



Journal of Language Modelling

VOLUME 10 ISSUE 1
JUNE 2022



*Institute of Computer Science
Polish Academy of Sciences
Warsaw*

Journal of Language Modelling

VOLUME 10 ISSUE 1
JUNE 2022

Articles

Learning reduplication
with a neural network that lacks explicit variables 1
Brandon Prickett, Aaron Traylor, Joe Pater

Editorials

Introduction to the special section on the interaction
between formal and computational linguistics 39
Timothée Bernard, Grégoire Winterstein

Special section on the interaction between formal and computational linguistics

20 years of the Grammar Matrix: cross-linguistic hypothesis testing
of increasingly complex interactions 49

*Olga Zamaraeva, Chris Curtis, Guy Emerson, Antske Fokkens, Michael
Wayne Goodman, Kristen Howell, T.J. Trimble, Emily M. Bender*

Implementing Natural Language Inference for comparatives 139
Izumi Haruta, Koji Mineshima, Daisuke Bekki



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

GUEST EDITORS OF THE SPECIAL SECTION

Timothée Bernard Université Paris Cité

Grégoire Winterstein Université du Québec à Montréal

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: 50 + 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 Slovak Academy of Sciences, Bratislava;
Constantine the Philosopher University in Nitra, SLOVAKIA

Igor Mel'čuk University of Montreal, CANADA

Glyn Morrill Technical University of Catalonia, Barcelona, SPAIN

Stefan Müller Humboldt-Universität zu 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 Paris-Saclay, 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

Learning reduplication with a neural network that lacks explicit variables

Brandon Prickett¹, Aaron Traylor², and Joe Pater¹

¹ University of Massachusetts Amherst

² Brown University

ABSTRACT

Reduplicative linguistic patterns have been used as evidence for explicit algebraic variables in models of cognition.¹ Here, we show that a variable-free neural network can model these patterns in a way that predicts observed human behavior. Specifically, we successfully simulate the three experiments presented by Marcus *et al.* (1999), as well as Endress *et al.*'s (2007) partial replication of one of those experiments. We then explore the model's ability to generalize reduplicative mappings to different kinds of novel inputs. Using Berent's (2013) *scopes of generalization* as a metric, we claim that the model matches the scope of generalization that has been observed in humans. We argue that these results challenge past claims about the necessity of symbolic variables in models of cognition.

Keywords:
neural networks,
reduplication,
symbolic
computation,
connectionism,
generalization,
phonology

¹The authors would like to thank Max Nelson, Gaja Jarosz, Brendan O'Connor, and the members of the UMass Sound Workshop for helpful discussion and feedback. This research was funded by NSF grant BCS-1650957 to the University of Massachusetts Amherst.

INTRODUCTION

Identity-based patterns in language have been used as evidence for explicit algebraic variables in models of cognition (Marcus 2001; Berent 2013). Marcus *et al.* (1999) demonstrated humans' ability to learn an identity relationship by training infants on reduplicative linguistic patterns of the form ABB and ABA, where A and B were nonce words made up of a single syllable each. Marcus *et al.*'s (1999) participants heard a series of "sentences" made up of such words (e.g. [linana] or [gatiti]) and were then tested on two kinds of novel stimuli: sentences that conformed to the repetition-based pattern in the training phase and sentences that did not. The infants listened longer to novel stimuli that did not conform to the pattern they were trained on than novel stimuli that did. This was taken as evidence that the subjects could successfully generalize the reduplicative pattern.

Marcus *et al.* (1999) demonstrated that a simple recurrent neural network (SRN; Jordan 1986; Elman 1990) could not learn this pattern in a way that led to human-like generalization,² given the data that the infants were exposed to in the experiment. They attributed this failure to a lack of explicit algebraic variables in the model. An example of a variable based analysis of the ABB pattern would be a mapping like $\alpha\beta_1 \rightarrow \beta_2$, where α and β demonstrate syllable identity and the subscripts represent two occurrences of identical syllables. A representation like this would be blind to individual differences within the syllables and would generalize to any kind of novel stimulus. Since the infants in the experiment generalized the pattern to novel items, and the variable free SRN *did not*, Marcus *et al.* (1999) concluded that algebraic variables were necessary to explain their results.

A number of attempts have been made to simulate the results of the experiment without using such variables (see Shultz and Bale 2001; Endress *et al.* 2007, for a summary). The majority of these attempts have been dismissed because they either failed to produce

²While we choose to focus on linguistic generalizations in this paper, a considerable amount of research has also explored non-linguistic generalization (see, e.g. Doumas and Hummel 2010).

a model that discriminated between novel conforming and nonconforming items or because the model used a mechanism that was equivalent to algebraic variables (although see Alhama and Zuidema 2018 for a successful attempt, described more in Section 2). These failures to simulate the results with variable-free models have been taken as further evidence that a symbolic account of cognition is necessary (Marcus 2001).

We reframe the reduplication problem within a modern context with a focus on generalization outside the training data. The remainder of the paper is structured as follows: Section 2 summarizes previous computational work on reduplication generalization, and Section 3 argues that the Sequence-to-Sequence network (Seq2Seq; Sutskever *et al.* 2014) is a straightforward architecture for sequence transduction and is a natural fit for the reduplication problem. Section 4 summarizes a series of simulations that show that a variable-free Seq2Seq network, when trained correctly, can successfully model Marcus *et al.*'s (1999) results. Section 5 then explores the model's ability to generalize to different kinds of novel items, using Berent's (2013) *scopes of generalization* as a metric for the model's success, and argues that its ability to generalize matches that which has been observed in humans. Finally, Section 6 summarizes our findings, discusses why our model was successful, suggests future work, and then concludes the paper.

BACKGROUND

2

The debate between connectionist and symbolic theories of language has often focused on the domain of morphology (for example, see Rumelhart and McClelland 1986; Pinker and Prince 1988). This includes reduplication, where all or part of a word is copied to convey some change in semantic information. Corina (1991) and Gasser (1993) first modeled reduplicative processes with recurrent neural networks. Gasser found an SRN to be insufficient for the task, citing the architecture's need for "a variable of a sort" (1993, p. 6).³

³For discussion on how to integrate variables into connectionist models, see Marcus (2001) and Smolensky and Legendre (2006).

To model the process with a neural network, he instead used a feed-forward model that could discriminate between identical and nonidentical pairs of syllables.

Marcus *et al.* (1999) sought to test how humans learned a reduplicative pattern to see whether variables were necessary to model their behavior (see Rabagliati *et al.* 2019, for evidence of the reliability of these results; for examples of other experimental work on reduplication, see Stemberger and Lewis, 1986 and Waksler 1999). To do this, they trained infants on a pattern that resembled natural language reduplication, in that two out of three syllables in each stimulus were copies of one another. This resulted in two experimental conditions: infants trained on AAB patterns (e.g. with sequences like [lilina]) and those trained on ABB patterns (e.g. with sequences like [linana]). After being trained on one of the two patterns, infants were tested on a variety of items that used novel syllables, as well as novel segments within the syllables. These were either pattern conforming (e.g. [wofefe] for the ABB condition) or pattern nonconforming (e.g. [wowofe] for the ABB condition).

Their results showed that infants looked in the direction of pattern nonconforming items for significantly longer than pattern conforming ones. They took this to mean that the nonconforming items were more surprising for their subjects and that the infants had correctly learned the reduplicative pattern. The final portion of their paper described simulations that they ran with an SRN in an attempt to model the generalization seen in their experiment. While they do not describe these simulations in detail, they do report that the variable free model failed to mimic the infants' behavior and, like Gasser (1993), Marcus *et al.* (1999) concluded that a recurrent neural network would need variables to learn reduplication in a human-like way.

To the best of our knowledge, the cognitive science literature lacks a formal definition for what exactly constitutes a *variable*,⁴ however there is a consensus that SRNs lack any explicit variables (see, e.g., Marcus *et al.* 1999; Seidenberg and Elman 1999). Here, we use the term *explicit* to refer to a representation that has been built into a model's architecture, pretraining, or the input/output features the

⁴ See Clark and Yoshinaka (2014, pp. 13–14) for some discussion of this from a formal language theoretic perspective.

model uses. While it might be the case that SRNs have the ability to capture variable-like representations in their connection weights, unless such weights were set by hand, this would not fall under our definition of an explicit variable.

Marcus *et al.* (1999) also related SRN’s inability to generalize reduplication to another linguistic phenomenon – compositionality, the ability for words to combine to make novel meanings. For example, even if a person had no prior exposure to the sentence, “the bicycling iguana won the game of hop-scotch”, they would be able to compose the meanings of each word to deduce the meaning of the full sentence. Additionally, even if the word “iguana” was substituted with a nonce word like “glork”, humans would still be able to intuit a certain amount of meaning from the sentence. Marcus (1998) demonstrated that SRNs failed to learn human-like compositionality from linguistic data, and more modern neural networks still seem to fail at this task (Lake and Baroni 2017), unless explicit variables are built into their architecture (Korrel *et al.* 2019).

A number of attempts have been made to model the Marcus *et al.* (1999) results without the use of explicit variables. Shultz and Bale (2001) laid out diagnostics for determining whether a simulation properly demonstrates that variables are not necessary for modeling Marcus *et al.*’s (1999) results (see also Marcus 1999). The first diagnostic that they described was that the model cannot be trained on any extra data that was made using an algebraic identity function. Seidenberg and Elman (1999) did not meet this requirement in their simulation of Marcus *et al.*’s (1999) experiment because they exposed their SRN to pretraining that mapped sequences of syllables to an indicator of whether or not each syllable was identical to its predecessor. After the model was familiarized with this identity-based information, it was able to correctly generalize a reduplicative pattern. Since there is no reason to assume the infants in the experiment received such pretraining, this simulation failed to provide evidence for variable free models’ ability to simulate Marcus *et al.*’s (1999) experiment.

Another example of this criterion’s relevance is Alhama and Zuidema’s (2018) *Incremental Novelty Exposure*. This training technique involves presenting data to a model in a way that slowly introduces it to increasing amounts of novelty over time. This forces the neural network to find a more general solution than it might otherwise

be biased toward learning, and was shown to enable a neural network to model the Marcus *et al.* (1999) results. Unfortunately, this use of Incremental Novelty Exposure does not meet Shultz and Bale's (2001) first criterion, since whatever mechanism creates the increasingly novel data would need an explicitly algebraic set of instructions to perform its task.⁵

Shultz and Bale's (2001) next criterion for a variable-free model was that it could not have an architecture that explicitly compares the similarity of separate points in time. Endress *et al.* (2007) point out that even Shultz and Bale's (2001) proposed model does not meet this criterion, since it assumes that there are dedicated, real-valued units representing each timestep in the input. Since these can act like variables over each input feature, and since they are explicitly compared to one another in the model's hidden layer, they are no different from variables in regards to this criterion.

The final requirement that Shultz and Bale (2001) discuss is that to generalize in a human-like way, a model must have more error for pattern non-conforming test items than for the pattern conforming ones. Christiansen and Curtin (1999) failed to meet this criterion, since their model could only differentiate between these two stimulus groups in a way that assigned more error to pattern-conforming items.

Numerous other attempts were made to model Marcus *et al.*'s (1999) results, however Shultz and Bale (2001) and Endress *et al.* (2007) argue that none of them truly meet these three criteria. Endress *et al.* (2007) go on to discuss a successful attempt by Altmann (2002) to model the experimental results without variables, but show that Altmann's (2002) model is unsuccessful given the majority of sampled initial weightings, and that the model makes an incorrect prediction regarding different types of nonconforming test items (i.e. items that followed an AAA pattern, where all three syllables in a sequence are identical). This pathological prediction by Altmann's (2002) learner will be discussed further in Section 4 where we show that our model succeeds on this new type of test item.

⁵Alhama and Zuidema (2018) also test a model without Incremental Novelty Exposure and find similar results to those presented in Section 4. We leave exploring the differences between their model and ours to future work.

OUR MODEL

3

In this section, we present the main differences between our model (a Seq2Seq network with LSTM layers) and the simpler recurrent network used by Marcus *et al.* (1999). For the documentation on the Python packages used to implement the model, see Chollet (2015) and Rahman (2016). The software that we used can be downloaded at <https://github.com/blprickett/Reduplication-Simulations>.

We chose to focus on Seq2Seq models because of their recent success in a number of linguistic tasks (Cotterell *et al.* 2016; Kirov and Cotterell 2018; Prickett 2019; Nelson *et al.* 2020). For example, Kirov and Cotterell (2018) showed that a Seq2Seq network could learn both regular and irregular past tense verbs with almost perfect accuracy. Additionally, when tested on novel verbs, the model’s judgments correlated more with human data gathered by Albright and Hayes (2003) than any previously proposed model (although generalizing to novel verbs in a human-like way was dependent on a particular set of starting weights. See Corkery *et al.* 2019 for more on this).

Crucially for our work, the Seq2Seq network has no algebraic symbols built into its architecture and does not explicitly compare the similarity of any two points in time, meaning that it meets the criteria from Shultz and Bale (2001) discussed in Section 2.⁶ For other recent approaches to computationally modeling reduplication, see Alhama and Zuidema (2018), Wilson (2019), Beguš (2021), Dolatian and Heinz (2020), and Haley and Wilson (2021).

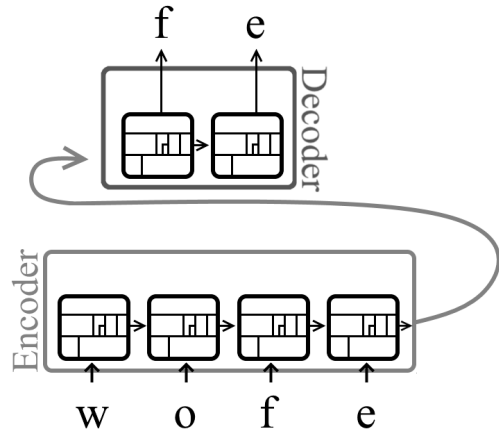
Seq2Seq architecture

3.1

Seq2Seq neural networks were originally designed for machine translation and have the ability to map from one string to another, without requiring a one-to-one mapping between the strings’ elements

⁶While it has become standard in machine translation for Seq2Seq models to use attention (Bahdanau *et al.* 2015), our model does not include this mechanism, since it could be considered to be an implementation of the variables that Marcus (2001) describes. See Nelson *et al.* (2020) for a discussion of how attention can help neural networks learn reduplicative patterns.

Figure 1:
 Illustration of Seq2Seq architecture modeling one of the stimuli (represented as a mapping from the first two syllables to the third syllable) in Marcus *et al.*'s (1999) experiments. Each transcribed sound represents a single timestep



(Sutskever *et al.*, 2014). For example, a sentence like “No, I am your father” could be mapped onto the Spanish sentence “No, soy tu padre”, even though the Spanish sentence has one fewer word. The model performs this mapping by having an encoder and decoder pair built into its architecture. Each member in the pair is its own recurrent network, with the encoder processing the input string one element at a time and the decoder transforming that processed data into an output string that it unpacks through time. Often these elements that make up the input and output sequences are referred to as “timesteps”. In our simulations, each timestep represents a single phonological segment as either a vector of arbitrary features or a vector of phonetically motivated features adapted from the phonological literature.

Figure 1 shows an illustration of the Seq2Seq architecture that resembles the mappings we use in the simulations described in Section 4.1. Here, the encoder passes through the entire input (i.e. the first two syllables) before transferring information (in the form of hidden layer activations) to the decoder. The decoder then unpacks this information, and produces an output string (i.e. the predicted third syllable, [fe]). The Seq2Seq architecture allows these two strings to differ in their length, with the input being four segments long and the output being two.

At each timestep in the input, information is passed forward through the hidden layers of the encoder (represented in the figure by the black boxes within the encoder). Additionally, information is

passed *across* timesteps through the model’s recurrent connections (represented by the black, rightward pointing arrows in the figure). The final recurrent connection in the encoder (represented by the gray arrow) passes this processed information to the decoder, which un-packs it timestep-by-timestep in the output. In all of the simulations discussed in this paper, the encoder is unidirectional, meaning that it passes through the input string once, from left to right.

Long Short-Term Memory (LSTM)

3.2

In all of the simulations presented here, our model uses LSTM hidden layers (Hochreiter *et al.* 2001). These are a kind of recurrent neural network layer which enhances a model’s ability to store information over several timesteps. While this architectural innovation was originally designed to address the problem of vanishing gradients (Bengio *et al.* 1994), it has been demonstrated that LSTM layers can also provide models with added representational power (Levy *et al.* 2018).

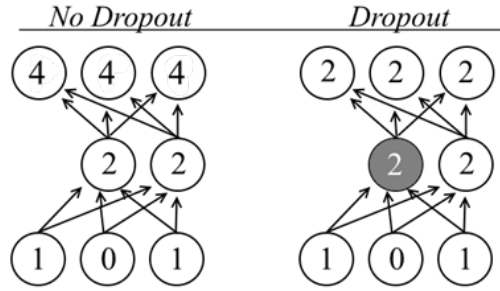
LSTM performs both of these by using cell states: bundles of interacting layers that can learn which information is important for the model to keep track of in the long term, and which information it can forget. This means that during training, the network is not only learning how to predict the output from the input at a given point in time, but also which information at that timestep will help it to predict the output in the future. Crucially, nothing in LSTMs explicitly implements an algebraic variable. While the use of LSTM layers likely has some effect on our model’s predictions, we do not expect it to be the primary factor affecting the network’s generalization and leave the question of how crucial this mechanism is to future work.

Dropout

3.3

Dropout is a regularization method that helps neural networks generalize correctly to items outside of their training data (Srivastava *et al.* 2014). When using dropout, a hyperparameter is chosen between 0 and 1 that represents the probability that any given unit in the network is “dropped out” during training (i.e. all of its incoming/outgoing weights are temporarily set to 0). The set of units that are dropped out

Figure 2:
 A simple feed-forward network, with and without dropout. Each circle is a unit and each arrow is a connection. Dropped out units are in gray. Each unit's output (before dropout) is denoted by the number inside of it. All connections have a weight of 1 and all activation functions are $f(x) = x$



is resampled at each weight update during learning, forcing the model to find a solution that does not depend too heavily on any single unit.

This is illustrated for a simple feed-forward network on the right side of Figure 2. In this illustration, dropout causes the output units to have an activation of 2, instead of 4, because a unit in the middle layer is being dropped out and cannot contribute to the activations in the layer above it. For the simulations presented here, dropout was applied with equal probability to all layers of the network.

4 MODELING MARCUS *ET AL.* (1999)

This section presents simulations of the three experiments described in Marcus *et al.* (1999). In addition to directly simulating these three experiments, Section 4.1 explores the impact of linguistic structure in the model's pretraining, and Section 4.2 simulates a partial replication of the original Marcus *et al.* (1999) experiment performed by Endress *et al.* (2007).

4.1 *Experiments 1 and 2*

In their first two experiments, Marcus *et al.* (1999) trained infants on ABB and ABA patterns (e.g. [wofefe] and [wofewo], respectively) and then measured the infants' listening times to determine whether they generalized the patterns to words containing novel segments. To simulate this, we trained our model to predict the third syllable in each

experimental item, based on the first two.⁷ For all of the simulations presented in this section, the Seq2Seq model was given a four segment input representing the first two syllables, and asked to produce a two segment output representing the third syllable (as illustrated in Figure 1). Segments were represented using vectors made up of 11 feature values, based on standard features used in phonological theory. These features, along with the segments from Marcus *et al.*'s (1999) experiments that they describe, are given in the Supplementary Materials. Both the encoder and decoder each had 4 LSTM layers with 11 units in each layer.

The model was trained using RMSProp (Tieleman and Hinton 2012), a gradual, error-based algorithm, with the default hyperparameter values used in Keras (Chollet 2015). The probability of dropout was .85 (chosen after a small amount of pilot testing before running our final simulations) and the loss that the model was trained to minimize was mean squared error (MSE). MSE was calculated by going through each feature in the model's predicted output, squaring the difference between the predicted value of this feature and the correct value, and averaging across all of these squared differences.

In addition to being trained on the same items as Marcus *et al.*'s (1999) subjects, given in Table 1, the model also went through a pretraining phase meant to familiarize it with the syllables used in the experiment.

Preliminary simulations that were run without this pretraining failed to reproduce the kind of generalization observed in the experiment. The pretraining can be thought of as simulating the experience that the infants would have had with English syllables prior to participating in the experiment (since all of the syllables that were used are attested in English). Unlike the pretraining used by Seidenberg and Elman (1999), there was no identity-based information in this pretraining, meaning that it did not violate the first criterion laid out by Shultz and Bale (2001). Each learning datum in pretraining was a set

⁷Note that this mapping is much simpler than some kinds of reduplication present in natural language (see, e.g., Dolatian and Heinz 2020, for more on this) and should be trivially easy for a neural network to learn. However, since Marcus *et al.* (1999) and the current study are primarily interested in generalization, the formal complexity and learnability of the patterns we look at is irrelevant.

Table 1:
Training data used in our simulations of the first two experiments in Marcus *et al.* (1999). For the phonological features used to represent each sound, see the Supplementary Materials

Experiment	Condition	Stimuli
1	ABA	[gatiga], [ganaga], [gagiga], [galaga], [litili], [ligili], [lilali], [nigini], [ninani], [nilani], [talata], [tatita], [linali], [nitini], [tanata], [tagita]
1	ABB	[tigaga], [nagaga], [gigaga], [lagaga], [tilili], [gilili], [lalili], [ginini], [nanini], [lanini], [latata], [titata], [nalili], [tinini], [natata], [gitata]
2	ABA	[ledile], [lejele], [lelile], [lewele], [widiwi], [wijewi], [wiliwi], [wiwewi], [jidiji], [jijeji], [jiliji], [jiweji], [dedide], [dejede], [delide], [dewede]
2	ABB	[dilele], [jejele], [lilele], [welele], [diwiwi], [jewiwi], [liwiwi], [wewiwi], [dijiji], [jejiji], [lijiji], [wejiji], [didede], [jedede], [lidede], [wedede]

of two randomly sampled syllables that mapped to another randomly chosen syllable.

After being trained on 1000 of these randomly produced data for 1000 epochs (i.e. full passes through the data) with batches of size 50 (i.e. the model made weight updates based on the average error on 50 data points), the model’s decoder weights were set back to their original values (with the encoder weights being preserved) and the experiment simulation began. The model was then trained for 500 epochs (again, with batches of size 50) on a dataset that contained three copies each of the items from Marcus *et al.*’s (1999) training phase. A new random ordering of these data was sampled for each simulation.

At the end of this training, the model was tested on a dataset that contained three copies each of the four test items used by Marcus *et al.* (1999): [wofefe], [dekoko], [wofewo], and [dekede] for Experiment 1 and [bapopo], [kogaga], [bapoba], [kogako] for Experiment 2. Testing involved feeding the model a set of prespecified input values and comparing the model’s resulting output values to the correct outputs (as mentioned above, this comparison is reported using MSE). We used

Table 2: Results from our simulations and the corresponding experiments in Marcus *et al.* (1999)

	<i>MSE</i>				<i>Listening Time</i>			
	Conf.	Nonconf.	<i>t</i> (99)	<i>p</i>	Conf.	Nonconf.	F(14)	<i>p</i>
Exp. 1	.49	.52	-2.8	<.01*	6.3	9.0	25.7	<.01*
Exp. 2	.67	.68	-3.3	<.01*	5.6	7.35	25.6	<.01*

the MSE values obtained from these tests as a dependent variable to compare to the infant listening times reported by Marcus *et al.* (1999). The results for 200 simulations⁸ (50 per condition, per experiment) are given in Table 2, along with the results reported by Marcus *et al.*'s (1999) 32 subjects (8 per condition, per experiment). All MSE values are rounded to the nearest hundredth and averaged across runs.

The results in Table 2 demonstrate that the model, like the infants, differentiates between conforming and nonconforming items in the test data. After running paired t-tests on the MSE values, both Experiment 1 ($t[99] = -2.8, p = .003$) and Experiment 2 ($t[99] = 3.3, p = .0006$) showed significantly less MSE for conforming test stimuli than for nonconforming ones.⁹ This means that the nonconforming stimuli were predicted more poorly by the model, meeting the final diagnostic laid out by Shultz and Bale (2001) for knowing whether a simulation successfully captures the infants' behavior without explicit variables.

One major difference between Marcus *et al.*'s (1999) results and those produced by our model is their respective effect sizes. We do not find this difference troubling, for a number of reasons. First of all, the comparison we make above assumes a linking hypothesis in which each run of the Seq2Seq network is equivalent to a single infant in the experiment. However, it is not obvious that this is the correct

⁸To avoid p-hacking, we ran numerous pilot tests to gauge how many simulations were necessary to gain statistical significance. After the pilots, we reran all 200 simulations and ran all t-tests on these new results.

⁹Following Marcus *et al.* (1999), we combined results from both the ABB and ABA conditions in each experiment, however both groups showed qualitatively similar results, with differences in average MSEs between conforming and nonconforming items of 0.034 and 0.019, respectively.

assumption. For example, one could imagine combining the results from several separate runs to simulate a single human's behavior in the experiment (for example, by averaging their MSE on the test stimuli). This kind of "ensemble" technique, where predictions are combined from multiple models with the same training data, is common in machine learning (Kuncheva 2014) and would reduce the variability in our results (thus increasing the effect size). While it is difficult to determine what linking hypothesis is the most realistic, ensemble learning is an example of how the variable-free model we use here could have an effect size comparable to that of the infants.

Another way that we could reduce the variability and increase the effect size in our results is by reducing the range in which the model's initial weights can vary. Currently, each connection's weight was randomly chosen at the start of each run, which is why each repetition of the simulation got different results, despite getting similar pretraining and identical training data. However, the variability present in these initial weights was due to the default settings in the software we were using, rather than any principled measurement based on actual variability in the brains of newborns. It could be that infants have a relatively low level of variability in their initial state of learning – and if we replicated this in our simulations it could increase our effect size considerably, since we could choose a set of starting weights that led to high levels of generalization and low amounts of variance across runs (for work that pursues this possibility in the context of other phonological patterns, see McCoy *et al.* 2020).

