic, Language and Information*, 20:329–342.

Carl RUBINO (2005), Reduplication: Form, function and distribution, in Hurch (2005), pp. 11–29.

Carl RUBINO (2013), *Reduplication*, Max Planck Institute for Evolutionary Anthropology, Leipzig, <http://wals.info/chapter/27>.

- Jesse SABA KIRCHNER (2010), *Minimal reduplication*, Ph.D. thesis, University of California, Santa Cruz.
- Jesse SABA KIRCHNER (2013), Minimal reduplication and reduplicative exponence, *Morphology*, 23(2):227–243.
- Bridget SAMUELS (2010), The topology of infixation and reduplication, *The Linguistic Review*, 27(2):131–176.
- Walter J. SAVITCH (1982), *Abstract machines and grammars*, Little Brown and Company, Boston.
- Walter J. SAVITCH (1989), A formal model for context-free languages augmented with reduplication, *Computational Linguistics*, 15(4):250–261.
- Paul SCHACHTER and Victoria FROMKIN (1968), A phonology of Akan: Akuapem, Asante, Fante, in *UCLA Working Papers in Phonetics 9*, University of California, Los Angeles, Los Angeles.
- Marcel-Paul SCHÜTZENBERGER (1975), Sur certaines opérations de fermeture dans les langages rationnels, in *Symposia Mathematica*, volume 15, pp. 245–253.
- Hiroyuki SEKI, Takashi MATSUMURA, Mamoru FUJII, and Tadao KASAMI (1991), On multiple context-free grammars, *Theoretical Computer Science*, 88(2):191–229.
- Hiroyuki SEKI, Ryuichi NAKANISHI, Yuichi KAJI, Sachiko ANDO, and Tadao KASAMI (1993), Parallel multiple context-free grammars, finite-state translation systems, and polynomial-time recognizable subclasses of lexical-functional grammars, in *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, pp. 130–139, Association for Computational Linguistics.
- Jeffrey SHALLIT (2008), *A Second Course in Formal Languages and Automata Theory*, Cambridge University Press, New York, NY, USA, 1 edition, ISBN 0521865727, 9780521865722.
- Patricia A. SHAW (2005), Non-adjacency in reduplication, in Hurch (2005), pp. 161–210.
- Daniel SILVERMAN (2002), Dynamic versus static phonotactic conditions in prosodic morphology, *Linguistics*, 40(1):29–60.
- Michael SIPSER (1980), Lower bounds on the size of sweeping automata, *Journal of Computer and System Sciences*, 21(2):195–202.
- Philip SPAELTI (1997), *Dimensions of Variation in Multi-Pattern Reduplication*, Ph.D. thesis, University of California, Santa Cruz.
- Richard William SPROAT (1992), *Morphology and Computation*, MIT press, Cambridge, MA.
- Donca STERIADE (1988), Reduplication and syllable transfer in Sanskrit and elsewhere, *Phonology*, 5(1):73–155.

Thomas STOLZ, Cornelia STROH, and Aina URDZE (2011), *Total Reduplication: The Areal Linguistics of a Potential Universal*, volume 8, Walter de Gruyter, Berlin.

Kristina STROTHER-GARCIA (2018), Imdlawn Tashlhiyt Berber syllabification is quantifier-free, in *Proceedings of the Society for Computation in Linguistics*, volume 1, pp. 145–153, doi:10.7275/R5J67F4D.

Kristina STROTHER-GARCIA (2019), *Using model theory in phonology: a novel characterization of syllable structure and syllabification*, Ph.D. thesis, University of Delaware.

Gregory STUMP (1995), Two types of mismatch between morphology and semantics, in Eric SCHILLER, Elisa STEINBERG, and Barbara NEED, editors, *Autolexical Theory: Ideas and Methods*, number 85 in Trends in Linguistics: Studies and Monographs, pp. 291–318, Mouton De Gruyter, Berlin.