Finally, since the infants would have been exposed to repetition in language prior to their participation in the experiment, their learning could have been aided by this previous linguistic experience. Examples of reduplication are common in both infant-directed speech (Ferguson 1964; Mazuka *et al.* 2008) and adult English (Nevins and Vaux 2003; Ghomeshi *et al.* 2004; Štekauer *et al.* 2012). For example, many of the words directed toward infants (such as "mama" and "choochoo") contain repetition that could be considered an ABB pattern (since two adjacent syllables repeat). Similarly, *Shm* Reduplication (e.g. "pizza-shmizza"; Nevins and Vaux 2003) could be represented as an ABA pattern, with the B representing the [ʃm] sequence and the A's representing the copied material. Mazuka *et al.* (2008) estimate that as much as

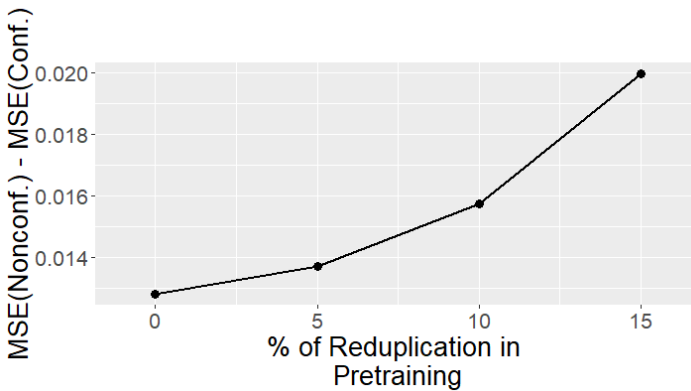


Figure 3:
The effect of reduplication in pretraining on the effect size of the results. Each datapoint represents the average difference between the MSE of conforming and nonconforming items, over 100 repetitions. Half of the simulations were in the ABB experiment condition and half were in the ABA condition

65% of the word types in infant-directed speech could contain some kind of repetition, based on self reporting from Japanese mothers.

We tested the hypothesis that infants might be aided by native language reduplication by running another set of simulations in which we added ABB and ABA conforming words to the model’s pretraining. We varied the percentage of the pretraining that contained these reduplicative words to see if more reduplication in pretraining changed the effect size when simulating the experiments. Additionally, we added a feature to represent the semantic information that would be associated with this repetition. In pretraining, this semantic feature was always -1 when words followed an ABA pattern and 1 whenever words were ABB. When simulating the experiment, this feature was always 0 (to represent the lack of meaning associated with the experimental stimuli). All other hyperparameters were the same as the simulations described above, and the results from them are shown in Figure 3.

Figure 3 demonstrates that the more repeating items that were added into the model’s pretraining, the larger its effect size became when simulating the experiment. While the effect size of the model does not reach the same levels as the infants in Marcus *et al.*’s (1999) study, this demonstrates that adding structure into the model’s pretraining does have the potential to increase effect size. Since the infants in the study were exposed to much more linguistic structure than just reduplication, the benefit they received from their prior experience with English could have had an even larger influence on their ability to generalize in an experimental setting.

Table 3:
Training data used in our simulations of the third experiment in Marcus *et al.* (1999). For the phonological features used to represent each sound, see the Supplementary Materials

Experiment	Condition	Stimuli
3	AAB	[leledi], [leleje], [leleli], [lelewe], [wiwidi], [wiwije], [wiwili], [wiwiwe], [jijidi], [jijije], [jijili], [jijiwe], [dededi], [dedeje], [dedeli], [dedewe]
3	ABB	[dilele], [jelele], [lilele], [welele], [diwiwi], [jewiwi], [liwiwi], [wewiwi], [dijiji], [jejiji], [lijiji], [wejiji], [didede], [jedede], [lidede], [wedede]

4.2

Experiment 3

Marcus *et al.*'s (1999) third experiment required a different set-up than our previous simulations. As shown in Table 3, this experiment replaced the ABA pattern with AAB, exposing infants to either this or ABB words in training, depending on the condition they were assigned.

This was designed to ensure that the infants had not simply learned to expect changes across syllable boundaries in the ABA condition, and a lack of such change in ABB. However, as pointed out by Endress *et al.* (2007), this means that the problem can no longer be modeled as a mapping from the first two syllables to the third, since the model would have no way of predicting the third syllable in AAB sequences.

To overcome this issue, we designed a new kind of simulation in which the model's input included three syllables, but the middle syllable in the input was represented by two empty segments (i.e. segments that had a value of 0 for every feature). The output of the model was a single syllable that was intended to represent the material that the empty syllable was supposed to include (see Devlin *et al.* 2019, for a similar approach in natural language processing). This is illustrated in Figure 4.

Since the second syllable is predictable in both the AAB and ABB conditions, given the other two syllables, this allowed us to test the model on a mapping that was relevant to the design of Experiment 3. While it is unlikely that infants were performing this exact task, framing the problem in this way allows us to work within the constraints of the Seq2Seq network (i.e. that all tasks are a string to string mapping)

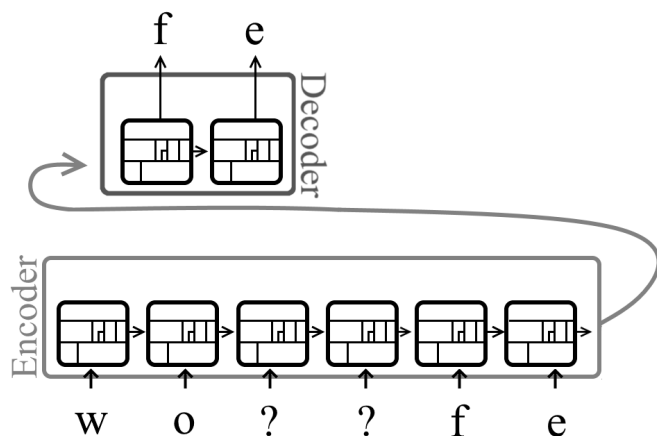


Figure 4:
Illustration of an ABB
mapping in Experiment 3’s
simulations. The
“?” symbols represent
empty segments

and still ensure that, like the infants, the model is not just learning to attend to changes across syllable boundaries.

For pretraining in these simulations, the model was trained to map two randomly chosen syllables with an empty syllable in between them to another randomly chosen syllable. After this pretraining, as in the previous simulations, the decoder’s weights were set back to their initial values. To simulate the experiment’s training phase, the models then trained on a data set similar to those in the previous section. The test phase was also similar to the other experiments, with the model being tested on the words [bapopo], [kogaga], [babapo], and [kokoga]. The results on these test items, averaged over 20 simulations (10 in each condition) are shown in Table 4.

<i>MSE</i>				<i>Listening Time</i>			
Conf.	Nonconf.	<i>t</i> (19)	<i>p</i>	Conf.	Nonconf.	<i>F</i> (14)	<i>p</i>
.56	.57	-2.3	.01635*	6.4	8.5	40.3	<.001*

Table 4:
Results for the
Experiment 3 simulation,
compared to Marcus *et al.*’s
(1999)

The Experiment 3 simulations also included an additional kind of test item. This was designed to simulate the AAA stimuli in Endress *et al.*’s (2007: Appendix A) replication of Marcus *et al.*’s (1999) third experiment. Endress *et al.* (2007) included these stimuli in the test phase to explore a prediction made by Altmann’s (2002) model. That model correctly predicted a preference for conforming stimuli over

Table 5:
Results on Endress *et al.*'s (2007) conforming
and nonconforming stimuli

		MSE	
Conf.	Nonconf. (AAA)	$t(19)$	p
.56	.57	-2.22	.01933*

Marcus *et al.*'s (1999) nonconforming ones, however it predicted an even stronger preference for stimuli that followed an AAA style pattern. That is, stimuli such as [bababa], where all three syllables are the same.

Endress *et al.* (2007) showed that when a replication of Marcus *et al.*'s (1999) third experiment was run that also tested participants' preferences for this kind of stimulus, humans still preferred items that conformed to the reduplicative pattern they were trained on. To ensure that the interpretation of our model's results does not fall into the same trap as Altmann's (2002), we also tested it on the Endress *et al.* (2007: Appendix A) test items: [bababa] and [kokoko]. The results, averaged over 20 simulations (10 in each condition), are given in Table 5.

These simulations show that our model can predict the results of Marcus *et al.*'s (1999) third experiment, as well as Endress *et al.*'s (2007) partial replication of that experiment. The model's MSE was significantly higher for both the standard nonconforming items ($t[19] = -2.30, p = .01635$), as well as the AAA nonconforming ones ($t[19] = -2.22, p = .01933$).

5 EXPLORING THE MODEL'S SCOPE OF GENERALIZATION

In Section 4, we demonstrated our model's ability to simulate Marcus *et al.*'s (1999) experiment results, despite its lack of variables. However, these results only paint a partial picture of how well the model is able to generalize reduplication. Marcus *et al.* (1999) tested infants on words that used segments that were completely novel in the context of the experiment (i.e. they were not present in the words that infants were trained on), however, all of the segments in the experiment

	i	e	o	a
p	pi	pe	po	pa
b	bi	be	bo	ba
t	ti	te	to	ta
d	di	de	do	da

Table 6:
Example of generalization to a novel syllable.
Gray cells represent training data,
bolded item indicates the crucial testing item

were present in English, which means that the infants would have had a considerable amount of experience with them. We simulated this experience in our models using randomly produced pretraining, which entails that the model never needed to generalize reduplication to completely novel phonemes. This also means that it is impossible to know, based on those results, whether the model learned an algebraic function like $\alpha\beta_1 \rightarrow \beta_2$, or whether it learned a less general pattern like “if feature F is 1 in the third sound in the input, feature F' should be 1 in the first sound of the output”.

To better understand the mappings being learned by the Seq2Seq network, we structured the simulations in this section to map a single syllable (e.g. [ba]) to two copies of itself (e.g. [baba]).¹⁰ We then tested how well the model generalized this mapping when given withheld data at various levels of novelty. To do this, we followed Berent’s (2013) proposal regarding the *scopes of generalization* that are possible for such identity-based patterns. We summarize the three scopes here, and then in Section 5.1–5.3, we explain the series of simulations we ran to determine which scope best describes our model’s performance.

The simplest form of generalization that Berent (2013) discussed is to novel words (which in this context is equivalent to generalization to novel syllables, since the network is blind to the difference between these levels of representation). This is illustrated for a reduplicative pattern in Table 6, with the gray cells representing the input syllables seen in the training data and the bolded syllable being the input for a test item withheld from training.

¹⁰This also resembles natural language reduplication more closely than the Marcus *et al.* (1999) pattern does. For an example, see reduplication in the language Karao, which doubles the stem of a word to change the number of some verbs: [manbakal] “fight each other, 2 people” \rightarrow [manbabakal] “fight each other, > 2 people” (Štekauer *et al.* 2012).

Table 7:
 Example of generalization to a novel segment.
 Gray cells represent training data,
 bolded items indicate crucial testing item

	i	e	o	a
p	pi	pe	po	pa
b	bi	be	bo	ba
t	ti	te	to	ta
d	di	de	do	da

If a model correctly predicts the mapping [da]→[dada] after being trained on data that does not include the input [da] (but that does include other syllables containing both [d] and [a]), it would successfully be performing this scope of generalization. This would demonstrate that the model did not simply memorize individual input + output pairs, but doesn't show that the model has learned anything more sophisticated than how to copy individual segments. For example, it could have learned patterns like “if [d] occurs as the first segment in the input, make [d] the first and third segments in the output.”

The next scope is generalizing to novel segments. As mentioned in Section 4, we represent segments as vectors of phonological features. When testing this scope, we trained the model on every relevant value for each feature, but not on all of the possible feature value combinations. This is demonstrated in Table 7, using the same shading scheme that was described above.

In the example in Table 7, the model is trained on syllables containing [p], [b], and [t], with [d] remaining outside of its training data. This would give it experience in training with all of the feature values that make up [d] (since it shares every value but [voice] with [t] and it *does* share its value for [voice] with [b]), without ever seeing them together in the same vector. This scope of generalization demonstrates that a learner is doing more than just memorizing a mapping for each segment. Instead, if a model generalizes at this level, it has acquired a broader generalization that might reference specific feature values. For example, it may have learned the generalization “if the first segment in the input is −1 for [voice], make the first and third segments in the output have a value of −1 for [voice].”

Berent (2013) points out that generalization to novel segments would still not demonstrate that a model has learned a full identity-based function. To show this, a model would need to demonstrate

	i	e	o	a
p	pi	pe	po	pa
b	bi	be	bo	ba
t	ti	te	to	ta
d	di	de	do	da
n	ni	ne	no	na

Table 8:

Example of generalization to a novel feature value.

Gray cells represent training data, bolded item indicates the crucial testing item

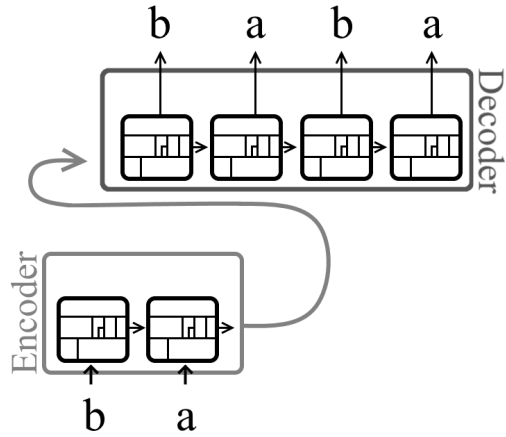
its ability to generalize to novel feature values, which Berent (2013) calls “across the board” generalization and Marcus (1998) describes as “outside of the training space”.¹¹ This is demonstrated in Table 8, where the learner is only trained on oral consonants (i.e. sounds made without nasal resonance) and then tested on the nasal consonant [n].

In the example from Table 8, the model has only been exposed to the feature value [nasal] = -1 in its input, so if it generalizes to [na], there is no way it could have learned a pattern that depends on feature value based mappings. Generalization to novel feature values means that a model has learned that the pattern is independent of any particular feature. For example, the model could have learned the function $\alpha \rightarrow \alpha\alpha$, where α can be any arbitrary syllable.

To test which scope of generalization our model could achieve, we ran three kinds of simulations that were more carefully aimed at this question than the Marcus *et al.* (1999) experiment: one in which the model was tested on a novel syllable made up of segments it had seen reduplicating in its training data (Section 5.1), one in which the model was tested on a syllable made with a segment that it had not received in training (Section 5.2), and one in which the model was tested on a syllable with a novel segment containing a feature value that had not been presented in the training data (Section 5.3). None

¹¹ Note that all scopes of generalization talked about so far can be thought of as being “outside the training space”, but Marcus and colleagues often use this term to specifically refer to generalization to novel feature values. By “feature” here we mean the most atomic level of a model’s representation. For our model, this is the level of phonological features, following standard linguistic theory (see, e.g., Chomsky and Halle 1968).

Figure 5:
Illustration of mappings
in this section’s simulations



of the simulations described here used a pretraining phase like those in Section 4.

In the results presented in this section, the set of possible segments and the feature values representing those segments were randomly produced in each simulation, unless otherwise noted. Input features for these simulations were binary (either -1 or 1), to avoid ambiguity in interpreting the model’s success. To ensure that each language had consonants and vowels present in its segment inventory, segments were divided into these two categories by treating the first feature as [syllabic], i.e. any of the randomly produced feature vectors that began with -1 were considered a consonant and any that began with 1 were considered a vowel. No randomly produced language inventories were used that consisted of only consonants or only vowels.

The toy language for any given simulation consisted of all the possible consonant+vowel syllables that could be made with that simulation’s randomly created segment inventory (all inventories contained forty segments total, unless otherwise noted). Crucially, before the data was given to the model, some portion of it was withheld for testing (see the subsections below for more information on what was withheld in each testing condition). The mappings that the model was trained on took a single syllable (e.g. [ba]) as input and produce two syllables (e.g. [baba]) as output, as shown in Figure 5.

The models were trained for 1000 epochs, with batches that included all of the training data. There were 18 units in the model’s

hidden layer, the probability of dropout was either 0 or .75, and all other hyperparameters were the same as in Section 4 (as in the previous section, hyperparameters were chosen after a small amount of piloting was performed). To test whether the model generalized to withheld data at the end of training, a much stricter definition of success was used than in the Marcus *et al.* (1999) experiments. The model was given the relevant withheld item as input, and the output it predicted was computed using Keras’s “predict()” function (Chollet, 2015), which performs a single forward pass through the network. Since the model is not probabilistic, these predictions do not vary given the same input and set of connection weights. These predictions were compared to the corresponding correct outputs (i.e. the reduplicated form of the stem it was given). If every feature value in the predicted output had the same sign (positive/negative) as its counterpart in the correct output, the model was considered to be successfully generalizing the reduplication pattern. However, if any of the feature values did not have the same sign, that model was considered to have failed at the generalization task.

Generalization to novel syllables

5.1

Our first set of simulations tested whether the model could generalize to novel syllables. If the model failed at this task, then it would mean that it was memorizing whole syllables in the training data, rather than extracting any actual pattern from the mappings that it was trained on. The model successfully reduplicated all of the syllables it had been trained on in all runs for this condition. Additionally, when no dropout was used, it successfully generalized to novel syllables in 22 of the 25 simulations (88%). This shows that a standard Seq2Seq model, with LSTM but no dropout, can perform generalization to novel syllables, and does so a majority of the time. Dropout did not have a noticeable effect on the model’s ability to generalize. When the probability of units dropping out was .75, it again generalized to novel syllables in 22 of the 25 simulations (88%).

Generalization to novel segments

5.2

Our next set of simulations tested the model’s ability to generalize to novel segments. If the model failed at this task, it would mean that it

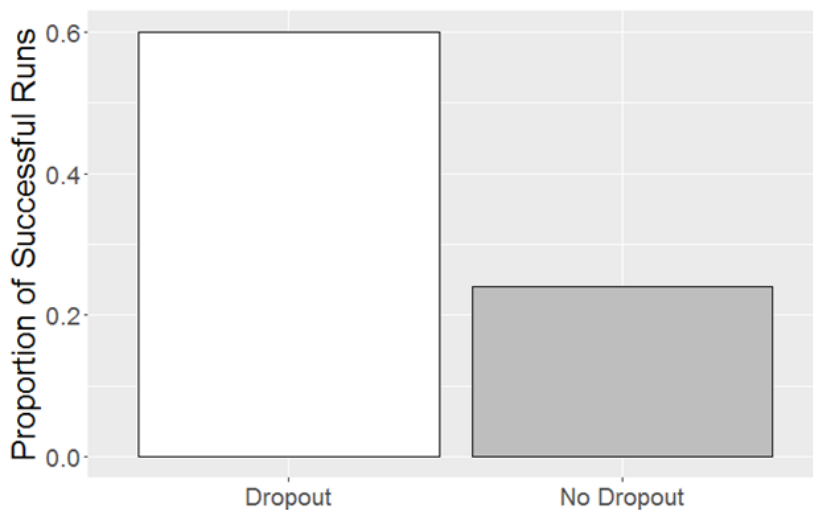


Figure 6: Difference between dropout with probabilities of .75 and 0 in generalization to novel segments

was only learning generalizations that referred to individual sounds, such as “if [d] is the first segment in the input, make [d] the first and third segments in the output.” The model successfully reduplicated syllables from training in 24 of the 25 runs for this condition when no dropout was applied. However, it failed to generalize to novel segments in the majority of runs, with only 6 out of 25 simulations being successful (24%). This shows that a standard Seq2Seq model, with LSTM but no dropout, does not reliably generalize to unseen segments.

However, when the probability of a unit dropping out was increased to .75, the model successfully reduplicated syllables from training in all runs and generalized to novel segments in 15 out of 25 runs (60%). This means that as long as dropout is used in training, the model will reliably achieve this scope of generalization. This difference between the two dropout conditions is illustrated in Figure 6.

5.3

Generalization to novel feature values

Our next set of simulations tested the model’s ability to generalize to novel feature values. Failing at this means that the model learned generalizations that depend on individual features, rather than completely abstract algebraic functions like $\alpha \rightarrow \alpha\alpha$. In this condition, the

inventory was designed by hand and always contained 43 segments, in order to more easily withhold a single feature value. The feature vectors that represented these segments are given in the Supplementary Materials. The withheld segment was always [n], with the withheld feature value being [nasal] = 1. A variety of other segment inventories were tested, with no changes in the model's performance.

Despite the fact that the model achieved perfect performance on trained syllables, it was never able to generalize to novel feature values, regardless of whether dropout probability was 0 or .75. A number of other dropout settings were attempted with no success at increasing the scope of generalization to this level. This suggests that Seq2Seq models, regardless of whether they are regularized with dropout, cannot generalize to novel feature values.¹²

*Which scope of generalization is observed
in human language learning?*

5.4

In this section, we argue that the generalization observed in our Seq2Seq simulations matches the generalization demonstrated in past experiments involving humans. As we'll discuss, the ability of humans to generalize identity-based patterns to novel words and segments is well documented and uncontroversial, but we find that the evidence for humans generalizing to novel feature values is weak.

When discussing generalization of reduplicative patterns, Berent (2013) used Hebrew speakers' judgments regarding an AAB pattern present in their language's phonotactics. In Hebrew, the first two consonants in a word's stem cannot be identical (i.e. the first three consonants are not allowed to match the pattern AAB, where the A's represent a repetition of the same consonant). For example, the word [simem] 'he intoxicated' is acceptable, while the nonce word *[sisem] is not. Berent (2013) reviewed a number of past experiments that showed speakers generalizing this pattern by having them rate the acceptability of various kinds of novel words.

¹²One reason why you might expect this behavior is that novel feature values represent a particularly strong violation of the "independent and identically distributed" assumption (see Le Boudec 2011, for an introduction) often made in statistical learning.

Generalization to novel words/syllables was demonstrated by Berent and Shimron (1997) in an experiment that asked Hebrew-speaking participants to rate nonce words. These words were made up of segments that were attested in Hebrew, such as [s] and [m], making them equivalent to the novel syllables that we tested our model on in Section 5.1. Speakers in this experiment rated words with s-s-m stems (like *[sisem]) as significantly less acceptable than words with s-m-m and p-s-m stems. This demonstrated that Hebrew speakers were doing more than just memorizing the lexicon of their language (i.e. that they could extract phonotactic patterns).

Generalization to novel segments by Hebrew speakers was shown in Berent *et al.* (2002), corresponding to the scope of generalization that the network with dropout achieved in Section 5.2. The segments of interest were /tʃ/, /dʒ/ and /w/, all of which are not present in native Hebrew words. Even when these non-native phonemes were used, Hebrew speakers rated words whose first two consonants were identical (e.g. dʒ-dʒ-r) as worse than those that did not violate the phonotactic restriction (e.g. r-dʒ-dʒ). This demonstrated that speakers had not just memorized a list of consonants that cannot cooccur (e.g. *pp, *ss, *mm, etc.) while acquiring their phonological system, since this list would not have included sounds like [w].

Finally, Berent *et al.* (2002) showed that speakers can generalize the *AAB pattern to the segment [θ], which they claimed represented generalization to the novel feature value [wide]. However, [wide] is not used in any standard phonological feature theory (e.g. Chomsky and Halle 1968; Hayes 2011). Using a standard featural representation for [θ], such as [+anterior, +continuant, –strident], would mean that [θ] does not represent a novel feature value for Hebrew, since the language contains other, native, [+anterior], [+continuant], and [–strident] sounds (e.g. [t], [ʃ], and [f], respectively). This is illustrated in Table 9.

Berent *et al.* (2002) present a number of arguments in favor of using the feature [wide], rather than a more standard phonological representation. First, they argue that since [wide] is a more phonetically invariant feature value than representations like [+anterior] and [–strident], that it is more likely to be psychologically real (see also Gafos 1999). However, it is unclear whether phonological features *should* have invariant phonetic correlates (see Hamann 2010,

	[anterior]	[continuant]	[strident]
t	+	-	
ʃ	-	+	+
f		+	-
θ	+	+	-

Table 9:
 Demonstration that [θ] does not represent any novel feature values for Hebrew speakers when a standard set of features is used. Gray cells represent the crucial feature values needed to describe [θ] that are present in native Hebrew sounds

sec. 2.1 for some discussion of this), since the process that the mind uses to map phonetic information to phonological features is an open question.

Their second piece of evidence was that Hebrew speakers map [θ] to [t] when borrowing non-Hebrew words, and that this must be the result of a representational difference between it and other novel sounds that are borrowed faithfully into the language. However, it has been widely observed that interdental sounds like [θ] are more likely to be mapped incorrectly than other phonemes when words containing them are borrowed into a language (see, e.g., Rau *et al.* 2009; Hanulikova and Weber 2010). This is likely due to phonetic difficulty, since children acquiring English as their first language are more likely to make production (Moskowitz 1975) and perception (Skeel 1969) errors when dealing with interdental sounds than other kinds of phonemes.

Another experiment claiming to demonstrate generalization to novel feature values is Berent *et al.* (2014). In this paper, the authors claim to observe generalization of a reduplicative pattern in American Sign Language to novel signs made up of novel feature values. However, they are using the word “feature” differently than we do here. While they define features as the description of an entire hand shape, our definition is closer to the sign language features proposed by Brentari (1998), where feature values are the most atomic part of a sign’s representation (for example, the position of individual fingers). Since their participants would have had prior linguistic and non-linguistic experience with visual stimuli that involved hands in a variety of positions (analogous to the pretraining we used in Section 4), these would not be truly novel feature values for them. Berent *et al.* (2016) also used signed language to test whether humans could generalize to novel feature values. Specifically, they showed that na-

tive speakers of auditory languages seemed to generalize reduplicative patterns from their L1 to signed nonce words. However, this study also used visual stimuli that could easily be represented using features that participants were already exposed to in non-linguistic contexts (i.e. the different positions of the parts of a hand). Furthermore, since the participants in the experiment were not experienced speakers of a signed language, they could have been mapping the signs to auditory representations in their mind, which would mean that they were not generalizing the reduplicative patterns to novel features at all.

To our knowledge, no experiment has conclusively tested humans' ability to generalize to novel feature values. Such an experiment would be difficult, since children stop reliably perceiving most novel feature contrasts at a relatively young age (see, e.g., Werker and Tees 1983). Because of this, we conclude that our model generalizes in a way that captures the scopes observed thus far in human behavior: generalization to novel syllables and generalization to novel segments.

6 DISCUSSION

6.1 *Summary of results*

In Section 4, we showed that a Seq2Seq model without any explicit variables can capture the results from all three of Marcus *et al.*'s (1999) experiments. Results from these simulations are summarized in Figure 7.

We also demonstrated that unlike Altmann's (2002) model, ours does not predict a preference toward AAA items when trained on AAB and ABB sequences. This means our model can also predict the results reported by Endress *et al.* (2007: Appendix A).

Next, we probed our model further in Section 5, more carefully testing which scope of generalization it could capture when trained on a reduplicative pattern. A summary of these results can be viewed in Figure 8.

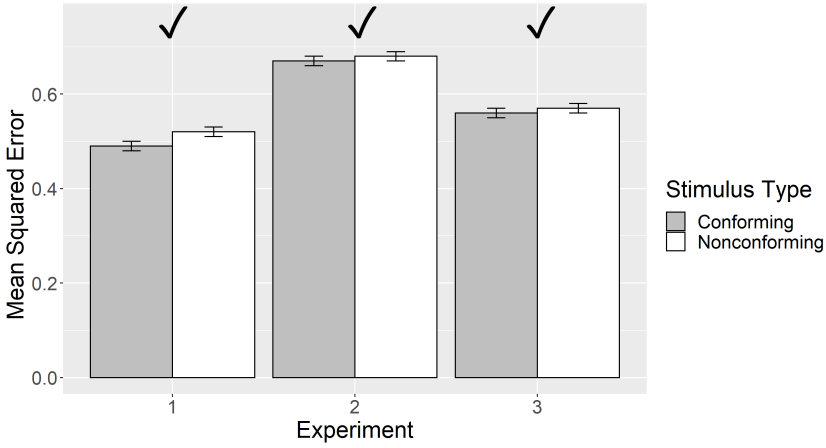


Figure 7: Results from our simulations of the three experiments described in Marcus *et al.* (1999). Error bars show standard error of the mean, check symbols indicate successful simulations of the behavior observed in each experiment

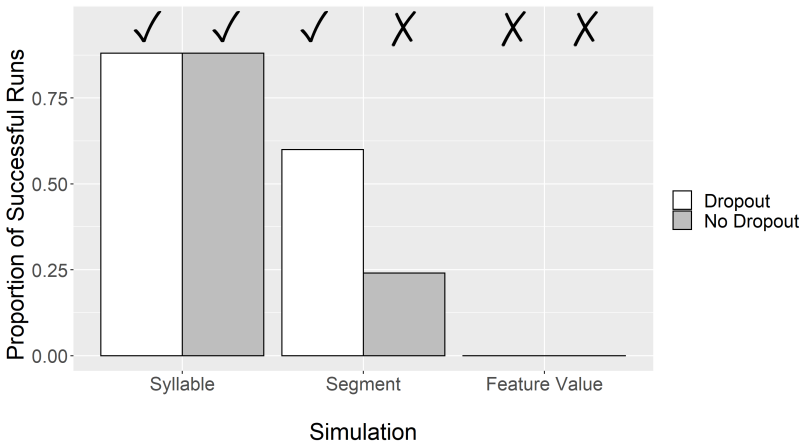


Figure 8: Summary of results for each dropout condition and scope of generalization. Checks and 'X' symbols indicate which conditions the model reliably succeeded and failed in, respectively

The findings from this series of simulations showed that even without dropout, a Seq2Seq model is not simply memorizing mappings for each individual datum, since it was able to generalize reduplication to novel syllables. We also showed that the model, when using dropout in training, can reliably generalize reduplication to novel segments. However, generalization to novel feature values was never achieved, regardless of whether or not dropout was used.

6.2 *Why can the Seq2Seq model learn generalizable reduplication?*

Why neural networks generalize in the way that they do is still an open question (see, e.g., Valle-Perez *et al.* 2018). However, the results presented in Section 5 shed some light on why our model succeeded in capturing the infant behavior reported by Marcus *et al.* (1999), while past neural networks failed (for similar work on probing neural networks using generalization tasks, see, e.g., Linzen *et al.* 2016; McCoy *et al.* 2018). First of all, we found that the network could never generalize to novel feature values. This explains why past models that were given no pretraining could not capture the infant generalization – since the pretraining exposed our model to all of the feature values present in both the training and testing phase of the experiment.

Additionally, we found that generalization to novel segments only occurred reliably for our model when it used dropout (Srivastava *et al.* 2014), a standard regularization technique in machine learning. This also explains the failure of past models, since (to our knowledge) dropout has not been used in past attempts to simulate the experiment (although, see Alhama and Zuidema 2018, for the successful application of a related mechanism).

It remains an open question whether other forms of regularization (such as an L2 prior) would be as successful at this task as dropout was. One hypothesis for why dropout worked is that it caused certain training data to be indistinguishable from crucial testing data. For example, if the training set included the inputs [pa] and [da], but [ta] was withheld, a model without dropout would not generalize to the novel item because it was never trained on reduplicating [t]. However, if dropout is applied, then in a subset of epochs, the unit activations distinguishing [t] from [d] would no longer be available to the

model. This would allow it to learn how to reduplicate a syllable that is ambiguous between [ta] and [da]. While this would not allow the model to generalize to novel feature values that were never activated in training, it could provide enough information for generalization to withheld segments. If this hypothesis is correct, then other forms of regularization may not be as successful at increasing the model's scope of generalization. Testing these other methods is an important avenue that future research should explore.

Future work

6.3

There are a number of other opportunities that present themselves for future work. For example, running experiments on humans that test for generalization of reduplicative patterns to truly novel feature values (if such a test is possible) would be beneficial, since it would help shed more light on what scope of generalization computational models need to achieve.

Probing the Seq2Seq model further to better understand the representations it learns when acquiring reduplication is another important direction for future research to investigate. Our results suggest that when dropout is used, the model is likely learning a feature-based representation, but understanding which parts of the model's architecture are responsible for this is still an open question. Methods exist for probing networks in this way (see, e.g., Beguš 2021; Dankers *et al.* 2021) and could help shed light on what exactly is necessary for a model to capture the results from Marcus *et al.* (1999).

Another area future research should pursue is the relationship between formal descriptions of reduplication (e.g. Clark and Yoshinaka 2014; Dolatian and Heinz 2020; Wang 2021) and the results discussed here. While both experimental (Moreton *et al.* 2021) and computational (Nelson *et al.* 2020) work has touched on the formal complexity of reduplication, there is still much work to be done to bridge the kind of modeling done here with models like finite state automata that are often used to more precisely describe the learnability of patterns.

The learning biases inherent to the Seq2Seq model should also be explored. For example, Endress *et al.* (2007) and Gallagher (2013) both found that identity-based patterns were easier for humans to

learn than more arbitrary ones, and concluded that explicit variables were necessary to model this behavior. Testing to see whether Seq2Seq networks with dropout show a similar bias for identity-based patterns could be another way of testing whether variables are needed in models of cognition.

Additionally, the question of compositionality should be revisited, given our findings on reduplication. If neural networks' ability to model these two phenomena is related, as Marcus *et al.* (1999) suggested, then given the right pretraining, a Seq2Seq network with dropout should be able to learn compositional linguistic patterns. Capturing compositionality may require testing novel kinds of featural representations, since our results suggest that novel feature values in the input or output will always be impossible for the model to generalize to (see Lake and Baroni 2017, sec. 5, for a similar suggestion).

6.4

Conclusions

In the past, it has been claimed that it is impossible for variable-free neural networks to generalize reduplicative patterns in a human-like way (Marcus *et al.* 1999; Marcus 2001; Berent 2013). Here, we presented results showing that a network with no variables, that has been pretrained on randomized data, can capture Marcus *et al.*'s (1999) experimental results. Since our simulations met all three of the criteria laid out by Shultz and Bale (2001) for a successful variable-free simulation of the experiment, our results challenge the claim that simulating these results is only possible with a symbolic model of cognition.

We also probed our model's abilities to determine more precisely what scope of generalization it was using. We found that it could generalize to novel syllables and novel segments, but not to novel feature values. This matches the scope of generalization observed thus far in humans, and also explains why pretraining was necessary for our model to simulate Marcus *et al.*'s (1999) results.

More broadly, this paper challenges the idea that variable-free neural networks are insufficient for modeling human behavior and provides another example of the Seq2Seq architecture successfully mirroring the linguistic capabilities of humans.

REFERENCES

- Adam ALBRIGHT and Bruce HAYES (2003), Rules vs. analogy in English past tenses: A computational/experimental study, *Cognition*, 90(2):119–161.
- Raquel G. ALHAMA and Willem ZUIDEMA (2018), Pre-Wiring and pre-training: What does a neural network need to learn truly general identity rules?, *Journal of Artificial Intelligence Research*, 61:927–946.
- Gerry T.M. ALTMANN (2002), Learning and development in neural networks – the importance of prior experience, *Cognition*, 85(2):B43–B50.
- Dzmitry BAHDANAU, Kyunghyun CHO, and Yoshua BENGIO (2015), Neural machine translation by jointly learning to align and translate, in Yoshua BENGIO and Yann LECUN, editors, *3rd International Conference on Learning Representations, Conference Track Proceedings*.
- Gašper BEGUŠ (2021), Identity-based patterns in deep Convolutional Networks: Generative Adversarial Phonology and reduplication, *Transactions of the Association for Computational Linguistics*, 9:1180–1196.
- Yoshua BENGIO, Patrice SIMARD, and Paolo FRASCONI (1994), Learning long-term dependencies with gradient descent is difficult, *IEEE Transactions on Neural Networks*, 5(2):157–166.
- Iris BERENT (2013), The phonological mind, *Trends in Cognitive Sciences*, 17(7):319–327.
- 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, Amanda DUPUIS, and Diane BRENTARI (2014), Phonological reduplication in sign language: Rules rule, *Frontiers in Psychology*, 5(560):1–15.
- Iris BERENT, Gary MARCUS, Joseph SHIMRON, and Adamantios I. GAFOS (2002), The scope of linguistic generalizations: Evidence from Hebrew word formation, *Cognition*, 83(2):113–139.
- Iris BERENT and Joseph SHIMRON (1997), The representation of Hebrew words: Evidence from the obligatory contour principle, *Cognition*, 64(1):39–72.
- François CHOLLET (2015), Keras, <https://github.com/keras-team/keras>.
- Noam CHOMSKY and Morris HALLE (1968), *The sound pattern of English*, Harper & Row.
- Morten H. CHRISTIANSEN and Suzanne L. CURTIN (1999), The power of statistical learning: No need for algebraic rules, in Martin HAHN and Scott C. STONESS, editors, *Proceedings of the 21st Annual Conference of the Cognitive Science Society*, pp. 114–119, Routledge.

Alexander CLARK and Ryo YOSHINAKA (2014), Distributional learning of parallel multiple context-free grammars, *Machine Learning*, 96(1–2):5–31.

David Paul CORINA (1991), *Towards an understanding of the syllable: evidence from linguistic, psychological, and connectionist*, PhD Thesis, University of California, San Diego.

Maria CORKERY, Yevgen MATUSEVYCH, and Sharon GOLDWATER (2019), Are we there yet? Encoder-decoder neural networks as cognitive models of English past tense inflection, in Anna KORHONEN, David TRAUM, and Lluís MÀRQUEZ, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3868–3877.

Ryan COTTERELL, Christo KIROV, John SYLAK-GLASSMAN, David YAROWSKY, Jason EISNER, and Mans HULDEN (2016), The SIGMORPHON 2016 shared task—morphological reinflection, in Micha ELSNER and Sandra KUEBLER, editors, *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pp. 10–22.

Verna DANKERS, Anna LANGEDIJK, Kate MCCURDY, Adina WILLIAMS, and Dieuwke HUPKES (2021), Generalising to German plural noun classes, from the perspective of a Recurrent Neural Network, *Conference on Computational Natural Language Learning*, <https://aclanthology.org/2021.conll-1.8>.

Jacob DEVLIN, Ming-Wei CHANG, Kenton LEE, and Kristina TOUTANOVA (2019), BERT: Pre-training of deep bidirectional transformers for language understanding, in Jill BURSTEIN, Christy DORAN, and Tamar SOLORIO, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, <https://aclanthology.org/N19-1423>.

Hossep DOLATIAN and Jeffrey HEINZ (2020), Computing and classifying reduplication with 2-way finite-state transducers, *Journal of Language Modelling*, 8(1):179–250.

Leonidas DOUMAS and John E. HUMMEL (2010), A computational account of the development of the generalization of shape information, *Cognitive Science*, 34(4):698–712.

Jeffrey L. ELMAN (1990), Finding structure in time, *Cognitive Science*, 14(2):179–211.

Ansgar D. ENDRESS, Ghislaine DEHAENE-LAMBERTZ, and Jacques MEHLER (2007), Perceptual constraints and the learnability of simple grammars, *Cognition*, 105(3):577–614.

Charles A. FERGUSON (1964), Baby talk in six languages, *American Anthropologist*, 66(6_PART2):103–114.

Adamantios I. GAFOS (1999), *The articulatory basis of locality in phonology*, Taylor & Francis.

Michael GASSER (1993), *Learning words in time: Towards a modular connectionist account of the acquisition of receptive morphology*, Indiana University, Department of Computer Science.

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, ISSN 0167-806X, <https://www.jstor.org/stable/4048061>.

Coleman HALEY and Colin WILSON (2021), Deep neural networks easily learn unnatural infixation and reduplication patterns, *Proceedings of the Society for Computation in Linguistics (SCiL)*, pp. 427–433.

Silke HAMANN (2010), Phonetics-phonology interface, in Nancy C. KULA, Bert BOTMA, and Kuniya NASUKAWA, editors, *The Bloomsbury Companion to Phonology*, Bloomsbury Companions, Bloomsbury.

Adriana HANULIKOVA and Andrea WEBER (2010), Production of English interdental fricatives by Dutch, German, and English speakers, in Magdalena WREMBEL, Malgorzata KUL, and Katarzyna DZIUBALSKA-KOLACZYK, editors, *New Sounds 2010: Sixth International Symposium on the Acquisition of Second Language Speech*, pp. 173–178, Peter Lang Verlag.

Bruce HAYES (2011), *Introductory phonology*, John Wiley & Sons.

Sepp HOCHREITER, Yoshua BENGIO, Paolo FRASCONI, and Jürgen SCHMIDHUBER (2001), *Gradient flow in recurrent nets: the difficulty of learning long-term dependencies*, A field guide to dynamical recurrent neural networks. IEEE Press.

Michael I. JORDAN (1986), Serial order: A parallel distributed processing approach, Technical report, University of California, San Diego.

Christo KIROV and Ryan COTTERELL (2018), Recurrent Neural Networks in linguistic theory: Revisiting Pinker & Prince (1988) and the past tense debate, *Transactions of the Association for Computational Linguistics*, 6:651–665.

Kris KORREL, Dieuwke HUPKES, Verna DANKERS, and Elia BRUNI (2019), Transcoding compositionally: Using attention to find more generalizable solutions, in *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 1–11, Association for Computational Linguistics, doi:10.18653/v1/W19-4801.

Ludmila I. KUNCHEVA (2014), *Combining pattern classifiers: methods and algorithms*, John Wiley & Sons.

Brenden M. LAKE and Marco BARONI (2017), Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks, in Jennifer DY and Andreas KRAUSE, editors, *Proceedings of the 35th International Conference on Machine Learning*.

- Jean-Yves LE BOUDEC (2011), *Performance evaluation of computer and communication systems*, Epfl Press.
- Omer LEVY, Kenton LEE, Nicholas FITZGERALD, and Luke ZETTEMAYER (2018), Long Short-Term Memory as a dynamically computed element-wise weighted sum, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 732–739.
- Tal LINZEN, Emmanuel DUPOUX, and Yoav GOLDBERG (2016), Assessing the ability of LSTMs to learn syntax-sensitive dependencies, *Transactions of the Association for Computational Linguistics*, 4:521–535.
- Gary MARCUS (1998), Rethinking eliminative connectionism, *Cognitive Psychology*, 37(3):243–282.
- Gary MARCUS (1999), Do infants learn grammar with algebra or statistics? Response, *Science*, 284(5413):436–437.
- Gary MARCUS (2001), *The algebraic mind*, Cambridge, MA: MIT Press.
- Gary MARCUS, Sugumaran VIJAYAN, S. Bandi RAO, and Peter M. VISHTON (1999), Rule learning by seven-month-old infants, *Science*, 283(5398):77–80.
- Reiko MAZUKA, Tadahisa KONDO, and Akiko HAYASHI (2008), Japanese mothers' use of specialized vocabulary in infant-directed speech: infant-directed vocabulary in Japanese, in *The origins of language*, pp. 39–58, Springer.
- R. Thomas MCCOY, Erin GRANT, Paul SMOLENSKY, Thomas L GRIFFITHS, and Tal LINZEN (2020), Universal linguistic inductive biases via meta-learning, *Proceedings of the 42nd Annual Conference of the Cognitive Science Society*.
- Richard Thomas MCCOY, Robert FRANK, and Tal LINZEN (2018), Revisiting the poverty of the stimulus: hierarchical generalization without a hierarchical bias in recurrent neural networks, in Chuck KALISH, Martina RAU, Jerry ZHU, and Timothy ROGERS, editors, *Proceedings of CogSci 2018*, pp. 2096–2101.
- Elliott MORETON, Brandon PRICKETT, Katya PERTSOVA, Josh FENNELL, Joe PATER, and Lisa SANDERS (2021), Learning repetition, but not syllable reversal, in Ryan BENNETT, Richard BIBBS, Mykel L. BRINKERHOFF, Max J. KAPLAN, Stephanie RICH, Amanda RYSLING, Nicholas VAN HANDEL, and Maya Wax CAVALLARO, editors, *Proceedings of the Annual Meetings on Phonology*.
- Breyne Arlene MOSKOWITZ (1975), The acquisition of fricatives: A study in phonetics and phonology, *Journal of Phonetics*, 3(3):141–150.
- Max NELSON, Hossep DOLATIAN, Jonathan RAWSKI, and Brandon PRICKETT (2020), Probing RNN Encoder-Decoder generalization of subregular functions using reduplication, *Proceedings of the Society for Computation in Linguistics (SCiL)*, pp. 31–42.
- Andrew NEVINS and Bert VAUX (2003), Metalinguistic, shmetalinguistic: The phonology of shmreduplication, in J. CIHLAR, A. FRANKLIN, D. KAISER, and

J. KIMBARA, editors, *Proceedings from the 39th Annual Meeting of the Chicago Linguistic Society*, pp. 702–721, Chicago Linguistic Society.

Steven PINKER and Alan PRINCE (1988), On language and connectionism: Analysis of a parallel distributed processing model of language acquisition, *Cognition*, 28(1):73–193.

Brandon PRICKETT (2019), Learning biases in opaque interactions, *Phonology*, 36(4):627–653, doi:10.1017/S0952675719000320.

Hugh RABAGLIATI, Brock FERGUSON, and Casey LEW-WILLIAMS (2019), The profile of abstract rule learning in infancy: Meta-analytic and experimental evidence, *Developmental Science*, 22:1–18.

Fariz RAHMAN (2016), seq2seq: Sequence to sequence learning with Keras, <https://github.com/farizrahman4u/seq2seq>.

D. Victoria RAU, Hui-Huan Ann CHANG, and Elaine E. TARONE (2009), Think or sink: Chinese learners' acquisition of the English voiceless interdental fricative, *Language Learning*, 59(3):581–621.

D.E. RUMELHART and J.L. MCCLELLAND (1986), On learning the past tenses of English verbs, in J.L. MCCLELLAND and D.E. RUMELHART, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 2: Psychological and Biological Models, pp. 216–271, The MIT Press.

Mark S. SEIDENBERG and Jeff L. ELMAN (1999), Do infants learn grammar with algebra or statistics?, *Science*, 284(5413):433.

Thomas R. SHULTZ and Alan C. BALE (2001), Neural network simulation of infant familiarization to artificial sentences: Rule-like behavior without explicit rules and variables, *Infancy*, 2(4):501–536.

Mary H. SKEEL (1969), Perceptual confusions among fricatives in preschool children., Technical report, The University of Wisconsin, <https://files.eric.ed.gov/fulltext/ED036789.pdf>.

Nitish SRIVASTAVA, Geoffrey HINTON, Alex KRIZHEVSKY, Ilya SUTSKEVER, and Ruslan SALAKHUTDINOV (2014), Dropout: A simple way to prevent neural networks from overfitting, *The Journal of Machine Learning Research*, 15(1):1929–1958.

Pavol ŠTEKAUER, Salvador VALERA, and Lívía KÖRTVÉLYESSY (2012), *Word-formation in the world's languages: a typological survey*, Cambridge University Press.

Joseph Paul STEMBERGER and Marshall LEWIS (1986), Reduplication in Ewe: Morphological accommodation to phonological errors, *Phonology*, 3:151–160.

Ilya SUTSKEVER, Oriol VINYALS, and Quoc V. LE (2014), Sequence to sequence learning with neural networks, in *Advances in neural information processing systems*, pp. 3104–3112.

Tijmen TIELEMAN and Geoffrey HINTON (2012), Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude, *COURSERA: Neural Networks for Machine Learning*, 4(2):26–31.

Guillermo VALLE-PEREZ, Chico Q CAMARGO, and Ard A. LOUIS (2018), Deep learning generalizes because the parameter-function map is biased towards simple functions, in Yoshua BENGIO and Yann LECUN, editors, *Proceedings of the 6th International Conference on Learning Representations*.

Rachelle WAKSLER (1999), Cross-linguistic evidence for morphological representation in the mental lexicon, *Brain and Language*, 68(1–2):68–74.

Yang WANG (2021), Recognizing reduplicated forms: Finite-state buffered machines, in Garrett NICOLAI, Kyle GORMAN, and Ryan COTTERELL, editors, *Proceedings of the 18th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pp. 177–187.

Janet F. WERKER and Richard C. TEES (1983), Developmental changes across childhood in the perception of non-native speech sounds, *Canadian Journal of Psychology/Revue Canadienne de Psychologie*, 37(2):278–286.

Colin WILSON (2019), Re (current) reduplication: Interpretable neural network models of morphological copying, *Proceedings of the Society for Computation in Linguistics (SCiL)*, 2:379–380.

Brandon Prickett

① 0000-0001-9217-2130
bprickett@umass.edu

Joe Pater

① 0000-0002-4784-3799
pater@linguist.umass.edu

Department of Linguistics,
University of Massachusetts Amherst

Aaron Traylor

① 0000-0002-0975-1914
aaron_traylor@brown.edu

Department of Computer Science,
Brown University

Brandon Prickett, Aaron Traylor, and Joe Pater (2022), *Learning reduplication with a neural network that lacks explicit variables*, *Journal of Language Modelling*, 10(1):1–38

① <https://dx.doi.org/10.15398/jlm.v10i1.274>

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

④ <http://creativecommons.org/licenses/by/4.0/>

Introduction to the special section on the interaction between formal and computational linguistics

*Timothée Bernard*¹ and *Grégoire Winterstein*²

¹ Université Paris Cité

² Université du Québec à Montréal (UQAM)

INTRODUCTION

1

While computational linguistics is historically rooted in formal linguistics, it might seem that the distance between these two fields has only grown larger as each field evolved. Still, whether this impression is correct or not, not all links have been cut, and new ones have appeared. Indeed, while we are currently witnessing a growing interest within formal linguistics in both explaining the remarkable successes of neural-based language models and uncovering their limitations, one should not forget the contribution to theoretical linguistics provided, for example, by the computational implementation of grammatical formalisms. And while neural-based methods have recently received the lion's share of the public attention, interpretable models based on symbolic methods are still relevant and widely used in the natural language processing industry.

The links that exist between formal and computational linguistics have been the subject of discussion for a long time. At the 2009 European Meeting of the Association for Computational Linguistics, a workshop entitled “Interaction between Linguistics and Computational Linguistics: Virtuous, Vicious or Vacuous?” was organised. This workshop led to the publication a couple of years later of the sixth volume of *Linguistic Issues in Language Technology* (Baldwin and Kordoni 2011). At the centre of this publication were discussions about

how and why formal linguistics and computational linguistics went down different paths, about the benefits and drawbacks of specialisation and about how our scientific communities could improve the situation. On this occasion, Church (2011) predicted that computational approaches to language would come back to symbolic approaches, observing that most of the “low hanging fruits” of statistical methods had already been picked. In a similar vein, Kay (2011) argued that natural language processing (NLP) – distinguished from computational linguistics in that the former was held to show little interest in language, and to be oriented towards pure performance matters – would disappear.

However, far from dwindling, statistical methods garnered renewed interest due to impressive advances in machine learning and, in particular, the progress made in the development of word embeddings generated as a product of the optimisation of neural-based language models (Mikolov *et al.* 2013b,a; Bengio *et al.* 2001). This stream of research eventually led to the apparition of the Transformer architecture (Vaswani *et al.* 2017), with famous implementations such as BERT (Devlin *et al.* 2019) and GPT-3 (Brown *et al.* 2020) which offer linguistic representations that are routinely used for a wide array of NLP applications, from classification to language generation. Though remarkably effective, with benchmark performances regularly smashed by newer and bigger models, the representations offered by these systems are largely shunned by the linguistic community, who often sees them as irrelevant to our understanding of language (see *infra*).

As already mentioned, it would, however, be a little hasty to declare a divorce between linguistic and computational methods. First, using computational methods to validate theoretical models remains common practice in many circles (e.g. among the LFG, HPSG or categorial grammar communities) and the use of such implementations can also be used to investigate and test typological hypotheses about language universals (such as with the LinGo Grammar Matrix; Bender *et al.* 2002). The use of symbolic methods also remains common in the industry, especially for applications for which humanely interpretable models are necessary (for various reasons including ethical ones; see Lipton 2018; Miller 2019 and references therein). Second, the properties of stochastic language models have also come under increasing

scrutiny. On the one hand, there is a lively debate about the ability of these models to properly represent natural language meaning (Bender and Koller 2020), and about how representative they are of the linguistic practices of the members of a linguistic community (Bender *et al.* 2021). On the other hand, there are efforts to explain the sheer effectiveness of these language models, in a way that goes beyond the mere mention of the distributional hypothesis (Gastaldi 2020), and to investigate how such models are sensitive (or not) to complex linguistic phenomena such as presupposition projection (Jiang and de Marneffe 2019) or syntactic generalisations (Hu *et al.* 2020) (see also the domain of “BERTology”, which seeks to study the properties of the representations manipulated by BERT-like models, though not necessarily from a linguistic angle; Ettinger 2020; Rogers *et al.* 2020).

OVERVIEW OF THE SPECIAL SECTION

2

Inspired by these tensions and connections, we organised a one-day online event on the interactions between formal and computational linguistics which took place in June 2021.¹ The guiding thread for the talks at that event was, roughly, to focus on and discuss recent advances in computational linguistics (be they symbolic or not), their relationship with linguistic data, and what such systems can do for language and linguistics itself. These questions were tackled from different angles: practical, theoretical and philosophical. The present special section takes its roots in that event, as we offered the presenters a chance to elaborate on the themes developed in the workshop in the form of long papers.