Gregory STUMP (2001), *Inflectional morphology: A theory of paradigm structure*, number 93 in Cambridge Studies in Linguistics, Cambridge University Press, Cambridge.

Terry TAI, Wojciech SKUT, and Richard SPROAT (2011), Thrax: An open source grammar compiler built on OpenFst, in *IEEE Automatic Speech Recognition and Understanding Workshop*, volume 12.

Suzanne URBANCZYK (1999), Double reduplications in parallel, in René KAGER, Harry VAN DER HULST, and Wim ZONNEVELD, editors, *The prosody-morphology interface*, pp. 390–428, Cambridge University Press, Cambridge.

Suzanne URBANCZYK (2001), *Patterns of reduplication in Lushootseed*, Garland, New York.

Suzanne URBANCZYK (2007), Themes in phonology, in Paul DE LACY, editor, *The Cambridge Handbook of Phonology*, pp. 473–493.

Suzanne URBANCZYK (2011), Reduplication, in Mark ARONOFF, editor, *Oxford Bibliography*,
<http://oxfordindex.oup.com/view/10.1093/obo/9780199772810-0036>.

Odile VAYSSE (1986), Addition molle et fonctions p-locales, in *Semigroup Forum*, volume 34, pp. 157–175, Springer.

Rachelle WAKSLER (1999), Cross-linguistic evidence for morphological representation in the mental lexicon, *Brain and Language*, 68(1-2):68–74.

Markus WALTHER (2000), Finite-state reduplication in one-level prosodic morphology, in *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, NAACL 2000, pp. 296–302, Association for Computational Linguistics, Seattle, Washington,
<http://dl.acm.org/citation.cfm?id=974305.974344>.

Ronnie B. WILBUR (1973), *The Phonology of Reduplication*, Ph.D. thesis, University of Indiana, Bloomington, Indiana.

Ronnie B WILBUR (2005), A reanalysis of reduplication in American Sign Language, in Hurch (2005), pp. 595–623.

Colin WILSON (2019), Re (current) reduplication: Interpretable neural network models of morphological copying, *Proceedings of the Society for Computation in Linguistics*, 2(1):379–380.

Moira YIP (1995), Repetition and its avoidance: The case of Javanese, in Keiichiro SUZUKI and Dirk ELZINGA, editors, *Proceedings of the South Western Optimality Theory workshop 1995. Arizona Phonology Conference Volume 5*, pp. 238–262, University of Arizona, Tucson, AZ.

Alan C.L. YU (2007), *A Natural History of Infixation*, number 15 in Oxford Studies in Theoretical Linguistics, Oxford University Press, Oxford.

Kristine YU (2017), Advantages of constituency: Computational perspectives on Samoan word prosody, in *International Conference on Formal Grammar 2017*, p. 105–124, Springer, Berlin.

Sam ZUKOFF (2017), *Indo-European Reduplication: Synchrony, Diachrony, and Theory*, Ph.D. thesis, Massachusetts Institute of Technology.

Kie ZURAW, M. Yu KRISTINE, and Robyn ORFITELLI (2014), The word-level prosody of Samoan, *Phonology*, 31(2):271–327.

Hossep Dolatian

© 0000-0001-5044-8434
hossep.dolatian@stonybrook.edu

Jeffrey Heinz

© 0000-0002-5954-3195
jeffrey.heinz@stonybrook.edu

Department of Linguistics
Institute of Advanced Computational Science
Stony Brook University
Stony Brook, NY, US
<https://you.stonybrook.edu/deovlet/>

Hossep Dolatian and Jeffrey Heinz (2020), *Computing and classifying reduplication with 2-way finite-state transducers*, *Journal of Language Modelling*, 8(1):179–250

doi <https://dx.doi.org/10.15398/jlm.v8i1.-1>

This work is licensed under the *Creative Commons Attribution 4.0 Public License*.

© <http://creativecommons.org/licenses/by/4.0/>