Both articles in this special section illustrate the theme of the seminar in two complementary ways, both in their use of computational methods to address theoretical issues of formal models of language,

¹See <https://gdr-lift.loria.fr/news/ilfc-en/>. The event then turned into a monthly online seminar <https://gdr-lift.loria.fr/monthly-online-ilfc-seminar/>.

and in the way they use linguistically inspired symbolic methods to achieve their goals.

In their paper, Olga Zamaraeva and her co-authors retrace the evolution of the “Grammar Matrix”, a meta-grammar engineering framework that relies on the HPSG (Pollard and Sag 1994) and MRS (Copestake *et al.* 2005) frameworks. The Grammar Matrix is a tool that automates the implementation of the grammar for a given language. To do so, the user provides information about the properties of the language they wish to implement a grammar of, along with a sample lexicon. On the basis of those properties and known analyses of the related phenomena in HPSG and MRS, the matrix is able to produce an implemented grammar that can be used, among other things, to test the coverage of the grammar on a set of sentences. Beyond that, the authors also highlight how the Grammar Matrix can be used to investigate cross-linguistic variation, and formulate and test general hypotheses about the structure of language. On the basis of a test set of sentences in 60 different languages from 40 distinct families, a regression testing system is used to check how modification in the analyses of phenomena affect the overall architecture of the system. The Grammar Matrix is thus a prime example of how computational methods can directly influence linguistic analysis, both as a tool to test such analyses, and as a way to get better insight about language using an approach that is both theoretically and empirically grounded.

Haruta *et al.* present the theoretical foundations and the practical implementation of an automatic Natural Language Inference (NLI) solver for English, i.e. a system that, given two input texts, aims at detecting whether the first entails, contradicts or is neutral toward, the second. One characteristic of their solver is that it is *symbolic*; while a recent popular approach in NLI (as in other NLP tasks) consists in training a classifier using only vector representations obtained via a language model (see *supra*), the system they describe relies on logical representations of the input texts produced by a parser and fed to a theorem prover. The various parsers they use are based on the Combinatory Categorical Grammar formalism (CCG; Steedman and Baldridge 2011); after a little bit of post-processing of the output trees, the logical representations are standardly derived from the syntactic analyses in a compositional fashion.

Obviously, the success of the enterprise crucially depends, among other things, on the expressive power of the logical language used. Haruta et al. have chosen to express the semantics of sentences in a version of First Order Logic (FOL) that incorporates events and integers/degrees, allowing them to translate a wide range of constructions involving adjectives, comparatives, generalised quantifiers and numerals. Key to many of their analyses is the notion of degree. For example, they analyse *Tom is taller than Mary* following the A-not-A analysis (see Schwarzschild 2008 and references therein) as meaning that there is some degree such that Tom has, but Mary has not, this degree of tallness. Haruta et al. evaluate their system on a large number of NLI datasets, including a novel one they have designed to cover the phenomena that they have been particularly interested in, usually absent from existing datasets such as FraCas (Cooper *et al.* 1996). Results show that, in general, non-symbolic models perform significantly worse than state-of-the-art symbolic models, and that, in particular, the system presented here is particularly effective. This very interesting paper thus contributes to showing that formal syntax and semantics are still relevant to natural language processing and that, in some domains, symbolic reasoning is still one step ahead of the purely neuronal alternatives that have progressively taken the spotlight in the last decade.

ACKNOWLEDGMENTS

The workshop and seminar out of which this special section grew was logistically supported by the “Groupe de Recherche en Linguistique Informatique, Formelle et de Terrain” (GdR LIFT, <https://gdr-lift.loria.fr/>), which is a structure funded by the French national scientific research agency in a effort to foster discussion among three communities related to language: the communities of field linguistics, formal linguistics and computational linguistics.

REFERENCES

- Timothy BALDWIN and Valia KORDONI (2011), The Interaction between Linguistics and Computational Linguistics, *Linguistic Issues in Language Technology*, 6, doi:10.33011/lilt.v6i.1233, <https://journals.colorado.edu/index.php/lilt/article/view/1233>.
- Emily M. BENDER, Dan FLICKINGER, and Stephan OEPEN (2002), The Grammar Matrix: An Open-Source Starter-Kit for the Rapid Development of Cross-Linguistically Consistent Broad-Coverage Precision Grammars, in John CARROLL, Nelleke OOSTDIJK, and Richard SUTCLIFFE, editors, *Proceedings of the Workshop on Grammar Engineering and Evaluation at the 19th International Conference on Computational Linguistics*, pp. 8–14, Taipei, Taiwan.
- Emily M. BENDER, Timnit GEBRU, Angelina MCMILLAN-MAJOR, and Shmargaret SHMITCHELL (2021), On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?, in *Proceedings of FAccT 2021*, pp. 610–623.
- Emily M. BENDER and Alexander KOLLER (2020), Climbing towards NLU: On Meaning, Form, and Understanding in the Age of Data, in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 5185–5198, Association for Computational Linguistics, Online, <https://www.aclweb.org/anthology/2020.acl-main.463>.
- Yoshua BENGIO, Réjean DUCHARME, and Pascal VINCENT (2001), A Neural Probabilistic Language Model, in T. K. LEEN, T. G. DIETTERICH, and V. TRESP, editors, *Advances in Neural Information Processing Systems 13*, pp. 932–938, MIT Press, <http://papers.nips.cc/paper/1839-a-neural-probabilistic-language-model.pdf>.
- Tom BROWN, Benjamin MANN, Nick RYDER, Melanie SUBBIAH, Jared D. KAPLAN, Prafulla DHARIWAL, Arvind NEELAKANTAN, Pranav SHYAM, Girish SASTRY, Amanda ASKELL, Sandhini AGARWAL, Ariel HERBERT-VOSS, Gretchen KRUEGER, Tom HENIGHAN, Rewon CHILD, Aditya RAMESH, Daniel ZIEGLER, Jeffrey WU, Clemens WINTER, Chris HESSE, Mark CHEN, Eric SIGLER, Mateusz LITWIN, Scott GRAY, Benjamin CHESS, Jack CLARK, Christopher BERNER, Sam MCCANDLISH, Alec RADFORD, Ilya SUTSKEVER, and Dario AMODEI (2020), Language Models are Few-Shot Learners, in H. LAROCHELLE, M. RANZATO, R. HADSELL, M.F. BALCAN, and H. LIN, editors, *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901, Curran Associates, Inc., <https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfc4967418bfb8ac142f64a-Paper.pdf>.
- Kenneth CHURCH (2011), A Pendulum Swung too Far, *Linguistic Issues in Language Technology*, 6, <https://journals.colorado.edu/index.php/lilt/article/view/1245>.

Introduction to the special section

Robin COOPER, Dick CROUCH, Jan VAN EIJK, Chris FOX, Josef VAN GENABITH, Jan JASPARS, Hans KAMP, David MILWARD, Manfred PINKAL, Massimo POESIO, and Steve PULMAN (1996), Using the Framework, Technical Report Deliverable D16, <https://gu-clasp.github.io/multifracas/D16.pdf>.

Ann COPESTAKE, Dan FLICKINGER, Ivan SAG, and Carl POLLARD (2005), Minimal Recursion Semantics: An Introduction, *Research in Language and Computation*, 3(2–3):281–332.

Jacob DEVLIN, Ming-Wei CHANG, Kenton LEE, and Kristina TOUTANOVA (2019), BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Association for Computational Linguistics, Minneapolis, Minnesota, doi:10.18653/v1/N19-1423, <https://www.aclweb.org/anthology/N19-1423>, 04472.

Allyson ETTINGER (2020), What BERT Is Not: Lessons from a New Suite of Psycholinguistic Diagnostics for Language Models, *Transactions of the Association for Computational Linguistics*, 8:34–48, doi:doi.org/10.1162/tacla00298.

Juan Luis GASTALDI (2020), Why Can Computers Understand Natural Language?, *Philosophy & Technology*, doi:10.1007/s13347-020-00393-9.

Jennifer HU, Jon GAUTHIER, Peng QIAN, Ethan WILCOX, and Roger P. LEVY (2020), A Systematic Assessment of Syntactic Generalization in Neural Language Models, in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 1725–1744.

Nanjiang JIANG and Marie-Catherine DE MARNEFFE (2019), Evaluating BERT for Natural Language Inference: A Case Study on the CommitmentBank, in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP 2019)*, pp. 6086–6091, doi:10.18653/v1/D19-1630, <https://aclanthology.org/D19-1630>.

Martin KAY (2011), Zipf's Law and *L'Arbitraire du Signe*, *Linguistic Issues in Language Technology*, 6, <https://journals.colorado.edu/index.php/lilt/article/view/1251>.

Zachary C. LIPTON (2018), The Mythos of Model Interpretability, *ACM Queue*, 16(3):1–27, doi:10.1145/3236386.3241340, <http://doi.acm.org/10.1145/3236386.3241340>.

Tomas MIKOLOV, Kai CHEN, Greg CORRADO, and Jeffrey DEAN (2013a), Efficient Estimation of Word Representations in Vector Space, in Yoshua BENGIO and Yann LECUN, editors, *1st International Conference on Learning*

Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings, <http://arxiv.org/abs/1301.3781>.

Tomas MIKOLOV, Ilya SUTSKEVER, Kai CHEN, Greg S. CORRADO, and Jeff DEAN (2013b), Distributed Representations of Words and Phrases and their Compositionality, in C. J. C. BURGES, L. BOTTOU, M. WELLING, Z. GHAMRANI, and K. Q. WEINBERGER, editors, *Advances in Neural Information Processing Systems 26*, pp. 3111–3119, Curran Associates, Inc., <http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf>.

Tim MILLER (2019), Explanation in Artificial Intelligence: Insights from the Social Sciences, *Artificial Intelligence*, 267:1–38, doi:10.1016/j.artint.2018.07.007, <http://www.sciencedirect.com/science/article/pii/S0004370218305988>.

Carl J. POLLARD and Ivan A. SAG (1994), *Head-Driven Phrase Structure Grammar*, University of Chicago Press, Chicago.

Anna ROGERS, Olga KOVALEVA, and Anna RUMSHISKY (2020), A Primer in BERTology: What We Know About How BERT Works, *Transactions of the Association for Computational Linguistics*, 8:842–866, doi:10.1162/tacl_a_00349, <https://aclanthology.org/2020.tacl-1.54>.

Roger SCHWARZSCHILD (2008), The Semantics of Comparatives and Other Degree Constructions, *Language and Linguistics Compass*, 2(2):308–331, doi:10.1111/j.1749-818X.2007.00049.x, <https://onlinelibrary.wiley.com/doi/10.1111/j.1749-818X.2007.00049.x>.

Mark STEEDMAN and Jason BALDRIDGE (2011), Combinatory Categorical Grammar, in Robert D. BORSLEY and Kersti BÖRJARS, editors, *Non-Transformational Syntax*, pp. 181–224, Wiley-Blackwell, ISBN 978-1-4443-9503-7, doi:10.1002/9781444395037.ch5, <http://onlinelibrary.wiley.com/doi/10.1002/9781444395037.ch5/summary>.

Ashish VASWANI, Noam SHAZEER, Niki PARMAR, Jakob USZKOREIT, Llion JONES, Aidan N. GOMEZ, Łukasz KAISER, and Illia POLOSUKHIN (2017), Attention is All you Need, in I. GUYON, U. V. LUXBURG, S. BENGIO, H. WALLACH, R. FERGUS, S. VISHWANATHAN, and R. GARNETT, editors, *Advances in Neural Information Processing Systems 30*, pp. 5998–6008, Curran Associates, Inc., <http://papers.nips.cc/paper/7181-attention-is-all-you-need>.

Introduction to the special section

Timothée Bernard

ORCID 0000-0003-4172-6986

timothee.bernard@u-paris.fr

Laboratoire de linguistique formelle
(LLF)

Université de Paris

8 place Paul Ricœur

75205 Paris Cedex 13, France

Grégoire Winterstein

ORCID 0000-0002-8951-2138

winterstein.gregoire@uqam.ca

Université du Québec à Montréal
(UQAM)

Timothée Bernard and Grégoire Winterstein (2022), *Introduction to the special section on the interaction between formal and computational linguistics*, *Journal of Language Modelling*, 10(1):39–47

DOI <https://dx.doi.org/10.15398/jlm.v10i1.325>

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

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

20 years of the Grammar Matrix: cross-linguistic hypothesis testing of increasingly complex interactions

Olga Zamaraeva¹, Chris Curtis², Guy Emerson³, Antske Fokkens^{4,5},
Michael Wayne Goodman⁶, Kristen Howell⁶, T.J. Trimble,
and Emily M. Bender⁷

¹ Universidade da Coruña

² Firemuse Research

³ University of Cambridge

⁴ Vrije Universiteit Amsterdam

⁵ Eindhoven University of Technology

⁶ LivePerson Inc.

⁷ University of Washington

ABSTRACT

The Grammar Matrix project is a meta-grammar engineering framework expressed in Head-driven Phrase Structure Grammar (HPSG) and Minimal Recursion Semantics (MRS). It automates grammar implementation and is thus a tool and a resource for linguistic hypothesis testing at scale. In this paper, we summarize how the Grammar Matrix grew in the last decade and describe how new additions to the system have made it possible to study interactions between analyses, both monolingually and cross-linguistically, at new levels of complexity.

Keywords: HPSG, grammar engineering, hypothesis testing, typology

INTRODUCTION

1

From its beginnings in 2001, the Grammar Matrix project (Bender *et al.* 2002, 2010, among others)¹ has investigated how grammar engineer-

¹ Olga Zamaraeva contributed the constituent questions and the clausal complements libraries to the Grammar Matrix framework and led the writing of this

ing can support research into cross-linguistic variation and similarity. Key to this approach has been the potential of computational implementation to handle complexity, in both data and analyses. In this paper we take stock of work on and with the Grammar Matrix since 2010, explore how that potential has been leveraged and envision future directions.

The Grammar Matrix is a meta-grammar engineering framework expressed in the Head-driven Phrase Structure Grammar formalism (HPSG; Pollard and Sag 1994), specifically in one particular restricted version of it (Copestake 2002a). Grammar engineering is a discipline and a methodology concerned with a particular empirical approach to modelled linguistic knowledge (Bierwisch 1963; Zwicky *et al.* 1965; Müller 1999; Butt *et al.* 1999; Bender 2008; Müller 2015); namely, grammar *modelling* and *testing*. *Modelling* grammar in this context means coming up with sets of grammar entities (in this case: types, rules and lexical entries) and implementing them as a computer program which can accept or reject strings by attempting (and either succeeding or failing) to find a syntactic structure that can correspond to the input string. *Testing* (analogous to “competence profiling” as defined by Oepen (2002, page 89)) means deploying this grammar program (usually along with a separate parser program) on a list of sentences and then assessing whether or not the grammar indeed *correctly* parsed all grammatical sentences and rejected all the ungrammatical ones – an alternative to doing the testing with pen and paper, performing computations in one’s head. *Correctly* here means that each structure assigned by the grammar to any grammatical string is in fact a correct linguistic representation of it. For the purposes of the

paper. Emily M. Bender is the initial developer and the long-time lead of the Grammar Matrix project, and the principal investigator of the National Science Foundation grants for both the Grammar Matrix and AGGREGATION projects. The rest of the authors are in alphabetical order. Chris Curtis contributed the valence change library to the project; Guy Emerson is the author of append-lists and computation types; Antske Fokkens is the author of CLIMB and contributed to the word order library; Michael W. Goodman contributed to the morphotactics library and to the regression test system; Kristen Howell contributed the clausal modifiers and the nominalized clauses libraries and contributed to the AGGREGATION project. T. J. Trimble added the support for adjectives and copulas and made other contributions to the lexicon component of the Grammar Matrix.

Grammar Matrix, this last assessment is done with respect not to the syntactic tree but to the resulting sentence semantics that is directly paired with the syntactic structure. In our case, the semantics is encoded in the Minimal Recursion Semantics formalism (MRS; Copestake *et al.* 2005). In terms of practical set up, the Grammar Matrix allows the linguist-user to enter a typological, lexical, and morphological description of a grammar via a web-based questionnaire, and obtain a grammar fragment implemented in HPSG automatically. This, in turn, allows this linguist to test the set of hypotheses that the grammar encodes against the data stored in text files, in a computer-aided fashion.

We see the Grammar Matrix as a flexible framework for building up, over time and in a data-driven fashion, a set of analyses which are demonstrably useful for describing the repertoire of grammatical variation in the world's languages. Our conviction is rooted in three properties of the framework: (i) the Grammar Matrix design is informed by typological literature (while relying on established HPSG concepts); (ii) the development methodology prioritizes cross-linguistic applicability of the analyses and as such leaves flexibility to define HPSG features motivated by the data; (iii) for any new analyses proposed for inclusion in the Grammar Matrix, there is a system in place which allows one to automatically test the new analyses in integration with the existing ones (Bender *et al.* 2007). Long term, this builds and extends a system of analyses for which there is a demonstrated area of applicability – which also grows over time.

The paper is structured as follows. In Section 2, we describe the syntactic formalism which the Grammar Matrix uses and the grammar engineering philosophy which it follows. Section 3 gives a summary of the additions to the Grammar Matrix since 2010 and describes the development methodologies, including a detailed description of how the analyses which are part of the Grammar Matrix are being tested as a holistic system. In Section 4, we discuss several specific examples of how the Grammar Matrix helps identify tensions between different analyses, while Section 5 gives examples of the analyses which have proven particularly robust over the years. The paper concludes with the discussion of how the Grammar Matrix serves as infrastructure to support other research projects (Section 6) and a look ahead to future directions (Section 7).

This section is intended to give the reader basic background on HPSG (Section 2.1) and grammar engineering (Section 2.2), and on the particular version of the HPSG formalism that the Grammar Matrix project uses (Section 2.3). Section 2.3 also describes several major grammar engineering projects and initiatives, some using HPSG and some using other formalisms.

Head-driven Phrase Structure Grammar (Pollard and Sag 1994) is a syntactic framework characterized by a sign-based approach to grammar,² the use of formally precise constraint-based formalisms (Carpenter 2005), and an emphasis (shared with Construction Grammar (Fillmore *et al.* 1988) and especially Sign-Based Construction Grammar (Sag *et al.* 2012)) on modelling both the broad generalizations at play in a given language and the rich details of lexical and constructional idiosyncrasy in a single, coherent grammar. Multiple inheritance hierarchies serve as the central device to capture generalizations in HPSG. Like any grammatical framework, HPSG encompasses a variety of related theoretical proposals and also has multiple competing formalisms (for some discussion, see Richter 2021). Despite this variety, the formalisms used in HPSG are relatively stable over time, making possible the development of software which implements those formalisms and can be used for sustained grammar development (for further discussion, see Bender and Emerson 2021).

HPSG formalisms are based on constraint unification. In constraint unification, variables may be constrained to have a particular value or to be equal to the value of another variable. In order for any two types to unify, there must be a single (unique) type in the hierarchy which represents their combination (Copestake 2002b, page 42). Ultimately, an HPSG parser checks whether, given a tokenized sentence string and a set of types and lexical entries that represent the

²See e.g. Pollard and Sag 1987, page 2 for a summary of de Saussure’s (1916) theory of language signs.

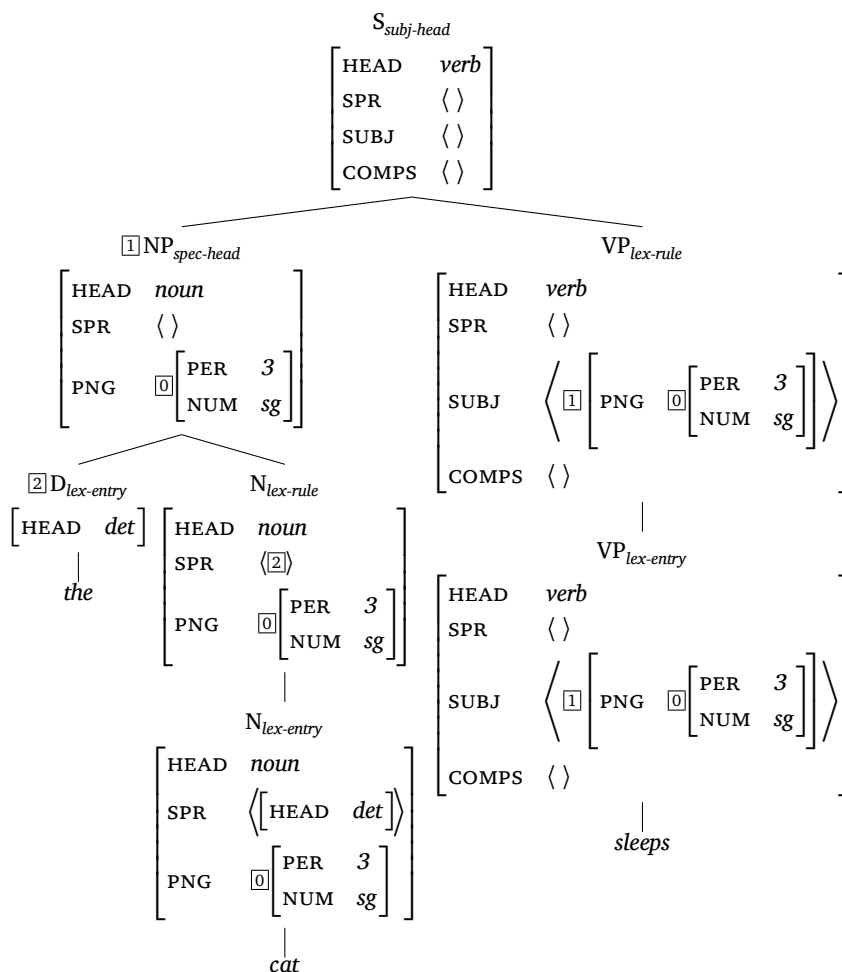


Figure 1: An HPSG derivation visualized as a tree

grammar, it can find a feature structure such that each token corresponds to some lexical entry and together they form some syntactic structure in which all the constraints dictated by the grammar unify.

As a simplified example, consider a tree of feature structures in Figure 1 representing an HPSG parse for the English [eng] (Indo-European) sentence (1).³ The feature structures in the tree are visu-

³This tree is a simplified version of a tree produced by a grammar of English that was output by the Grammar Matrix. In particular, we only show the features

alized as attribute-value matrices (AVMs). This tree includes only selected feature-value pairs and substructure sharing tags (\square etc.), to illustrate the particular role of each node in the tree that we would like to emphasize here for purposes of exposition.

(1) The cat sleeps. [eng]

Consider the tree in Figure 1 bottom-up, along with related examples (2) and (3) located on page 55. Suppose that the terminal nodes in the tree (corresponding to the lexemes *the*, *cat*, and *sleep*) are provided by some lexicon. The lexical entries in that lexicon are instances of lexical types and specify, among other things, the syntactic category of each word (such as noun or verb) and what arguments they require, if any. For example, the intransitive verb *sleep* requires exactly zero complements and one subject element; furthermore, it requires an NP subject. The noun *cat* has a PNG feature which in turn has PER and NUM features appropriate for it, the values of which in this particular lexical entry are specified to be PER 3 at the lexical entry level, NUM underspecified to just *number* in the lexical type to which the lexical entry belongs (2), and further specified to NUM *sg* after a lexical rule (3) applies. The SYNSEM feature in the lexical rule (3) is the “mother” of the unary rule; the DTR feature is the “daughter”.⁴ Note that while the NUM value is identified between the mother and the daughter in the lexical rule, a lexical entry like (2) can unify with the daughter of (3) because its own value is underspecified. In the fully specified tree in Figure 1, the NUM has already been identified with the mother’s NUM in the lexical rule (same with the SUBJ identity between the verb’s lexical entry node and the verb’s lexical rule node).

HEAD, SPR (specifier), SUBJ (subject), COMPS (complements), and PNG (person, number, gender), with only NUM and PER within the last of these (whereas in reality, there is also GEN). The tree and the explanation are adapted from Zamaraeva 2021a, page 33.

⁴This lexical rule does not have an overt grammatical marking (these are sometimes called “zero-marking” rules); a lexical rule for plural marking would add the affix *s* to the orthography.

(2)	<table style="border-collapse: collapse; width: 100%;"> <tr> <td colspan="2" style="padding: 5px;"><i>noun-lex</i></td> </tr> <tr> <td style="padding: 5px;">HEAD</td> <td style="padding: 5px;"><i>noun</i></td> </tr> <tr> <td style="padding: 5px;">PNG</td> <td style="border-left: 1px solid black; border-right: 1px solid black; padding: 5px;"> <table style="border-collapse: collapse; width: 100%;"> <tr> <td style="padding: 5px;">PER</td> <td style="padding: 5px;">3</td> </tr> <tr> <td style="padding: 5px;">NUM</td> <td style="padding: 5px;"><i>number</i></td> </tr> </table> </td> </tr> <tr> <td style="padding: 5px;">SPR</td> <td style="border-left: 1px solid black; border-right: 1px solid black; padding: 5px;"> <table style="border-collapse: collapse; width: 100%;"> <tr> <td style="padding: 5px;">HEAD</td> <td style="padding: 5px;"><i>det</i></td> </tr> </table> </td> </tr> </table>	<i>noun-lex</i>		HEAD	<i>noun</i>	PNG	<table style="border-collapse: collapse; width: 100%;"> <tr> <td style="padding: 5px;">PER</td> <td style="padding: 5px;">3</td> </tr> <tr> <td style="padding: 5px;">NUM</td> <td style="padding: 5px;"><i>number</i></td> </tr> </table>	PER	3	NUM	<i>number</i>	SPR	<table style="border-collapse: collapse; width: 100%;"> <tr> <td style="padding: 5px;">HEAD</td> <td style="padding: 5px;"><i>det</i></td> </tr> </table>	HEAD	<i>det</i>	(3)	<table style="border-collapse: collapse; width: 100%;"> <tr> <td colspan="2" style="padding: 5px;"><i>sg-lex-rule</i></td> </tr> <tr> <td style="padding: 5px;">SYNSEM</td> <td style="border-left: 1px solid black; border-right: 1px solid black; padding: 5px;"> <table style="border-collapse: collapse; width: 100%;"> <tr> <td style="padding: 5px;">ORTH</td> <td style="padding: 5px;">0</td> </tr> <tr> <td style="padding: 5px;">HEAD</td> <td style="padding: 5px;"><i>noun</i></td> </tr> <tr> <td style="padding: 5px;">PNG NUM</td> <td style="padding: 5px;">1 sg</td> </tr> </table> </td> </tr> <tr> <td style="padding: 5px;">DTR</td> <td style="border-left: 1px solid black; border-right: 1px solid black; padding: 5px;"> <table style="border-collapse: collapse; width: 100%;"> <tr> <td style="padding: 5px;">ORTH</td> <td style="padding: 5px;">0</td> </tr> <tr> <td style="padding: 5px;">HEAD</td> <td style="padding: 5px;"><i>noun</i></td> </tr> <tr> <td style="padding: 5px;">PNG NUM</td> <td style="padding: 5px;">1</td> </tr> </table> </td> </tr> </table>	<i>sg-lex-rule</i>		SYNSEM	<table style="border-collapse: collapse; width: 100%;"> <tr> <td style="padding: 5px;">ORTH</td> <td style="padding: 5px;">0</td> </tr> <tr> <td style="padding: 5px;">HEAD</td> <td style="padding: 5px;"><i>noun</i></td> </tr> <tr> <td style="padding: 5px;">PNG NUM</td> <td style="padding: 5px;">1 sg</td> </tr> </table>	ORTH	0	HEAD	<i>noun</i>	PNG NUM	1 sg	DTR	<table style="border-collapse: collapse; width: 100%;"> <tr> <td style="padding: 5px;">ORTH</td> <td style="padding: 5px;">0</td> </tr> <tr> <td style="padding: 5px;">HEAD</td> <td style="padding: 5px;"><i>noun</i></td> </tr> <tr> <td style="padding: 5px;">PNG NUM</td> <td style="padding: 5px;">1</td> </tr> </table>	ORTH	0	HEAD	<i>noun</i>	PNG NUM	1
<i>noun-lex</i>																																			
HEAD	<i>noun</i>																																		
PNG	<table style="border-collapse: collapse; width: 100%;"> <tr> <td style="padding: 5px;">PER</td> <td style="padding: 5px;">3</td> </tr> <tr> <td style="padding: 5px;">NUM</td> <td style="padding: 5px;"><i>number</i></td> </tr> </table>	PER	3	NUM	<i>number</i>																														
PER	3																																		
NUM	<i>number</i>																																		
SPR	<table style="border-collapse: collapse; width: 100%;"> <tr> <td style="padding: 5px;">HEAD</td> <td style="padding: 5px;"><i>det</i></td> </tr> </table>	HEAD	<i>det</i>																																
HEAD	<i>det</i>																																		
<i>sg-lex-rule</i>																																			
SYNSEM	<table style="border-collapse: collapse; width: 100%;"> <tr> <td style="padding: 5px;">ORTH</td> <td style="padding: 5px;">0</td> </tr> <tr> <td style="padding: 5px;">HEAD</td> <td style="padding: 5px;"><i>noun</i></td> </tr> <tr> <td style="padding: 5px;">PNG NUM</td> <td style="padding: 5px;">1 sg</td> </tr> </table>	ORTH	0	HEAD	<i>noun</i>	PNG NUM	1 sg																												
ORTH	0																																		
HEAD	<i>noun</i>																																		
PNG NUM	1 sg																																		
DTR	<table style="border-collapse: collapse; width: 100%;"> <tr> <td style="padding: 5px;">ORTH</td> <td style="padding: 5px;">0</td> </tr> <tr> <td style="padding: 5px;">HEAD</td> <td style="padding: 5px;"><i>noun</i></td> </tr> <tr> <td style="padding: 5px;">PNG NUM</td> <td style="padding: 5px;">1</td> </tr> </table>	ORTH	0	HEAD	<i>noun</i>	PNG NUM	1																												
ORTH	0																																		
HEAD	<i>noun</i>																																		
PNG NUM	1																																		

All types in the grammar are part of the type hierarchy, with subtypes inheriting all constraints of their supertypes. Each type is specified to have features appropriate for it, and each subtype of a type may set the values for those features (which in turn are constrained to be specific types). HPSG uses the type hierarchy to define feature appropriateness, to constrain which feature structures can unify with each other, and to capture generalizations. In that final function, the type hierarchy supports compactness and elegance and thus maintainability and scalability of grammars.

HPSG theory is characterized by rich lexical types and relatively schematic phrase structure rules. The properties of any given node in a tree are established by combining constraints from lexical entries and rules, including constraints which propagate information through the tree. When information is identified between different parts of feature structure or tree, this is called *structure sharing*. One example is the Head Feature Principle (Pollard and Sag 1994, page 31), which stipulates that the value of the feature HEAD (including all feature-value pairs inside HEAD) in a phrase licensed by a headed rule must be shared between the mother and the head daughter. Accordingly, for a phrase structure rule, the grammarian must indicate if it is a headed rule and, if so, which daughter is the head daughter. In Figure 1, the HEAD category is propagated because the Head Feature Principle is implemented in the grammar. Other information is propagated because the particular phrase structure or lexical rules are defined specifically to do that; for example, the head-specifier rule identifies the non-head daughter with the sole element on the SPR list of the head daughter in Figure 1, and so on.

Structure sharing means some parts of the structure are the same. Any feature structure can also be visualized as a graph (see Pollard and

Sag 1994, pages 16–17), and indeed graph data structures provide a convenient and frequently used implementation of feature structures. In such cases, identity tags in the AVM notation correspond to reentrancies in the graph, meaning the arcs will converge in the exact same place. In other words, in Figure 1 the subject of the head daughter and the entire non-head daughter are not only similar (identical); they are literally the same structure.

The notion of structure-sharing is closely related to the notion of constraint unification generally, and to *unification failure*, which is the mechanism which leads to HPSG grammars rejecting ungrammatical input, or in other words not generating ungrammatical strings. Suppose the same HPSG grammar that licenses sentence (1) by assigning it the structure in Figure 1 is given the string (4) as input instead.

(4) *The cats sleeps.

In order for the grammar to license the plural orthography *cats*, the instance of the lexical entry for *cat* had to go through a lexical rule which specifies its value as *pl*. This means that if the grammar attempts to use the head-subject rule to license (4), there will be a unification failure between the verb's expected subject's PNG value and the one specified for the noun phrase, as illustrated in Figure 2.

Finally, one other thing about the HPSG formalism that is important for understanding this paper is the notion of *list*, seen in Figure 1 as the value type for SPR, SUBJ, and COMPS (specifier, subject, and complement lists; the list notation being the angle brackets $\langle \rangle$). List is a type in the type hierarchy, just like everything else. Lists are convenient for modelling different parts of grammar, most notably the notion of children of a node in the tree, and also arguments (e.g. of a verb).

2.2

Grammar engineering

Grammar engineering is the implementation of formal precision grammars as computer programs such that parsing and generation can be done automatically. Precision grammars are machine-readable models of language which encode notions of grammaticality and linguistic knowledge. The concept of precision grammar engineering arises

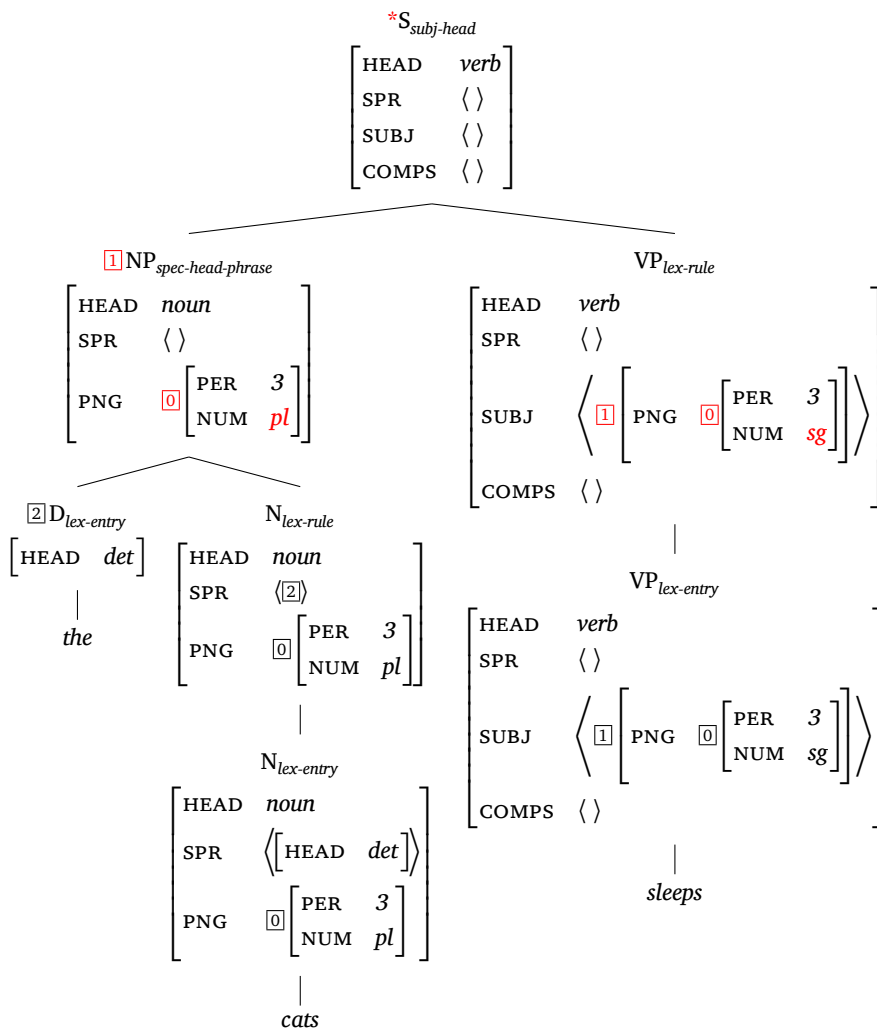


Figure 2: Unification failures (red) visualized as an impossible derivation tree. The asterisk (*) in the top node signifies that this node is impossible given the constraints

naturally from the idea that modelling grammar is akin to writing a computer program that accepts or rejects strings. An important characteristic of a grammar engineering system is rigor: it actually implements the grammar-program idea on the computer, precluding human mistakes that are due to e.g. human operational memory constraints or inconsistency of attention. It was suggested at least as

early as in Bierwisch 1963 that, without computational aid, the task of tracking how exactly multiple complex analyses interact with each other (and therefore how exactly even a small change in an analysis affects the grammar) becomes virtually impossible.⁵

One of the biggest benefits of grammar engineering projects like the Grammar Matrix, the focus of this paper, is that they allow us to empirically test syntactic theories by creating explicit models of them on a computer and then deploying those models on test suites of data from human languages. Engineered grammars make it much harder for grammarians to fool themselves into thinking that the grammar (a set of syntactic analyses) covers something it actually does not cover. The computer will definitively show which strings from the test set are parsed and which are not, and the grammarians will then be left with the task of investigating any failures. Conversely, at any point the grammarians can be confident in stating that the grammar covers a specific set of strings, namely the ones in the test suites which the grammar actually parsed. A complete system of analyses covering the entire set of human languages remains a very distant goal, and the field proceeds towards it in steps, carefully documenting issues along the way. This can thus be seen as a practical implementation of the Montagovian method of fragments (Montague 1974; Partee 1979; Gazdar *et al.* 1985).

The grammar engineering landscape includes multiple projects carried out in various formalisms. The Grammar Matrix is expressed in one particular version of HPSG developed by the DEep Linguistic Processing with Hpsg INitiative (DELPH-IN, Section 2.3). In addition to DELPH-IN projects, there are other implementations based on the ideas of HPSG, including PAGE (later DISCO) (Uszkoreit *et al.* 1994), ALE (Penn 2000) and its successor TRALE (Meurers *et al.* 2002; Penn 2004; Müller 2007), LIGHT (Ciortuz 2002; Ciortuz and Saveluc 2012), Alpino (Bouma *et al.* 2001b; van Noord 2006, focussing on Dutch), and Enju (Miyao and Tsujii 2008, focussing on probabilistic disambiguation). A grammar engineering project similar in some ways to the Grammar Matrix, called CoreGram (Müller 2015), uses

⁵ See Fokkens 2014, page 13 for a discussion of Bierwisch 1963 in English and Müller 2015, page 34 for similar discussion and for excerpts translated from German into English.

TRALE's version of HPSG. In Lexical Functional Grammar (LFG; Kaplan and Bresnan 1982), ParGram (Butt and King 2002) is an analogous project. In Minimalism, there is implementation work associated with the strongly lexicalized version of the formalism introduced by Stabler (1997), e.g. Graf and Kostyszyn 2021 and Torr 2018. Candito (1999) proposes a metagrammar for creating French and Italian grammars using the Lexicalized Tree-Adjoining Grammar (LTAG) formalism (Joshi and Schabes 1997). This approach was further developed into the eXtensible MetaGrammar (Crabbé *et al.* 2013, XMG). Clément and Kinyon (2003) propose a metagrammar for generating LFG grammars, inspired by Candito's work. Ranta (2011) implements complex syntactic structures in the multilingual Grammatical Framework Resource Grammar Library. This resource supports the development of grammars for natural language processing (NLP) applications that consist of simple rules that inherit the more complex foundations of the Resource Grammar Library. OpenCCG (Baldrige *et al.* 2007) provides a grammar engineering framework for Combinatory Categorical Grammar (Steedman 2000).⁶

DELPH-IN consortium and formalism

2.3

DELPH-IN⁷ is an international consortium of researchers interested in developing implemented grammars with HPSG and MRS and deploying them in the context of practical applications. DELPH-IN produces software support for grammar engineering, grammars, and applications built on grammars, all of which are open source. The software support includes grammar development environments (of which the most widely used is the LKB (Copestake 2002b)), parsing and/or generation engines (the LKB, as well as PET (Callmeier 2000), agree (Slayden 2012), and ACE (Crysmann and Packard 2012)), treebanking

⁶ Perhaps the strongest current influence of grammar engineering on the rest of the field of NLP is through treebanks. Treebanks are collections of syntactically annotated corpora on which machine learning systems can train. All treebanks were initially either produced by manual annotation, with annotators relying on a linguistic formalism, or using an engineered grammar and manual parse selection.

⁷ <http://www.delph-in.net>, <https://github.com/delph-in/>

platforms (Oepen *et al.* 2004; Packard 2015),⁸ grammar coverage and efficiency profiling facilities (Oepen 2002), Python libraries for a wide variety of data manipulation tasks (Goodman 2019), and the Grammar Matrix meta-grammar engineering toolkit (Bender *et al.* 2002, 2010) which is the focus of this paper.

By far the largest DELPH-IN grammar is the English Resource Grammar (ERG; Flickinger 2000, 2011), but DELPH-IN work has been multilingual from the consortium’s inception in 2002, and the original motivation of the Grammar Matrix was to support the development of grammars for many languages which are interoperable with the same grammar development and application software (Bender *et al.* 2002). Applications developed with DELPH-IN grammatical resources include machine translation (e.g. Oepen *et al.* 2007; Bond *et al.* 2011), computer-assisted language learning (Flickinger and Yu 2013; Suppes *et al.* 2014; Morgado da Costa *et al.* 2016, 2020), and summarization (Fang *et al.* 2016). For further discussion of applications, see Bender and Emerson 2021, Section 4.2.

Important to the success of the DELPH-IN international consortium is the coordination at the level of formalisms. The particular variant of the typed-feature structure formalism used in DELPH-IN (Copestake 2002a) is dubbed the DELPH-IN Joint Reference Formalism (DELPH-IN JRF) and builds on Type Description Language (TDL; Krieger and Schäfer 1994) as its predecessor. A key design decision in the DELPH-IN JRF is to keep the formalism simple by disallowing e.g. set-valued and disjunctive features as well as relational constraints.⁹ These restrictions ensure that unification in any grammar will yield a unique well-formed feature structure (if it exists) (Copestake 2002a, page 230), reducing parsing and generation to well-formed unification and allowing for efficient algorithms leading to faster processing times.¹⁰

⁸In the context of precision grammars, treebanking refers to manually selecting and storing the linguistically correct tree(s) from the “forest” of all trees provided for a sentence by the grammar and the parser.

⁹Relational means the value of a feature may be constrained to be the result of an operation over some other features’ values.

¹⁰For example, eliminating feature value disjunctions in favour of explicit encoding via underspecified types preserves generality (Flickinger 2000, pages 18–24) while allowing unification methods to be optimized to simplify feature

This section described the research and engineering landscape in which the Grammar Matrix exists and is being developed. The Grammar Matrix uses the HPSG theory of syntax (Pollard and Sag 1994; Müller *et al.* 2021) with a deliberately restricted version of the formalism (Copestake 2002a) and Minimal Recursion Semantics for semantic representations (Copestake *et al.* 2005). Generally, Grammar Matrix developers subscribe to the grammar engineering philosophy based on the Montagovian method of fragments (Montague 1974) and are accumulating a complex cross-linguistic system of grammatical analyses while maintaining empirical rigor.

THE GRAMMAR MATRIX: TWO DECADES OF CONTINUOUS DEVELOPMENT AND RESEARCH

The Grammar Matrix (Bender *et al.* 2002, 2010)¹¹ is a DELPH-IN-based meta-grammar engineering framework that includes a web-based questionnaire,¹² a core HPSG grammar, and a grammar customization system programmed in Python.¹³ A user fills out a questionnaire with typological, lexical, and morphological information about a language, and, based on the particular combination of their choices, the system applies the customization logic to output a grammar fragment which includes the core as well as additional, custom types, custom lexical entries, and custom rules. This grammar can be used to parse and generate sentences from the language described through the questionnaire. One of the main goals of the Grammar Matrix project is rigor in grammatical hypothesis testing; the system makes more explicit the relationship between a grammar description,

structure subsumption and equality checks (Malouf *et al.* 2002, pages 114–122).

¹¹<https://github.com/delph-in/matrix#readme>

¹²<http://www.delph-in.net/matrix/customize/matrix.cgi>

¹³<https://www.python.org/>

or hypothesis, and the actual data from the language for which the description is intended.

The Grammar Matrix has been in active development for two decades; the original version, documented in Bender *et al.* 2002, was developed in late 2001. In that work, Bender *et al.* selected portions of the ERG (Flickinger 2000) which they believed would be useful cross-linguistically and put them together in the first version of the Grammar Matrix. The idea was that for a new grammar, this core distilled from the ERG could be included as a foundation, eliminating the need to write the grammar from scratch. Later, it was observed that some portions of the core lexical types and phrase structure rules could be customized to accommodate various typological profiles. This led to future iterations of the Grammar Matrix project which include the customization system (Bender and Flickinger 2005; Drellishak and Bender 2005; Drellishak 2009; Bender *et al.* 2010), which has been used as a starting point for a number of grammars (described in Section 6.3). The main purpose of the customization system is to automate the mapping between a language's typological profile and a particular set of lexical and phrasal HPSG types which serves this typological profile. As such, the Grammar Matrix is a research framework which aims to combine typological breadth with formal-syntactic depth (Bender *et al.* 2010). The relationship between the core and the customization system is such that it can be refined over time, as support for more and more syntactic phenomena is added for more and more typological profiles. For example, once newly considered data makes it obvious that something in the Grammar Matrix core retains any Indo-European (or specifically English) biases, the constraints representing those biases can be removed from the core and added instead to the customization system.¹⁴ Conversely, features and types can be added to the core when a general analysis is developed that is alternative to the one in the ERG.¹⁵

¹⁴For example, Trimble (2014, pages 60–67) moved all copula types and most adjective types to the customization system to account for languages without copulas and various adjectival phenomena – primarily switching and constrained argument agreement – that required significant reworking of the ERG's analysis.

¹⁵For example, Zamaraeva (2021a, pages 168–169) added to the core a feature named WH, which is a generalized version of a feature found in the Zhong

Linguistic hypothesis testing has been one of the main goals of the Grammar Matrix project since day one, but the range and the complexity of the hypotheses which can be tested depend directly on the syntactic and typological coverage of the system, which at first was modest. Bender *et al.* (2010) marked a significant milestone of the Grammar Matrix project, with support added for multiple phenomena and a wide range of typological profiles. Since then, another decade of contributions to the system have taken place (Section 3.1). After a brief overview of how the system can be used to generate a grammar (Section 3.2), we discuss the formal (Section 3.3) and methodological (Section 3.4) innovations that have expanded the capabilities of the system since 2010.

Grammar Matrix libraries added since 2010

3.1

Table 1 lists all the Grammar Matrix libraries that are currently available via the web questionnaire. Twelve new libraries have been added since 2010, increasing the system's scope and the complexity of interactions which can be studied. In particular, the libraries for complex clauses (Howell and Zamaraeva 2018; Zamaraeva *et al.* 2019b) enable the Grammar Matrix-derived grammars to parse recursive sentences, meaning much larger test suites can be used for development and evaluation (see Section 3.4.1). The library for information structure (Song 2014) brought in the important potential to associate information structural meanings with a range of syntactic phenomena used to mark information structure in the world's languages. This, in turn, opened up the possibility of modelling aspects of interrogatives in terms of information structure (Zamaraeva 2021a). The revamped morphotactics library (Goodman 2013) and lexicon and morphology extensions for adjectives and copulas (Trimble 2014) in combination with the new libraries for adnominal possession (Nielsen 2018; Nielsen and Bender 2018), evidentials (Haeger 2017), valence change (Curtis 2018a,b), and nominalization (Howell *et al.* 2018) allow us to model

grammar of Chinese (Fan 2018), to accommodate cross-linguistic patterns of question word fronting. For the discussion, see Zamaraeva 2021a, page 188, footnote 61.

Table 1: The Grammar Matrix libraries with selected typological sources

Library	Citation(s)	Selected typological sources
Coordination	Drellishak and Bender 2005	Payne 1985; Stassen 2000; Drellishak 2004
Polar Questions	Bender and Flickinger 2005	–
Person, Number, Gender	Drellishak 2009	Cysouw 2003; Siewierska 2004; Corbett 2000
Agreement	Drellishak 2009	Corbett 2006
Case; Direct-Inverse	Drellishak 2008, 2009	Givón 1994
Argument Optionality	Saleem and Bender 2010; Saleem 2010	Ackema <i>et al.</i> 2006; Dryer 2013a
Tense	Poulson 2011	Comrie 1985; Dahl 1985
Aspect	Poulson 2011	Comrie 1976; Bybee <i>et al.</i> 1994
Lexicon	Drellishak and Bender 2005; Trimble 2014	Dixon 2004
Morphotactics	O'Hara 2008; Goodman 2013	–
Sentential Negation	Crowgey 2012, 2013	Dahl 1979; Dryer 2013b
Information Structure	Song 2014	Féry and Krifka 2008; Büring 2009
Adjectives; Copulas	Trimble 2014	Dixon 2004; Stassen 1997, 2013
Evidentials	Haeger 2017	Aikhenvald 2004; Murray 2017
Nominalized Clauses	Howell <i>et al.</i> 2018	Noonan 2007
Clausal Modifiers	Howell and Zamaraeva 2018	Thompson <i>et al.</i> 1985
Valence Change	Curtis 2018b,a	Haspelmath and Müller-Bardey 2001
Adnominal Possession	Nielsen and Bender 2018; Nielsen 2018	Payne and Barshi 1999; Heine 1997
Clausal Complements	Zamaraeva <i>et al.</i> 2019b	Noonan 2007
Constituent Questions	Zamaraeva 2021a	Haspelmath <i>et al.</i> 2013; Hagege 2008

grammars which account for a fairly wide range of data from descriptive sources on languages with very different typological profiles, as discussed in Section 3.4.2 and illustrated in Figure 10.

How to use the libraries: an example

3.2

To illustrate how a user would use the Grammar Matrix customization system to model a particular phenomenon, we show an example of how one could fill out the questionnaire for constituent questions (Zamaraeva 2021a) in Paresi-Haliti [pab] (Arawakan). As we will describe in Section 3.4.1, the process of library development and evaluation involves using the customization system to generate grammars and then using those grammars to parse sentences from test suites. In this case, we describe how Zamaraeva (2021a) created a customized grammar for Paresi-Haliti based on the examples and description in Brandão 2014. Later in Section 4.1, we present a case study related to the evaluation of the constituent questions library on this language.

Based on the description in Brandão 2014, the subpage for constituent questions may look as in Figure 3. For example, Figure 3 reflects the hypotheses that Paresi-Haliti fronts one question phrase obligatorily and that the question words may be overtly marked with focus. The reader can see in Figure 3 that the Constituent Questions subpage of the Grammar Matrix web questionnaire references two other subpages, namely Information Structure and Lexicon. Given the specifications shown in Figure 3, at least one question word must be specified on the Lexicon subpage, as shown in Figure 4. Likewise, on the Information Structure subpage (Song 2014), an affix or a clitic which can attach to question words (as well as other words) must be added. In this case, a contrastive focus marker is specified as in Figure 5. In combination with other grammar specifications made through these and other subpages of the Grammar Matrix web questionnaire, it is possible to obtain an implemented grammar of Paresi-Haliti. We can then test its behaviour with respect to a test suite of grammatical and ungrammatical examples, as discussed further in Section 4.1.

The web questionnaire is capable of producing human-readable grammar specifications that can be saved and re-uploaded later or

Constituent (wh-) Questions [\[documentation\]](#)

Please indicate which strategy your language uses to form constituent (aka wh-) questions. You may leave this section blank, in which case your grammar will not include a wh-question-forming strategy.

If you fill out this page, you must also fill out the Question Words section on the Lexicon page.

Choices regarding the position of question phrases

Question phrases can appear at the left edge of the sentence regardless of the position the questioned constituent would appear in (*Who did you see? I know who you saw* etc.):

- Only one question phrase can be fronted
- All question phrases can be fronted
- Question phrases cannot be fronted (*stay in situ*)

There is **obligatory** fronting:

- of at least one question phrase
- of all question phrases
- fronting is optional

There is pied piping of:

- noun phrases (*Which book did you read?* is possible), and it is obligatory (**Which did you read book?* is impossible in your language);
- adpositional phrases (*To whom did you speak?* is possible), and it is obligatory (**Who did you speak to?* is impossible in your language).

Other choices

- Only one question phrase is allowed per sentence.
- Constituent questions are marked morphologically (specify lexical rules on the Morphology page).
- Question words may bear overt focus marking (specify on Information Structure page).
- Interrogative verbs (add appropriate entries on the Lexicon page).

If you specified Auxiliary-Subject inversion for polar questions, does it also happen in constituent questions?

- yes, in matrix clauses
- also in embedded clauses with constituent questions
- but not in questions about subjects.

Figure 3: The Grammar Matrix web questionnaire, Constituent Questions subpage, filled out for Paresi-Haliti [pab]

hand edited, and also acts as the intermediary to the customization system. As an example, the portion of the text specification corresponding to Figures 3 and 5 can be seen in Figure 6.

Based on specifications such as those shown in Figure 6, the customization system applies logic that outputs a customized grammar including the core types as well as language-specific types, rules and

Main page
Gen Info Word Order Number Person Gender Case Poss Dir-inv TAM Evidentials Features Neg Coord Y/N Qs Wh-Qs Info Str Arg Opt Nmz Embed Claus ?Clausal Mod Lexicon ?Morph Toolbox Import Test S TbG Options
Choices file <small>(right-click to download)</small> Save & stay Clear current subpage Create grammar: lga , zip

Noun Types | [visualize noun hierarchy](#) (experimental)

Some nouns in this language take adjectives as incorporated affixes:

▶ common (noun1)

▼ wh (noun2)

Noun type 2:

Type name:

Supertypes: ▼

This is a personal pronoun type

This is a question pronoun (like *who/what*)

Features:

Name: ▼ Value: ▼

For nouns of this type, a determiner is: obligatory optional impossible

Stems:

Spelling: Predicate:

Figure 4:
A portion of the
Lexicon subpage
of the Grammar
Matrix web
questionnaire,
filled out
for Paresi-Haliti
[pab]

Contrastive Focus

My language uses the same position to express contrastive focus as non-contrastive focus.

My language places contrastively focused constituents in a specific position. The position is

- clause-initial.
- clause-final.
- preverbal.
- postverbal.

▼ c-focus-marker2

This marker is

- an affix (You should create this affix on Morphology.)
- an adposition (You should create this adposition on Lexicon.)
- a modifier that appears ▼ ▼ Spelling:

Figure 5:
A portion of
the Information
Structure
subpage of the
Grammar Matrix
web
questionnaire,
filled out
for Paresi-Haliti
[pab]

lexical entries. For example, specifying an information structure clitic as in Figure 5 will result in the types shown in Figures 7–8 being added to the grammar. These types, in turn, rely on supertypes such as *no-rels-hcons-lex-item* and *one-icons-lex-item* in Figure 7 which are defined in the Grammar Matrix’s core grammar.

The grammar code in Figures 7–8 represents HPSG feature structures in a machine readable form, specifically in TDL, which is compatible with the DELPH-IN JRF. Assuming the grammar files contain

Figure 6:
Text specification output
by the Grammar Matrix questionnaire

```
section=wh-q
front-matrix=single
matrix-front-opt=single-oblig
pied-pip=on
oblig-pied-pip-noun=on
focus-marking=on
wh-q-inter-verbs=on

section=info-str
c-focus-pos=clause-initial
c-focus-marker2_type=modifier
c-focus-marker2_pos=after
c-focus-marker2_cat=nouns, verbs
c-focus-marker2_orth==ala
```

```
infostr-marking-mod-lex := no-rels-hcons-lex-item &
                        one-icons-lex-item &
[ SYNSEM [ NON-LOCAL non-local-none,
          LOCAL [ CONT.ICONS.LIST < #icons &
                [ IARG2 #target ] >,
                CAT [ VAL [ SUBJ < >,
                          COMPS < >,
                          SPR < >,
                          SPEC < > ],
                    HEAD adv &
                    [ MOD < [ LIGHT luk,
                            LOCAL [ CONT.HOOK [ INDEX #target,
                                                ICONS-KEY #icons ],
                            CAT [ MKG [ FC na-or--,
                                        TP na-or-- ],
                            WH.BOOL bool ] ] ] > ] ] ] ].
```

Figure 7: Grammar code output by the customization system

```
contrast-focus-marking-mod-lex := infostr-marking-mod-lex &
[ SYNSEM.LOCAL.CAT [ MKG fc,
                    HEAD.MOD < [ L-PERIPH luk,
                                LOCAL [ CAT.HEAD +nv,
                                        CONT.HOOK.ICONS-KEY contrast-focus ] ] >,
                    POSTHEAD + ] ].
```

Figure 8: Grammar code output by the customization system

enough code to constitute a functional grammar, they can be directly used with DELPH-IN parsers/generators such as the LKB or ACE (see Section 2.3).¹⁶ Thus, by adding a new Grammar Matrix library for a particular syntactic phenomenon, we enable the user to obtain a machine-readable HPSG grammar capable of parsing and generating sentences featuring this phenomenon by simply filling out a web questionnaire and without the need to write the grammar by hand (see Section 6.2–6.3).

Formal innovations

3.3

Once a user has created a grammar with the Grammar Matrix, they can continue developing it by adding or revising analyses to cover more phenomena. At this point, they must engage directly with the DELPH-IN JRF. As mentioned in Section 2.3, this formalism is deliberately restricted (Copestake 2002a). This means some constraints that are used in theoretical HPSG cannot be directly expressed using the DELPH-IN JRF. In particular, the formalism does not support relational constraints, where an operation on a specific feature value influences the value of another feature. Examples of such relations are applying logical-OR to feature values, list append (used for semantic composition and the handling of non-local features, among other things), and the shuffle operator (used in some analyses of variable word order).¹⁷

Emerson (2017, 2019, 2021, and forthcoming) has shown that, without changing the formalism, relational constraints can be mimicked using “computation types” and “wrapper types”. These computation types can be used to trigger operations such as logical-OR, and recursive type constraints can result in several lists being appended.¹⁸

¹⁶The Grammar Matrix customization system includes a validation component tasked with ensuring that the grammar specification is both complete and consistent enough to produce a functioning grammar. When the validation component detects that this is not the case, it signals this information to the user through warnings and errors on the questionnaire web pages.

¹⁷For details on non-local features in HPSG see Pollard and Sag 1994 and Ginzburg and Sag 2000; for DELPH-IN list implementation of list-valued features, see Copestake 2002a.

¹⁸See also Aguila-Multner and Crysmann 2018 for the discussion of application of append-lists in the context of feature resolution in coordination.

A full explanation of the workings of computation types lies beyond the scope of this paper. The main point we wish to make here is that they can ease grammar development. We illustrate this by comparing the classic DELPH-IN implementation of append operations, through difference lists (for an exposition, see Copestake 2002b, Section 4.3), to the new *append-list* type.

Examples (5)–(7) illustrate how difference lists (*diff-lists*) may be used to represent appending operations.¹⁹ Example (5) provides the basic type definition of a *diff-list* specifying that it consists of two lists. Example (6) shows the definition of a *diff-list* $\langle !a, b! \rangle$. Note that the value of LAST is identical to the REST of the list starting with *b*. As such, LAST corresponds to the end of the list.

$$\begin{array}{l}
 (5) \quad \left[\begin{array}{ll} \textit{diff-list} & \\ \text{LIST} & \textit{list} \\ \text{LAST} & \textit{list} \end{array} \right]
 \end{array}
 \qquad
 \begin{array}{l}
 (6) \quad \left[\begin{array}{ll} \textit{diff-list} & \\ \text{LIST} & \left[\begin{array}{ll} \textit{nonempty-list} & \\ \text{FIRST} & a \\ \text{REST} & \left[\begin{array}{ll} \textit{nonempty-list} & \\ \text{FIRST} & b \\ \text{REST} & \boxed{1} \textit{list} \end{array} \right] \end{array} \right] \\ \text{LAST} & \boxed{1} \end{array} \right]
 \end{array}$$

In difference lists, the end of one list can be identified with the beginning of another list. The example below illustrates how this can be used to create the *diff-list* on the left by appending the two lists following it.

$$(7) \quad \left[\begin{array}{ll} \textit{diff-list} & \\ \text{LIST} & \boxed{1} \\ \text{LAST} & \boxed{3} \end{array} \right] \quad \left[\begin{array}{ll} \textit{diff-list} & \\ \text{LIST} & \boxed{1} \\ \text{LAST} & \boxed{2} \end{array} \right] \quad \left[\begin{array}{ll} \textit{diff-list} & \\ \text{LIST} & \boxed{2} \\ \text{LAST} & \boxed{3} \end{array} \right]$$

Using *diff-lists* for such operations requires carefully keeping track of the components of the list. The “end” of a difference list is actually an underspecified list, and for that reason, difference list appends are

¹⁹These examples correspond to examples (14)–(16) in Zamaraeva and Emerson 2020, pages 162–163. More details and examples can also be found in Zamaraeva 2021a, pages 42–43.

notoriously easy to break when introducing new types to the grammar, leading to such problems as overgeneration, spurious ambiguity, and semantic representations with missing predications. Also, it is difficult to count elements on a difference list.²⁰ In practice, wrongly defined difference lists are a well-known source of errors in the grammar.²¹

The *append-list* type, illustrated in Example (8) has a feature APPEND, which allows for simple and elegant syntax,²² thereby making grammars easier to develop and maintain. Example (9) illustrates what appending two lists looks like when using *append-list*. For a more detailed exposition of how the *append-list* type works, see Zamaraeva and Emerson 2020.²³

$$(8) \left[\begin{array}{l} \textit{append-list} \\ \text{LIST} \quad \boxed{0} \textit{ list} \\ \text{APPEND} \quad \left[\begin{array}{l} \textit{list-of-list-wrappers-with-append} \\ \text{RESULT} \quad \boxed{0} \end{array} \right] \end{array} \right]$$

$$(9) \left[\begin{array}{l} \textit{append-list} \\ \text{APPEND} \quad \langle \boxed{1}, \boxed{2} \rangle \end{array} \right] \quad \boxed{1} \left[\begin{array}{l} \textit{append-list} \\ \text{LIST} \quad \langle a, b \rangle \end{array} \right] \quad \boxed{2} \left[\begin{array}{l} \textit{append-list} \\ \text{LIST} \quad \langle c \rangle \end{array} \right]$$

Implementing the *append-list* type in the Grammar Matrix allowed for faster development of analyses which relied heavily on manipulating non-local lists, such as the ones developed for the constituent questions library (Zamaraeva 2021a).

Methodological innovations

3.4

This section describes several important methodological principles characteristic of the Grammar Matrix development (Section 3.4.1) and what innovations took place in the recent years with respect to

²⁰ See Zamaraeva and Emerson 2020 for details.

²¹ We base this claim on our experience as grammar engineers and our experience of teaching grammar engineering to others.

²² In the sense of programming language syntax, not a branch of linguistics.

²³ Examples (8) and (9) correspond to examples (17) and (18) in Zamaraeva and Emerson 2020, page 164.

those principles. The first innovation is that we extended the practice of testing analyses for generalizability against held-out languages to using held-out language families (Section 3.4.2). The second important development is the evolution of “regression testing” (Section 3.4.3), which ensures an explicit area of applicability for the large and complex system of grammatical hypotheses. The third is CLIMB (Section 3.4.4), which is a methodology that allows one to track, after a starter grammar was created (e.g. with the Grammar Matrix), how one analysis influences subsequent decisions, and what the alternatives could have been. Finally, the last innovation is the “Spring cleaning” algorithm (Section 3.4.5), which allows the identification of portions of the grammar that are in fact unused.²⁴

3.4.1

Data-driven Grammar Matrix development

The overall methodology for Grammar Matrix development was first summarized in Bender *et al.* 2010. The development is driven by typology and data: it starts from aggregating typological descriptions and exemplar sentences for the phenomenon for which support is being added; data is used as a guide throughout the development of analyses; finally evaluation is performed using previously unseen (“held-out”) languages, usually from held-out language families.

Grammar Matrix libraries are developed in a data-driven manner, using a set of illustrative “development” languages and corresponding test sets. When adding support for a syntactic phenomenon, a Grammar Matrix developer typically first compiles test suites from several languages consisting of exemplar grammatical and ungrammatical sentences illustrating the syntactic phenomenon being modelled.²⁵

²⁴ Both CLIMB and Spring cleaning methods can be applied to any DELPH-IN grammar but both were developed in the context of Grammar Matrix development (Fokkens 2014).

²⁵ In the context of Grammar Matrix library development and in cases when exemplar sentences from a descriptive grammar contain phenomena which are not being modelled and are not already present in the Grammar Matrix, the developer may have to simplify/modify the sentences, e.g. remove a greeting, discourse particle, or even a relative clause (as relative clauses are not currently supported by the Grammar Matrix customization system). In such cases, it is ideal to get judgments on the resulting modified sentences from an

Many of these test suites come from natural languages which the developer encountered in their typological literature review. Others are constructed using artificial languages, or “pseudo-languages”, to ensure the testing of typological combinations not illustrated with specific examples in the typological literature.

To illustrate the data-driven development process in more detail, we will use an example that includes both a test suite from an actual language and a test suite made of artificial data representing a hypothesized language type (pseudo-language). It helps to (i) connect the methodology to software engineering practices and (ii) give the reader additional background for the case study presented later in Section 4.3. The process described below is exactly the same for real language data.

The process of library development starts from reading a descriptive source and compiling a test suite of examples, grammatical and ungrammatical, illustrative of the phenomenon for which the library is being added. For example, if the descriptive grammar states that the language has separate morphological paradigms for verbs in declarative (10a) and interrogative sentences (10b–c), the test suite will include examples of each.

- (10) a. oža-va iche-žee-v
track-ACC see-FUT-1SG
‘I will see the tracks.’ [neg] (Khasanova and Pevnov 2002; cited by Hölzl 2018, page 295)
- b. ii-ǰə-m = i?
enter-FUT.Q-1SG.Q = Q
‘Shall I come in?’ [neg] (Khasanova and Pevnov 2002; cited by Hölzl 2018, page 295)
- c. eeva iche-ža-m?
what see-FUT.Q-1SG.Q
‘What will I see?’ [neg] (Khasanova and Pevnov 2002; cited by Hölzl 2018, page 295)

L1 speaker; unfortunately that is not always possible to do. The methodology assumes that any modifications to the original sentences are carefully documented.

Examples (10a–c) come from Negidal [neg] (Tungusik). It uses the same paradigm for polar and constituent questions (e.g. *-m* in (10b–c)). This language represents one typological profile; another includes languages which use three distinct paradigms: one for declaratives, another for polar questions, and yet another for constituent questions. One such language is Makah [myh] (Wakashan). During the development of the constituent questions library, this typological profile was included as a formal experiment; Zamaraeva (2021a) was not aware that Makah has this feature but had hypothesized that such languages may exist and constructed illustrative artificial data (11a–i).

- (11) a. noun tverb-PQ noun?
b. who iverb-WHQ?
c. who tverb-WHQ what?
d. who tverb-WHQ noun?
e. noun tverb-WHQ what?
f. *noun tverb-WHQ noun?
g. *who tverb-PQ what?
h. *who tverb-PQ noun?
i. *noun tverb-PQ what?

At this point, the library developer has the test suite like the one above (11a–i), illustrating the fact that the language uses three separate morphological paradigms, and the appropriately filled out questionnaire which, assuming the questionnaire-customization system interface was already implemented, generates textual grammar specification like in Figure 9.²⁶

Given a (complete) specification containing the sections relevant to the separate morphological marking in polar and constituent questions, the goal is for the customization system to output a grammar which behaves correctly with respect to the data in (11a–i), namely one that maps the grammatical strings (11a–e) to their correct linguistic representations and rejects the ungrammatical strings (11f–i).

²⁶Only the morphology section of the specification is shown.


```
section=morphology
verb-pc1_name=mood
verb-pc1_obligatory=on
verb-pc1_order=suffix
verb-pc1_inputs=verb
  verb-pc1_lrt1_name=polar
    verb-pc1_lrt1_feat1_name=question
    verb-pc1_lrt1_feat1_value=polar
    verb-pc1_lrt1_feat1_head=verb
    verb-pc1_lrt1_lri1_inflecting=yes
    verb-pc1_lrt1_lri1_orth=-PQ
    verb-pc1_lrt2_feat1_name=question
    verb-pc1_lrt2_feat1_value=wh
    verb-pc1_lrt2_feat1_head=verb
    verb-pc1_lrt2_lri1_inflecting=yes
    verb-pc1_lrt2_lri1_orth=-WHQ
  verb-pc1_lrt3_name=ind
    verb-pc1_lrt3_feat1_name=question
    verb-pc1_lrt3_feat1_value=no
    verb-pc1_lrt3_feat1_head=verb
    verb-pc1_lrt3_lri1_inflecting=no
```

Figure 9:
Lexical rules specification output
by the Grammar Matrix questionnaire

At this point, the Grammar Matrix library developer has a clear map of what the finished library should cover, in terms of accepting and rejecting strings. This is the starting point of so-called test-driven development in software engineering (Beck 2003)²⁷ where first the tests are written and then the code is added to the program until all tests pass.

Building the library entails (i) deciding on target semantic representations, (ii) developing the implemented HPSG analyses that will produce those representations, (iii) developing the customization logic that will output the correct grammar components given a specification, and (iv) writing the questionnaire portions to elicit the specification. Grammar Matrix developers typically proceed by creating starter grammars through the customization system with enough other specifications to cover all other phenomena in the test suites and then using those grammars as test beds to work out analyses of specific variants of the phenomenon targeted by the library under development. Once those analyses are developed, and the semantic representations

²⁷ Beck (2003) is often credited for “rediscovering” the concept of test-driven development, as he popularized the term and the practice. The concept probably long predates his book, though we could find no other canonical citation.

they produce hand-verified, they can be generalized and added to the customization logic.

In the case of Grammar Matrix data-driven development, first we have the data (the test suite), and then we develop an analysis and add logic to the customization system until the system starts outputting a grammar which behaves correctly with respect to this data. It is an iterative process. For example, given only the specification in Figure 9 but no code designed to add something to the grammar based on this specification, the output grammar will not include the lexical rules for morphological marking and therefore will not cover sentences like (11a-i); its coverage over the test suite will be 0%.²⁸ The developer can then add the programmatic logic to the system such that, upon encountering a specification like the one in Figure 9, appropriate lexical rules, such as the one in (12), are added to the grammar. This is a type which is part of the analysis of distinct morphological marking in polar and constituent questions discussed in Section 4.3.

$$(12) \left[\begin{array}{l} \textit{polar-lex-rule} \\ \text{SYNSEM|LOCAL|CAT|VAL} \left[\begin{array}{l} \text{SUBJ} \left\langle \left[\text{NON-LOCAL|QUE|LIST} \langle \rangle \right] \right\rangle \\ \text{COMPS} \textit{non-wh-list} \end{array} \right] \end{array} \right]$$

Even in cases where there is already literature or other grammars containing similar types, developing analyses with the level of rigor and detail that the Grammar Matrix requires is often non-trivial. When building on the theoretical literature, the analyses may be sketched in too general a way, and may assume operations not available in the DELPH-IN JRF. When building on analyses already developed in other DELPH-IN grammars, the types are likely to be overly specific to another language. In many cases, especially characteristic of non-Indo-European typology, the analysis will simply not be found anywhere and will need to be developed from scratch. It is therefore expected that the process of developing a Grammar Matrix library involves many debugging cycles. The developer can then load the imperfect grammar into the software such as the LKB and perform interactive

²⁸The coverage can be computed automatically using tools such as [incr tsdb()] (Oepen and Flickinger 1998).

debugging. The LKB will show any errors that occur when loading the grammar, unification failures that prevent a sentence from parsing, every possible parse tree for each sentence allowing the developer to identify the cause of any spurious parses, and the semantic representation for each parse tree allowing the developer to inspect the correctness of each parse. The developer can then refine the analysis to ensure that grammatical sentences in the test suite are parsed correctly and ungrammatical sentences are not parsed.

To summarize, data-driven development first lays out the grammatical territory to cover in a set of test suites, and then extends the grammar customization system to cover that territory. As the analysis is developed and refined, the grammar engineer moves back and forth between increasing the system's ability to handle grammatical sentences (coverage) and working to minimize both spurious ambiguity (extraneous extra parses of grammatical sentences) and overgeneration (parses of ungrammatical sentences).²⁹ After the development is finished, as far as can be tested with the initial test suite, the library is evaluated as described in Section 3.4.2.

Evaluation with languages from held-out language families

3.4.2

After a library's development is concluded, the developer performs evaluation on held-out languages from held-out language families. This means each evaluation language must come from a different language family than other evaluation languages, and furthermore, from a family not represented in the languages used in the library's development (Zamaraeva 2021a, page 103).³⁰ The goal is to assess how well the Grammar Matrix analyses generalize with respect to a randomly selected set of languages, specifically languages which may have different properties compared to the languages that were driving the development. This is meant to approximate what will happen when users approach the Grammar Matrix to model additional languages.

²⁹In practice, lack of coverage, some spurious ambiguity, and some overgeneration may be unavoidable due to development time constraints, in which case the specific cases are documented and left for future work.

³⁰While at this stage we avoid including language families that we worked with directly in library development, we do not necessarily exclude families just because they were included in or informed the typological surveys we build on.

The process of evaluation is very similar to the one described above in Section 3.4.1, starting with descriptive sources and the compilation of a test suite, then filling out the questionnaire, obtaining a grammar from the customization system, and deploying it on the test suite. However, at the evaluation stage no modifications are made to the system and the coverage, the overgeneration, and the spurious ambiguity are expected to not be perfect and are reported as measures of the quality of the newly added library.³¹

The practice of using held-out languages (but not necessarily coming from unseen language families) goes back to at least Saleem 2010. Starting with Haeger 2017, only unseen language families are used as part of the methodology. Ultimately, all the test suites (both development and evaluation) are preserved along with the corresponding language specifications and expected “gold” semantic representations for each sentence in the test suite. These sets of files constitute the content of the regression testing system to ensure continued functioning of existing analyses (Section 3.4.3).

3.4.3 Automatic testing of all existing analyses

A crucial part of Grammar Matrix development methodology is the automatic testing of all analyses currently implemented in the system with respect to stored test suites containing data from different languages, both actual languages and abstract pseudo-languages (regression testing; Bender *et al.* 2007). Regression testing is what ensures that the combination of analyses that constitute the Grammar Matrix have at least a known, explicit area of applicability. Most importantly, regression testing makes it possible to see precisely whether and how any new analysis affects the previous system of analyses with respect to the previously established area of applicability. In other words, regression testing of the Grammar Matrix allows us to extend the computationally assisted method of fragments (Montague 1974; Partee 1979; Gazdar *et al.* 1985) to a cross-linguistic arena.

The term regression testing comes from software engineering where it describes tests that check functional behaviour of a system over time; i.e. each modification to the system can be tested on

³¹ Any issues discovered during the evaluation stage are documented such that they can be addressed later.

previously used inputs for which the expected (“gold”) output was recorded. If the system behaves differently after a modification, this is a “regression”, assuming there is no mistake in the stored gold outputs. Regressions need to be addressed, either by fixing the system such that the behaviour is back to what was expected or by updating the expected “gold”, if the new output is in fact correct or closer to correct. The practice of regression testing for monolingual grammar development is elaborated in Oepen and Flickinger 1998, Oepen 2002, and Oepen *et al.* 2002, among others.

In the context of the Grammar Matrix, regression tests are sets of grammar specifications (valid inputs to the customization system), input language strings from languages corresponding to the descriptions (with grammatical and ungrammatical items), and finally the corresponding “gold” semantic structures (one or more correct structures for each grammatical sentence and no structures for ungrammatical ones). For example, the test suite discussed above in (11a–i) along with the corresponding grammar specification and the set of correct MRS representations for all the grammatical sentences in the test suite will constitute a regression test after the development of this portion of the system is completed. Running a Grammar Matrix regression test involves invoking a system which automatically feeds a grammar specification to the customization system to obtain a grammar fragment, uses the grammar to process the input language strings, and compares the output to the stored “gold” results. If there is any difference between the expected output and the actual output, the test is flagged and the developer can investigate what causes the difference.

The regression testing system was in place by 2010 with 130 test suites; in the past decade it has had significant extensions, comprising 527 test suites at time of publication. Re-engineering of the system using modern software engineering methods³² resulted in a much faster, more robust system. This is crucial in the context of the greatly increased number of tests, as otherwise the time to run all the tests

³²The regression testing system was re-engineered twice in the decade, once by Sanghoun Song to use the faster parser ACE (Crysmann and Packard 2012) instead of the LKB parser (Copestake 2002b), and once by Michael W. Goodman to use the PyDelphin libraries and multiprocessing, both times with input from other DELPH-IN contributors.

often (such as for each small modification) would have become prohibitive. Another key change in Grammar Matrix regression testing practice, supported by the more robust regression testing facilities, is the move to include not only artificial (“pseudo-language”) tests in the regression testing suite, but also tests depicting natural languages used in library development (“illustrative languages”) and testing (“held-out languages”).³³

The current regression testing system counts 527 test suites, of which 75 are from 60 unique natural languages representing 40 language families. Figure 10 shows these 60 languages on the world map. Table 2 summarizes how many languages represent each family.³⁴

The Grammar Matrix regression testing system represents a crucial methodological principle of the project, namely that the analyses can be rigorously tested together, thus allowing Grammar Matrix developers to state confidently that the set of HPSG analyses implemented in the customization system definitely accounts at least for the data stored in the system. Over the years, the system has grown to cover 40 language families from all over the globe. The data compiled from descriptive resources for 60 languages that the Matrix regression testing system currently contains can be reused for research purposes by anyone who is interested in typologically diverse data on any of the syntactic phenomena represented in the Grammar Matrix.³⁵

³³ Thus, while all of the libraries were tested on natural languages as well as pseudo-languages, there are some natural languages described in the Grammar Matrix literature which never made their way into the regression test suite set.

³⁴ Language family and ISO 639-3 codes are given as in WALS (Dryer and Haspelmath 2013) or, if not found there, as in Glottolog (Hammarström *et al.* 2021), except in cases where we learned that the name listed in those resources contained a slur. The second column in the table is the number of unique languages represented per family, e.g. there are two unique Afro-Asiatic, Niger-Congo, etc., languages represented in the system. The third column shows the total number of test profiles. This last number includes any repetitions, e.g. Japanese is represented in the system by 3 test profiles which may contain different sentences and target different syntactic phenomena. That is why the number in column 3 is not necessarily obtained by multiplying the number in column 2 by the number of items in the corresponding cell in column 1.

³⁵ The median size of the test suites is 17 sentences. The largest test suite is for Umatilla Sahaptin [uma] (Penutian) (Drellishak 2009) and it contains over 6000

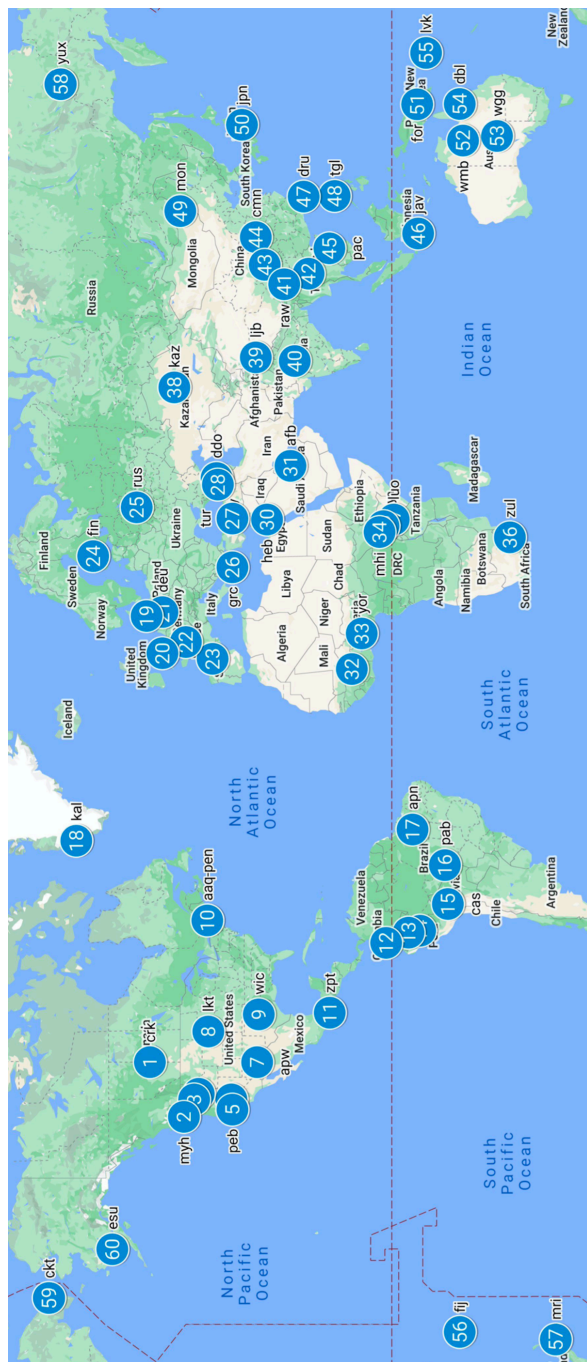


Figure 10: Languages in the Grammar Matrix regression testing system. Map: Google Maps; map data: SIO, NOAA, U.S. Navy, NGA, GEBCO Image Landsat/Copernicus. Locations/spellings as in WALS (Dryer and Haspelmath 2013) or Glottolog (Hammarström *et al.* 2021). Starred* numbers are not fully visible. 1. Cree [crk]; 2. Makah [myh]; 3. Yakima Sahaptin [yak]; 4*. Umatila Sahaptin [uma]; 5. Eastern Pomo [peb]; 6*. Washo [was]; 7. Western Apache [apw]; 8. Lakota [lkt]; 9. Wichita [wic]; 10. Penobscot [pen]; 11. Zapotec [zpt]; 12. Awa Pit [kwi]; 13. Uranina [ura]; 14*. Shipibo-Konibo [shp]; 15. Moseeten [cas]; 16. Paresi [pab]; 17. Apinaje [apn]; 18. West Greenlandic [kal]; 19. North Frisian [fr]; 20. English [eng]; 21. German [deu]; 22. French [fra]; 23. Basque [eus]; 24. Finnish [fin]; 25. Russian [rus]; 26. Ancient Greek [grc]; 27. Turkish [tur]; 28. Georgian [kat]; 29*. Tsez [tse]; 30. Hebrew [heb]; 31. Gulf Arabic [afb]; 32. Jalkunan [bxl]; 33. Yoruba [yor]; 34. Madi [mhi]; 35*. Lango [laj]; 36. Zulu [zul]; 37*. Luo [luo]; 38. Kazakh [kaz]; 39. Ladakhi [ljb]; 40. Hindi [hin]; 41. Rawang [raw]; 42. Blang [blr]; 43. Quiang [cng]; 44. Mandarin [cmn]; 45. Pacoh [pac]; 46. Javanese [jav]; 47. Rukai [dru]; 48. Tagalog [tgl]; 49. Mongolian [mon]; 50. Japanese [jpn]; 51. Fore [for]; 52. Wambaya [wmb]; 53. Wangkangurru [wgg]; 54. Dyiribal [dbl]; 55. Lavukaleve [lvk]; 56. Fijian [fij]; 57. Maori [mri]; 58. Yukaghir [yux]; 59. Chukchi [ckt]; 60. Yup'ik [esu]

Table 2:
Language families
represented
in the regression
testing system

Language family (ISO 639-3 codes)	N languages	N test suites
Indo-European (deu, eng, fra, frr, grc, hin, rus)	7	13
Austronesian (dru, fij, jav, mri, tgl)	5	6
Sino-Tibetan (cmn, cng, lbj, raw)	4	4
Afro-Asiatic (afb, heb), Algic (aaq-pen, crk), Altaic (kaz, tur), Austro-Asiatic (pac, blr), Eastern Sudanic (luo, laj), Inuit-Yupik-Unangan (esu, kal), Niger-Congo (yor, zul), Pama-Nyungan (dbl, wgg), Penutian (uma, yak)	2	19
Arawakan (pab), Barbacoan (kwi), Basque (eus), Caddoan (wic), Central Sudanic (mhi), Hokan (peb), Japanese (jpn), Kartvelian (kat), Macro-Ge (apn), Mande (bxl), Mirndi (amb), Mongolic-Khitian (mon), Mosestenan (cas), Na-Dene (apw), Nakh-Daghestanian (ddo), Otomanguean (zpt), Panoan (shp), Siouan (lkt), Solomons East Papuan (lvk), Trans-New Guinea (for), Uralic (fin), Urarina (ura), Wakashan (myh), Washo (was), Yukaghir (yux)	1	33

3.4.4

CLIMB

One motivation for implementing precision grammars is that natural languages are complex, consisting of many phenomena that interact. The analyses for these phenomena also interact, which makes it practically impossible to verify whether a newly proposed analysis interacts correctly with existing analyses without the aid of a computer (see also Section 4). Implementing grammars provides the means to test this through systematically adding test sets that represent covered

sentences; it was partially computer generated based on the examples found in a descriptive grammar. The largest test suite fully vetted by an L1 speaker is for Russian [rus] (Indo-European) and it contains 273 sentences (Zamaraeva 2021a).

phenomena and applying regression tests as described in Section 3.4.3. Regression testing is also used in the development of individual grammars, as grammar engineers create and continuously update test data, and test the grammar on the full set of test data after each change.

In this way, grammar engineering can contribute to more systematic syntactic research. A challenge that remains, however, is that regression tests only allow grammar engineers to look back. There are often multiple ways in which a phenomenon can be analyzed and the decision for a specific analysis can only be tested on those phenomena that have already been analyzed and not on those that are not covered yet.³⁶ It is inevitable that decisions are sometimes made based on inconclusive evidence. The order in which phenomena are considered can thus have a major impact on the resulting grammar (Fokkens 2011; Fokkens 2014, page 69).

The CLIMB³⁷ method (Fokkens *et al.* 2012) aims to address this challenge by extending the idea of grammar customization from providing a mere kick-start to a general methodology of grammar development. The basic idea behind CLIMB is that, in case of inconclusive evidence, alternative analyses are implemented in a *metagrammar* which can generate grammars with either of the solutions. The grammar developer can maintain the alternative analyses and keep on testing their interactions with analyses for other phenomena until sufficient evidence is found. CLIMB uses the Grammar Matrix customization software to continue the development of individual grammars *after* the kick-start from the general customization system has taken place. It is thus *per se* not a method for developing new libraries for the Grammar Matrix customization system itself. It can however also be used when developing new customization libraries. In fact, the method was first developed to compare alternative analyses for V2-word order across languages (Fokkens 2011). In practical terms, CLIMB consists of programmatic scripts which work with the Grammar Matrix files.

There are currently three versions of CLIMB for DELPH-IN grammars described in detail in Fokkens 2014, Chapters 3–4. In its origi-

³⁶ Naturally, linguists are not fully unaware of phenomena that are not covered yet and can take them into account to some extent.

³⁷ Comparative Libraries of Implementations with Matrix Basis (CLIMB).

nal form, CLIMB continues to use the Grammar Matrix customization system for grammar engineering. The grammar developer includes all analyses in the original customization system (possibly cleared of components that are not relevant for the language in question). The development cycle consists of (i) generating one or multiple versions of the grammar using the customization system; (ii) adding and testing a new analysis in one or more grammars; (iii) adding these analyses to the customization system; (iv) generating and testing grammars with all alternative analyses for previously covered phenomena; and (v) adapting the analysis for proper interaction with alternative analyses if applicable and testing again. The advantage of using this version of CLIMB is that it offers the full flexibility of the customization system.

A disadvantage is that it involves moving back and forth between declarative coding for grammar engineering and procedural coding in the customization system. Moreover, in practice, the full flexibility of the customization system is not likely to be exploited. Notably the morphotactics library offers countless options which are not likely to be considered as alternative options for which more evidence is needed. A second version, called declarative CLIMB, offers an alternative way of using CLIMB without writing procedural code. The grammar engineer can define alternative analyses and the accompanying modifications that are needed to make them work with the rest of the grammar with an indication of the analysis they belong to. The Grammar Matrix customization code is used behind the scenes to create well-formed grammars from a set of selected analyses.

The third version of CLIMB is an adaptation of declarative CLIMB meant to support research on large-scale grammars developed in the traditional way. Declarative CLIMB consists of a shared core and collection of (alternative) analyses from which grammars can be generated. In this third version, CLIMB provides a set of changes that can be applied to a working grammar. The grammar developer can define additions, replacements and components to be removed to adjust the grammar. The customization code is used to generate an adapted grammar based on the original grammar and the changes. The code can also generate the set of changes needed to replace the new alternative analysis with the original analysis. The developer can thus propose an alternative analysis, add new analyses to the grammar with

this alternative and generate a version of the grammar with these new analyses and the original analysis.

In the scenario above, CLIMB is used to test alternative hypotheses during (part of) the trajectory of grammar development for a specific language. The method can also be used for flexible parallel grammar development. For a specific language, CLIMB could support versions of the grammar that exhibit (slightly) different behaviour. This could be versions that are adapted for a specific domain (e.g. one that captures structures typically used on social media, one more aimed at newspaper text), that aim to be more robust rather than precise, or that are designed for a specific application. Fokkens (2014, Section 6.4) illustrates for instance how CLIMB can be used to include alternative rules to spot specific errors of second language learners, as also seen in e.g. Flickinger and Yu 2013, Morgado da Costa *et al.* 2016, and Morgado da Costa *et al.* 2020.

Another natural way of using CLIMB is for multilingual grammar development. In this case, grammar developers truly continue in the spirit of the Grammar Matrix customization system. When a new analysis is developed for one language, code generation can be integrated in grammars of other typologically related languages. The idea of aiming for full typological coverage is abandoned, which allows for more depth. In addition to parallel grammar development, this can result in a significantly larger jump start for a new grammar of an additional related language. For instance, Fokkens (2014, Section 6.2.5) illustrates the increase in coverage of phenomena captured when developing a grammar for Northern Frisian from gCLIMB (a metagrammar for German that also contains variations for Dutch) compared to developing it from the Grammar Matrix customization system alone. In addition to gCLIMB, CLIMB has been used to create a prototype for a Slavic metagrammar that can generate a basic grammar for Russian (Fokkens and Avgustinova 2013).

Spring cleaning

3.4.5

One of the questions that arises when using a resource that supports grammar development for typologically diverse languages, such as the Grammar Matrix, is how much of the generic core and provided jump-start implementations end up being used. Though it is straightforward

to automatically check which type definitions have been modified, it is less trivial to find out which type definitions are actually active in a grammar and which are never invoked. When grammar developers implement analyses that take a different approach than the grammar core, for instance, they do not necessarily remove the corresponding components from the core. Likewise, obsolete type definitions are not necessarily removed when analyses are adapted or replaced. The spring cleaning algorithm (Fokkens *et al.* 2011) can be used to identify which components of the grammar actually influence the grammar's behaviour. The algorithm was developed with the specific purpose of identifying components of the Grammar Matrix that are an active part of (larger) grammars and relies on the code from the customization system to process DELPH-IN JRF.

A DELPH-IN grammar defines a hierarchy of typed feature structures. Each type inherits all constraints from its supertypes. A grammar furthermore defines instances: lexical items or rules defined through the type hierarchy. The parser and the generator start with instances: the parser forms syntactico-semantic representations of words based on its lexicon and lexical rules and then combines them using grammar rules. Conversely, the generator generates surface strings using the lexicon and lexical rules and combining them using grammar rules. This means any type that defines (part of) an instance impacts the grammar. These types are referred to as *instantiated types*. Combining components is done through unification. Types that influence whether instantiated types can unify therefore also impact the grammar. Conversely, types which are neither instantiated nor influence unification of instantiated types have no effect on the grammar.

The spring cleaning algorithm starts from the instance definitions. It then goes through the grammar and tags all types that are a supertype of an instance and marks them as instantiated types. It then extracts the feature values from all instantiated type definitions and marks the type definitions of these values and their supertypes as relevant types as well. In the next step, the algorithm checks whether the remaining types (that are not instantiated types nor types that are part of the definition of an instantiated type) enable unification of any relevant type. Any type that influences unification of relevant types is also marked as relevant. Remaining types are flagged as redundant. The algorithm was applied to four Grammar Matrix grammars repre-

senting three languages (two grammars with alternative word order analyses for Wambaya, one for Mandarin Chinese, and one for Bulgarian). The outcome showed that even relatively small grammars contained noise and that identifying superfluous types can help identify errors in the grammar (Fokkens *et al.* 2011). Occasional spring cleanings of grammars are therefore recommended.

Summary

3.5

This section summarized how the support for syntactic phenomena in the Grammar Matrix has grown since 2010 and covered the most important formal and methodological innovations adopted in this context. We listed all current Grammar Matrix libraries (and will later illustrate interactions among some of these). We presented in detail the testing system that currently covers 60 languages from 40 language families and allows for automatic testing of any modification in any of the analyses with respect to data from this wide typological range. In addition, this section summarized some formal innovations particularly relevant to how the system implements non-local dependencies (further discussed in Section 4.3) and described algorithms which allow the grammar engineer to track how analyses in a DELPHIN grammar influence each other and how to determine whether some parts of the grammar remain unused – which are important for future improvements of the Grammar Matrix project and the grammars it gives rise to.

MAKING TENSIONS BETWEEN ANALYSES EXPLICIT

4

Grammar engineering allows a grammarian to identify unexpected interactions between analyses which might otherwise be overlooked due to the overall complexity of the grammar. Furthermore, in the case of the Grammar Matrix, this is done with respect to the entire cross-linguistic system that the framework provides. In this section, we present several examples of tensions between syntactic analyses which

were made explicit in the context of the Grammar Matrix’s development and testing. We start in Section 4.1 with a case study illustrating the range of problems in the analyses that we are now able to discover and document. The important thing to note here is that documenting such a range of issues only became possible with the recent additions to the Grammar Matrix libraries (Section 3.1), because these issues all have to do with *interactions* among analyses of phenomena such as clausal complementation and modification, nominalization, adnominal possession, information structure, constituent questions, and long-distance dependencies. We then present a particularly intricate formal issue having to do with non-local dependencies and coordination which would probably be impossible to detect without a computational framework but has bearing on very common, seemingly simple sentences (Section 4.2). Finally, in Section 4.3, we discuss tensions that inform decisions of what belongs in the core grammar vs. the libraries, again revealed in the process of evaluating a new library on a held-out language.

4.1 *Two word order hypotheses in Paresi-Haliti: A case study*

In the context of her work on the constituent questions library for the Grammar Matrix, Zamaraeva (2021a, Section 8.5.9) considers two alternative analyses for basic word order in Paresi-Haliti [pab] (Arawakan) based on a descriptive grammar of the language (Brandão 2014). Brandão 2014 features a number of long, complex examples, which is good material for testing the interaction between Grammar Matrix libraries. However, according to Zamaraeva (2021a, page 342), many examples are not yet fully glossed and some phenomena are not yet fully described or understood (as is normal for an underdocumented language). In particular, the word order is said to be mostly V-final (the most common order being SOV), yet personal pronouns can occur after the verb, and indeed if they are taken into account, then, according to another source, da Silva 2013 (cited by Brandão (2014, page 318)), the most frequent word orders in the language are SVO and OSV. All orders in fact occur with some frequency, according to Brandão (2014, page 319). The exact nature of the interaction of information structure with word order is not fully worked out, though

there is a section on focus and topic and many examples are glossed for information structure marking (13).

- (13) aliyakere = ta = la hatyohare
how = EMPH = FOC this
'How is this?' (Brandão 2014, page 335)

Due to multiple possible hypotheses for what the basic word order is, this part of the grammatical description of Paresi-Haliti is thus an excellent candidate for computationally assisted hypothesis testing, for example using the Grammar Matrix.

To that end, Zamaraeva (2021a, Section 8.5.9) presents two grammars of Paresi-Haliti, both produced automatically by the Grammar Matrix. The grammars represent two sets of hypotheses, one associated with SOV word order (with some of the other orders accounted for by information structure) and another with free word order (reflecting the fact that all orders are possible). The analyses are evaluated with respect to a Paresi-Haliti test suite containing grammatical and ungrammatical sentences. The test suite consists of 67 items, 64 of them grammatical. Out of those 64, 45 are directly from Brandão 2014 while 19 have modifications or were constructed by Zamaraeva (2021a) based on the information from Brandão 2014. Of the 3 ungrammatical examples, 2 are constructed by Zamaraeva (2021a) and 1 is from Brandão 2014. Table 3 presents the evaluation numbers obtained by running the LKB parser (Copestake 2002b) over the test suite.³⁸

While it can be relatively easy to achieve close-to-perfect numbers on a (small) test suite that is driving grammar development, the Paresi-Haliti results from Zamaraeva 2021a, Section 8.5.9 (Table 3) represent the evaluation of the Grammar Matrix system on a held-out language family after the development process was frozen, and so relatively low

³⁸Raw coverage refers to grammatical sentences for which a grammar can produce any reading at all; validated coverage is for sentences which get a parse with correct semantics; overgeneration is for ungrammatical sentences which nonetheless are accepted by the grammar; and finally ambiguity is the average number of readings per sentence. High overgeneration and ambiguity indicate problems with the grammar, as does low coverage; high raw coverage is not necessarily good unless validated coverage is also high.

Table 3:
Two word order
hypotheses
for Paresi-Haliti

Hypothesis	Raw coverage (%)	Validated coverage (%)	Overgeneration (%)	Ambiguity
SOV	40/64 (62.5)	25/64 (39.0)	2/3 (66.7)	43.95
free	53/64 (82.8)	36/64 (56.0)	2/3 (66.7)	36.17

validated coverage, high overgeneration, and high ambiguity can all be expected. However, after evaluation results are reported, we can still have a good look into which exact problems led to the missing coverage as well as to overgeneration and any spurious ambiguity. For this we use automated DELPH-IN tools, particularly [incr tsdb()] (Oepen and Flickinger 1998) and treebanking tools.

A close examination of the two grammars with respect to the Paresi-Haliti test suite revealed the following issues in the Grammar Matrix system. First of all, we found out that the information structure library was overconstraining SOV grammars such that complements could never be extracted out of VP.³⁹ Removing that constraint did not lead to any regressions in any of the 527 regression tests, so that problem can easily be fixed at the level of the entire Grammar Matrix system. Note that it took a complex test suite featuring both information structure marking and constituent questions in combination with the SOV word order to discover this problem; without testing the interaction, the problem went unnoticed for years.

Second, the large ambiguity in both grammars was caused in particular by an interaction between the adnominal possession and the constituent questions libraries in which unwanted underspecification led to spurious phrase structure rule application. The adnominal possession library (Nielsen 2018) provides lexical rules for constructions in which possession is marked morphologically (on the possessor, the possessum, or both). At the time this library was developed, only a few analyses within the information structure library exercised non-local features, and (without specific tests for this interaction) it was not apparent that the lexical rules were leaving the non-local features underspecified. A grammar with both adnominal possession lexical rules and an analysis involving, say, head-filler rules for constituent

³⁹More specifically, subject-head phrase was constrained to have an empty SLASH list.

questions will produce (nonsensical) “readings” where a filler-gap rule applies to a structure in which there is no gap (or strictly speaking, where the information about whether or not there is a gap was lost).

More sources of spurious ambiguity were discovered thanks to these two grammars. They include interactions between the analyses of constituent questions, information structure, and clausal modifiers and nominalization libraries. For example, we discovered that the customization system was not properly customizing our base analysis of the question pronoun for grammars that also had nominalizers. This led to nominalization lexical rules applying to question pronouns (while they should only apply to verbs) and ultimately to spurious parses. The same need for a nominalization-blocking constraint was found in the filler-gap rule added by the information structure library to grammars which have clausal modifiers and nominalizers.

Fixing the issues that we found in the Paresi-Haliti grammar dramatically reduced ambiguity in both grammars while also raising the validated coverage of the SOV grammar over the test suite from 39% to 50% percent and of the free word order from 56 to 65%, as presented in Table 4. Future work is required for a meaningful comparison of the two different word order hypotheses, though we can observe that the gain in validated coverage is bigger for the SOV grammar.⁴⁰

Hypothesis	Raw coverage (%)	Validated coverage (%)	Overgeneration (%)	Ambiguity
SOV	41/64 (64.1)	36/64 (56.2)	2/3 (66.7)	4.02
free	51/64 (79.7)	42/64 (65.6)	2/3 (66.7)	3.98

Table 4: Improved grammars for Paresi-Haliti

Long-distance dependencies

4.2

In this section, we discuss a complex interaction between analyses of coordination, adjuncts, and gapped complements. Each analysis has

⁴⁰The higher validated coverage of the free word order grammar does not necessarily mean it is a better hypothesis since the SOV grammar can be developed further such that more orders are covered by the information structure library.

a strong motivation, and they are all relevant for long-distance dependencies. In particular, they all manipulate non-local lists such as SLASH. Taking all the analyses together, non-local lists become over-constrained, incorrectly predicting ungrammaticality for many grammatical sentences. Without computationally implemented grammars, this interaction would almost certainly have gone unnoticed.⁴¹

Since the early days of HPSG, long-distance dependencies have been analyzed using non-local sets or lists such as SLASH (for a historical overview, see Flickinger *et al.* 2021). The Non-local Feature Principle (Pollard and Sag 1994) states that the value of a non-local feature on the mother is the concatenation of the values on the daughters.

An alternative approach, advocated by Bouma *et al.* (2001a), instead passes (or “threads”) the SLASH values of non-head daughters through the head daughter’s SLASH value. This analysis is particularly attractive for modelling lexical items that take gapped complements (see examples in (15)), as it allows each lexical entry to specify how its SLASH list relates to the SLASH lists of its complements, as in (14).

(14)

$$\left[\begin{array}{l} \textit{lexical threading type} \\ \text{SYNSEM} \left[\begin{array}{l} \text{NON-LOCAL|SLASH|APPEND} \quad \langle \boxed{1}, \boxed{2} \rangle \\ \text{LOCAL|CAT|VAL} \end{array} \right] \left[\begin{array}{l} \text{SUBJ} \quad \langle \left[\text{NON-LOCAL|SLASH} \boxed{1} \right] \rangle \\ \text{COMPS} \quad \langle \left[\text{NON-LOCAL|SLASH} \boxed{2} \right] \rangle \end{array} \right] \end{array} \right]$$

In the majority of cases, the SLASH list of the head is simply the concatenation of the SLASH lists of its arguments, as shown in (14). In cases like *eager* and *easy*, one element is first removed from the complement’s SLASH list. This analysis is known as non-local amalgamation (Bouma *et al.* 2001a; Ginzburg and Sag 2000) or lexical threading.

- (15) a. Kim is eager to please.
 b. Kim is easy to please.

⁴¹Specifically, this particular issue was discovered when using the Grammar Matrix in a graduate-level grammar engineering course; see also Section 6.2.

A lexical threading analysis is implemented in the English Resource Grammar (ERG) and was inherited by the Grammar Matrix in its initial construction, using lexical amalgamation of non-local features from arguments and phrasal amalgamation for head-modifier combinations. While integrated into the broad coverage monolingual grammar (ERG), this system was not thoroughly tested in the cross-linguistic Grammar Matrix context until the information structure (Song 2014) and especially the constituent question (Zamaraeva 2021b) libraries were added. On either analysis, the handling of non-local features requires a notion of append (e.g. the SLASH list of the mother is the append of the daughters'). Recall that relational constraints like append are not part of the DELPH-IN variant of the HPSG formalism. The Grammar Matrix initially implemented these appends with difference lists, like the ERG. However, in order to better handle SLASH lists with more than one element as well as for general better maintainability, the Grammar Matrix has moved to using append lists (Zamaraeva and Emerson 2020; see also Section 3.3). We thus had a system which implemented partially lexical and partially phrasal amalgamation of non-local features using append lists, combined with standard HPSG analyses of complementation and modification.

The Grammar Matrix also provides analyses of coordination (Drellishak and Bender 2005), and the interaction between this analysis and the handling of non-local features revealed complexities. To make sure that only compatible phrases can be coordinated, an attractive analysis is to identify large parts of the feature structures for the conjuncts (see Abeillé and Chaves 2021 for a recent review of approaches to coordination in HPSG). However, if these feature structures contain computation types (see Section 3.3), we are identifying not only the outputs of the computation, but the inputs as well. Implementing amalgamation (lexical or phrasal) of non-local features with computation types means that a verb phrase's SLASH contains not only a list, but also the history of append operations. If we identify not only the lists, but also the computation histories, then we have a much stronger constraint on compatibility for coordination.

For example, consider coordinated intransitive and transitive clauses, as illustrated in (16a). The SLASH list of an intransitive verb like *sleep* is precisely the SLASH list of its subject. However, the SLASH list of a transitive VP like *eat bananas* appends the subject's SLASH

list with the empty SLASH list of *bananas*. Appending an empty list results in the original list. However, the feature structure associated with a computation type includes all intermediate steps in the computation. With a different number of empty lists being appended, there is a different feature structure.

- (16) a. Monkeys sleep and eat bananas.
 b. Monkeys sleep soundly and eat bananas.

If we only had a small finite set of valence frames to consider, we could carefully define the append operations, so that the computation histories are compatible for coordination. However, adjuncts pose a problem here, as illustrated in example (16b), because recursive adjunction creates an unbounded set of possible computation histories.

We can see that the analyses of coordination and non-local dependencies are not fully compatible: combining them, the grammar would fail to parse grammatical sentences like (16a–b). We must therefore revise our system of hypotheses. The current approach in the Grammar Matrix involves two changes. First, we only identify the contents of computation types, without their computation histories. This is illustrated in (17), where the SLASH|LIST values are identified, but not the SLASH values. This is sufficient to resolve the problem noted here. In addition, we have dropped the lexical amalgamation of non-local features (although see Section 4.3). This was done in response to this investigation as well as others where the lexical amalgamation approach has made it very difficult to reason about analyses.

- (17)
$$\left[\begin{array}{l} \text{coord-phrase} \\ \text{SYNSEM|NON-LOCAL|SLASH|LIST} \\ \text{LCOORD-DTR|SYNSEM|NON-LOCAL|SLASH|LIST} \\ \text{RCOORD-DTR|SYNSEM|NON-LOCAL|SLASH|LIST} \end{array} \right]$$

This example illustrates how the Grammar Matrix methodology allows for discovery and resolution of conflicts between analyses and how considerations of maintainability (preferring analyses which are expected to be robust to changes elsewhere in the grammars) also impact analytical decisions.

The study of morphological marking of interrogatives by Zamaraeva (2021a, Section 6.8) in the context of developing a Grammar Matrix library for constituent questions turned up a tension between morphology and syntax in DELPH-IN HPSG.⁴² For the full details of this study, we refer the reader to Zamaraeva 2021b and Zamaraeva 2021a, Section 6.8; here we give a brief summary and contextualize the issue with respect to the Grammar Matrix development.⁴³

The tension lies between the analysis of long-distance dependencies (see also Section 4.2) and the analysis required for a particular kind of morphological marking of questions found in e.g. Makah [myh] (Wakashan) which maintains two separate verbal inflection paradigms for polar (18a) and for constituent (18b–c) questions (in addition to the indicative paradigm).

- (18) a. *dudu'k = 'aʃ = qa:k = s*
 sing = TEMP = POLAR = 1SG
 ‘Am I singing?’ [myh] (Davidson 2002, page 100)
- b. *ʔačaq = qa:ʃ dudu'k*
 who = CONTENT.3SG sing
 ‘Who is singing?’ [myh] (Davidson 2002, page 285)
- c. *baqiq = qa:ʃ ti'*
 what = CONTENT.3SG DEM
 ‘What is this?’ [myh] (Davidson 2002, page 285)

In a sense, this particular tension is related to the one presented in Section 4.2: while the syntax of coordination as modelled in the Grammar Matrix made using the non-local amalgamation principle less attractive, the example of Makah points in the opposite direction and suggests that lexical amalgamation of non-local features can still be

⁴²The tension lies not in the implementation of the lexical rules but rather in the treatment of non-local features involved in interrogative constructions.

⁴³The data and the AVM examples in this section are taken directly from Zamaraeva 2021b. We refer the reader to that work for a complete exposition of the issue summarized here.

very useful in the Grammar Matrix. Specifically, Zamaraeva (2021b) offers two alternative analyses for data such as in (18a–c) and shows that the one that uses lexical amalgamation of non-local features is a lot simpler than the one she developed without lexical amalgamation.⁴⁴

In particular, the analysis which assumes non-local amalgamation relies on a straightforward distinction between two lexical rule types, one for polar and one for constituent questions (19).⁴⁵

- (19) a.
$$\left[\begin{array}{l} \textit{polar-lex-rule} \\ \text{SYNSEM|SF} \qquad \textit{ques} \\ \text{DTR|SYNSEM|NON-LOCAL|QUE|LIST} \quad \langle \rangle \end{array} \right]$$
- b.
$$\left[\begin{array}{l} \textit{wh-lex-rule} \\ \text{SYNSEM|SF} \qquad \textit{ques} \\ \text{DTR|SYNSEM|NON-LOCAL|QUE|LIST} \quad \textit{cons} \end{array} \right]$$

The QUE list constraint in (19a) means that an affix which is an instance of this lexical rule type cannot apply to something that has a question word as an argument (e.g. subject or complement). Conversely, a word produced with (19b) must have at least one question word as an argument. This works because, under the lexical threading assumption, lexical heads (e.g. verbs) will all be subtypes of a supertype like (14) and will therefore inherit the constraints stated in (14); their own non-local lists, including the QUE lists, will be implicitly constrained to be a concatenation of their arguments' QUE lists.⁴⁶

⁴⁴Zamaraeva was motivated to develop that version for the Grammar Matrix after lexical amalgamation was removed, in light of the issue discussed in Section 4.2.

⁴⁵Both types are subtypes of a more general lexical rule supertype which in turn is part of the lexical rule hierarchy. The lexical rule hierarchy implements the various ways in which affixes can contribute to the form and/or meaning of a word. Here, only the constraints specific to the subtypes for polar and constituent question lexical rules are shown. The feature name SF stands for sentential force; the possible values for this feature include *question* and *proposition*. The type *cons* stands for non-empty list.

⁴⁶The type shown in (14) focuses on SLASH list but the constraint is exactly the same for all NON-LOCAL lists including QUE.

Any question word will have a non-empty QUE list by definition, and thus the head's list will be non-empty also. It is particularly important that for a type to be a subtype of (14) obviates the need to constrain subjects and complements of this type separately as to whether they are question words; it is sufficient to say that the head's QUE list is not empty.

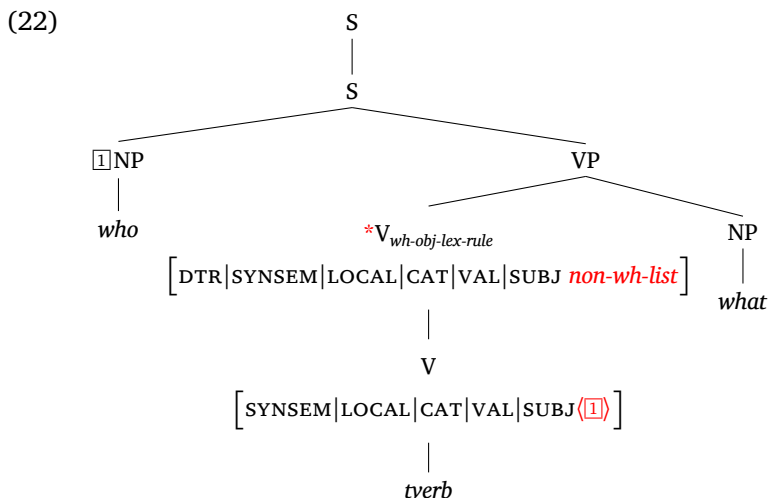
Without the non-local amalgamation assumption, however, heads will not inherit the constraints stated in (14), and it becomes necessary to explicitly constrain the valence lists of heads as to whether they contain question words or not. This necessitates a more complex hierarchy of interrogative lexical rules (shown in (20)) with separate subtypes for cases when a head has a question word as a subject (shown in (20b)) and cases when it has a question word as an object (shown in (20c)), in addition to the subtype for polar questions (shown in (20a)).

- (20) a.
$$\left[\begin{array}{l} \text{polar-lex-rule} \\ \text{SYNSEM|LOCAL|CAT|VAL} \left[\begin{array}{l} \text{SUBJ} \left\langle \left[\text{NON-LOCAL|QUE|LIST} \langle \rangle \right] \right\rangle \\ \text{COMPS} \text{ non-wh-list} \end{array} \right] \end{array} \right]$$
- b.
$$\left[\begin{array}{l} \text{wh-subj-lex-rule} \\ \text{SYNSEM|LOCAL|CAT|VAL|SUBJ} \left\langle \left[\text{NON-LOCAL|QUE|LIST} \text{ cons} \right] \right\rangle \end{array} \right]$$
- c.
$$\left[\begin{array}{l} \text{wh-obj-lex-rule} \\ \text{SYNSEM|LOCAL|CAT|VAL} \left[\begin{array}{l} \text{SUBJ} \text{ non-wh-list} \\ \text{COMPS} \left\langle \left[\text{NON-LOCAL|QUE|LIST} \text{ cons} \right] \right\rangle \end{array} \right] \end{array} \right]$$

Furthermore, Zamaraeva (2021b) shows that, even with this more complex hierarchy, an additional type is needed to constrain some valence lists either to be empty or to not include any question words (21). This is needed in order to avoid spurious ambiguity in cases where there is more than one question word in the sentence. Without these additional constraints, either lexical rule can apply to license the sole affix needed on the verb. Lacking sufficient data from Makah but based on the description in Davidson 2002, Zamaraeva (2021b) shows how this works on one pseudo-language example (11c), presented here

as (22), where a tree which would otherwise be licensed using (20c) is ruled out, leaving only (20b) as the possibility.

- (21)
$$\left[\begin{array}{l} \text{non-wh-cons} \\ \text{FIRST} \quad \left[\begin{array}{l} \text{synsem} \\ \text{NON-LOCAL|QUE|LIST} \quad \langle \rangle \end{array} \right] \\ \text{REST} \quad \text{non-wh-list} \end{array} \right]$$



Zamaraeva’s observation points towards the need for further exploration of how much the Grammar Matrix core should cover versus how much should be offloaded to the customization system, or in other words, which parts of the grammatical system we expect to be in every grammar (see also Section 3.4.5). For example, lexical amalgamation of non-local features could be brought back into the Grammar Matrix but not at the level of the core; instead, it would be provided by the customization system only for languages which seem to require it.

Our observation here is that this tension would be hard to notice without the Grammar Matrix which provides the tools to speed up grammar development (e.g. specifying a system of lexical rules, building here in particular on the robustness of the morphotactics library discussed in Section 5) and at the same time embraces a methodology which requires us to examine such a wide range of typological profiles simultaneously and in such formal detail.

Summary

4.4

In this section, we presented several examples of how the Grammar Matrix allowed us to discover tensions between different analyses which could have gone unnoticed had we attempted to track all the interactions between all the analyses manually without the means of computer programs for generating grammars and then parsing sentences with those grammars. This ability to computationally verify analyses is what supports the explicitness and empiricism of a grammar engineering approach to linguistic hypothesis testing. Identifying the tensions such as described above efficiently guides future work. Work-in-progress analyses such as the ones described here are thus seen as concrete building blocks which ultimately serve to build a robust system of analyses with an explicit area of applicability.

ACCUMULATING EVIDENCE
FOR ANALYSES' ROBUSTNESS

5

Fully implemented systems of syntactico-semantic analyses which are deployed in interaction with each other on test suites from diverse languages not only allow us to detect problems in analyses but also to accumulate, over time, a certain confidence that some analyses in fact work well. If a set of hypotheses continues to account for more and more data from more and more languages over time, it is reassuring in terms of the quality of those hypotheses.

In this section, we present some examples of robustness that have been observed in the Grammar Matrix. Section 5.1 gives an example of how a syntactic phenomenon's analysis can be built directly upon an existing analysis of another part of the grammar; Section 5.2 discusses how the system allows for easy reuse of the existing lexical types to introduce new ones; and finally Section 5.3 shows several analyses which relied heavily on the Grammar Matrix's morphotactics library.

5.1 *Predicative adjectives in polar questions*

An interaction between the polar questions library and the adjectives library is illustrative of how the Grammar Matrix's analyses, rooted in typological analyses, are robust to unseen interactions between libraries that occur as grammars grow in size.

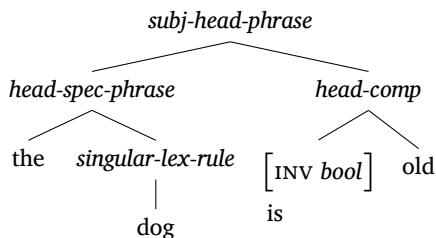
The Grammar Matrix customization system has supported the subject-auxiliary inversion strategy for forming polar questions found in English (e.g. *Is the dog barking?*) since Bender and Flickinger 2005. Poulson's (2011) work on tense/aspect marking and Fokkens's (2010) work on word order further developed the support for auxiliaries. However, the Grammar Matrix did not directly support copulas until Trimble (2014) added them in the context of the adjectives library, as some languages require copulas with predicative adjectives.

Though Trimble's work was done without reference to polar questions in particular, the customization system was able to produce grammars with subject-auxiliary inversion and copulas supporting predicative adjectives with the interaction of these libraries correctly supporting subject-copula inversion with minimal spurious ambiguity that was simple to eliminate. See the examples in (23).

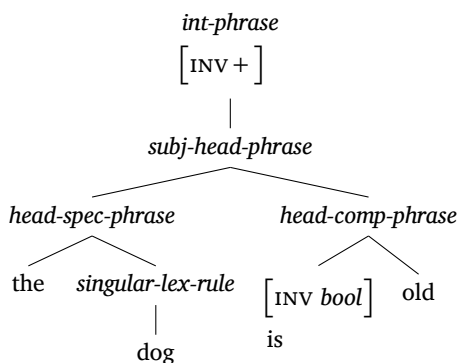
- (23) a. **copula, inverted:** *Is the dog old?* [eng]
 b. **copula, not inverted:** *The dog is old.* [eng]
 c. **auxiliary, inverted:** *Is the dog barking?* [eng]
 d. **auxiliary, not inverted:** *The dog is barking.* [eng]

The copula type introduced by Trimble (2014) was initially underspecified for the feature controlling inversion (INV) and subsequently the inversion phrase structure rule (labelled in (24) as *int* (interrogative)) was spuriously licensed in non-inverted copula phrases, such as *The dog is old*. See both the valid and spurious parses in the simplified schematic set of examples (24). Because the copula is underspecified for INV, even though *int-phrase* bears the constraint [INV +] the spurious parse in (24b) is produced.

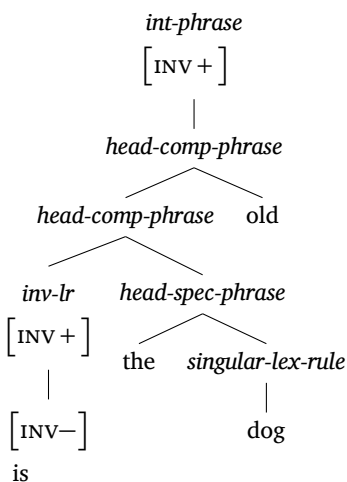
(24) a. **valid analysis**



b. **invalid analysis; spurious application of int-phrase rule**



c. **valid copula inversion analysis**



Grammars that licensed the spurious analysis in (24b) were only produced when both polar inversion and copulas were included in the

specification, a scenario not tested during the development of the polar question library, auxiliary support, or copula support. Once the connection between auxiliary inversion and copulas became apparent, it was a simple matter to include copulas in the list of types to which the existing customization logic for the polar question library was adding the [INV –] constraint. The resulting customization process was confirmed to work correctly for languages with inversion; it licensed inverted and non-inverted questions and did not produce the spurious ambiguity in the non-inverted questions.

This sort of analysis of the interaction between libraries with minimal effort demonstrates the robustness of both libraries and their underlying analyses based on the typological and syntactic literature as well as the grammar customization process.

5.2 *Reusing and extending the lexical type hierarchy*

The original Grammar Matrix of Bender *et al.* (2002) provided a hierarchy of lexical types, which early users of the Grammar Matrix extended by hand, by either creating lexical entries directly instantiating these types or by creating subtypes and then instantiating them. Drellishak (2009) developed the original web questionnaire and customization logic for the lexicon, which exposed a subset of the core grammar's lexical types through the questionnaire and allowed users to define lexical types in an easier and more abstracted way.

Since then, for over a decade, the lexicon's underlying structure as well as the web interface for adding lexical types and lexical entries have served the development of many new libraries, for the most part requiring only minor extensions. Most extensions have involved merely adding new subtypes (and exposing them via the web interface). From the point of view of HPSG, such developments confirm the generality of the original types. This is not to say that there haven't also been revisions to underlying types, reflecting the ability of the project to refine its analyses over time in a data-driven fashion.

HPSG is a lexicalist theory, which means it assumes a large number of lexical types in any grammar. Over the years, the core lexicon structure (consisting of basic supertypes such as *lex-item*

with a hierarchy of subtypes for different kinds of semantic composition and different valence frames) was successfully extended to support more parts of speech (Trimble 2014; Song 2014; Nielsen 2018).

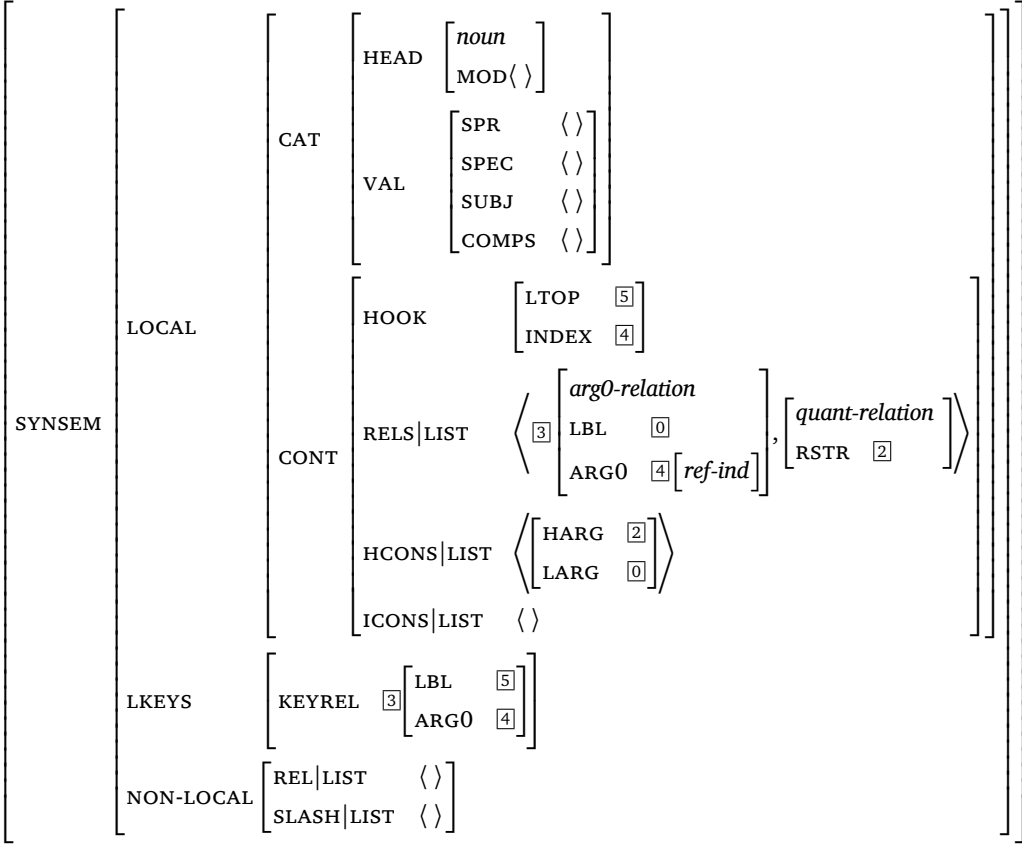
In addition to refining and extending the lexical type hierarchy in the core grammar, the various libraries have also provided a range of types, accessible via customization at need, which apply in some but not all languages. These include such items as the paired adverbs used in Mandarin [cmn] (Sino-Tibetan) and other languages to express clausal connectives like ‘because’, with one adverb in each clause (Howell and Zamaraeva 2018) and copulas in languages that use them with adjectival predicates (Trimble 2014, Section 4.2).

To take an example of how the existing type hierarchy is reused in more detail, consider the recently added support for question words (Zamaraeva 2021a, Section 6.1) which includes lexical types for question pronouns (such as *who/what* in English), question determiners (such as the English *which*), location in space and time adverbs (like the English *where* and *when*), and morphologically simple question verbs such as the Chukchi *req* which can be roughly translated into English as ‘do what?’.⁴⁷ Adding those lexical types was straightforward given the existing Grammar Matrix lexicon framework.

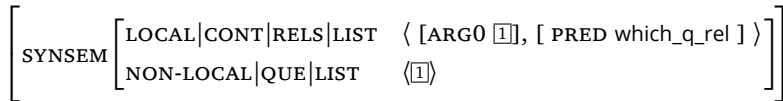
For example, the type for question pronouns is substantially similar to the already existing one for personal pronouns. The constraints shared between the two, shown in (25), model words which are nominal ([HEAD *noun*]), may not serve as modifiers ([MOD ⟨ ⟩]), are lexically saturated in their valence (need no dependents as shown by the empty lists as the values of the VAL features), and introduce two predications under the REL feature (the pronoun’s relation and the associated quantifier). The constraints specific to the question pronoun type are shown in (26), and are limited to two: the specific quantifier (*which_q_rel*, characteristic of constituent questions) and the non-empty value for QUE, used to detect the presence of these words in both the in-situ and head-filler analyses.

⁴⁷ Such verbs do not involve incorporation, according to Hagege (2008, page 7).

(25)



(26)



Similarly, the type for question determiners (not shown) only needs to add, to the basic determiner type, a dependency between itself and the entity for which the determiner serves as the specifier; interrogative verbs (not shown) differ from the already existing verb types in the semantic relations that they introduce (see Sections 6.1, 6.9 in Zamaraeva 2021a for details on all of these lexical types). In

Section 5.3.1, we give an example of how this analysis straightforwardly interacts with another part of the system, the analysis of adnominal possession.

Lexical rules

5.3

Significant progress has been made since 2010 on the support for syntactic phenomena that can be expressed via morphological marking, including adnominal possession, nominalization, evidentiality, information structure, incorporated adjectives, and more (see Table 1). The Grammar Matrix customization system prompts users to provide position classes (groups of lexical rule types which take the same position within a word), lexical rule types, and lexical rule instances which may specify spelling and, in some cases, contribute to the semantics.⁴⁸ The resulting specifications are assembled by the customization system into fairly complex hierarchies of types and sometimes automatically inferred intermediary types for simplifying the type descriptions. The Grammar Matrix's goal in general is to partially automate and therefore speed up grammar development, and its morphotactics library (O'Hara 2008; Goodman 2013) showcases this feature particularly well, as directly writing grammar code for large morphological systems manually would be especially time consuming.

Several significant grammar implementations have tested the limits of the Grammar Matrix morphological systems. Borisova (2010), Bender *et al.* (2014), Crowgey (2019), and the efforts discussed in Section 6.2 have implemented small to large morphological systems using the Grammar Matrix. Bender *et al.* (2012) describe a grammar fragment modelling the lexicon and morphology of Chintang [ctn] (Sino-Tibetan), defined entirely through the customization system, with lexical entries imported automatically from a Toolbox file.⁴⁹ This grammar fragment includes 160 lexical rules for verbs and 24 for nouns, hand-defined via the customization system based on the analysis in Schikowski 2012. Crowgey (2019, Section 5) describes a

⁴⁸ Examples include lexical rules for incorporated adjectives, certain valence changing morphology, and evidentials.

⁴⁹ Toolbox is a data management program designed for linguistic work by SIL International (<https://software.sil.org/toolbox/>).

grammar and morphophonological transducer of Lushootseed [lut] (Salishan) where the grammar includes two general position classes with seven lexical rules, two nominal position classes with four lexical rules, and nine verbal position classes with thirty-four lexical rules. Wax (2014, Section 6.1) describes an automatically specified grammar fragment of French [fra] with seventy position classes and twenty-one lexical types described by 938 individual lines in the grammar specification file that the Grammar Matrix outputs.

Specifying such large morphological systems would take significantly longer without the Grammar Matrix morphotactics library with its web-based user interface and abstracted grammar specifications and automatic hierarchy creation. But the morphological support in the Grammar Matrix extends well beyond morphotactics. In this subsection, we briefly survey examples of libraries that build on the morphotactic infrastructure to provide typologically broad support for a range of phenomena: adnominal possession, fine-grained differences between attributive and predicative adjectives, incorporated adjectives, and valence changing morphology.

5.3.1

Adnominal possession

Nielsen (2018) introduces a Grammar Matrix library for adnominal possession. For languages where the expression of adnominal possession involves affixes indicating features (e.g. person/number) of the possessor, this library makes use of the case library (Drellishak 2009) and the morphotactics library (O'Hara 2008; Goodman 2013). Lexical rule supertypes added to the Grammar Matrix system to support adnominal possession provide an example of how an implemented grammar based on a robust analysis can make correct predictions which humans working without the aid of a computer may overlook.

Zamaraeva (2021a, page 369) describes how, in the process of testing the constituent questions library on a held-out language, Jalkunan [bxl] (Mande), she added to the grammar specification a lexical entry for a question pronoun but was unable to specify an entry for the corresponding possessive pronoun since in the process of development, possessive question pronouns were left unimplemented due to time constraints. However, it turned out that the Grammar Matrix

system already contains everything necessary to produce the correct analysis for the Jalkunan example in question (27), thanks to the adnominal possession library having a lexical rule that can turn a personal pronoun into a possessive pronoun, and thanks to the constituent questions library implementing question pronouns very similarly to how personal pronouns are implemented (see also Section 5.2).

- (27) mā?ā-nĩ sàá = Ø nè = Ø
 who-INDEP house = be there = Q
 ‘Whose house is that?’ [bxl] (Heath 2017, page 273)

The DMRS⁵⁰ artifact in Figure 11 shows the correct semantics for the sentence obtained automatically with the Grammar Matrix-generated grammar: the questioned element is a person (which_q and person), the person is the possessor of the house (ARG1 of the possession relation is the possessum, ARG2 is the possessor), and the house is located in some space. The importance of this example is that it was not targeted when the grammar specification for Jalkunan was put together, and yet the resulting grammar provided appropriate analyses. This shows that the analysis of adnominal possession (Nielsen 2018) robustly generalizes to constituent questions, even though constituent questions were not part of the Grammar Matrix when the analysis for adnominal possession was developed and tested.

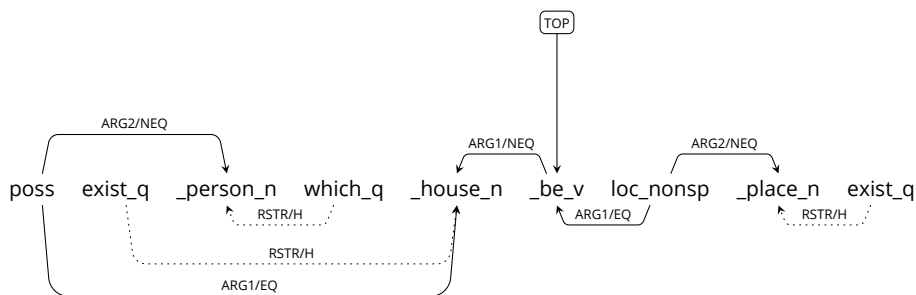


Figure 11: DMRS for sentence (27). The structure can be read as: *Which person possesses the house there?*

⁵⁰ DMRS (Copestake 2009) is Dependency MRS, which represents the same information as MRS, but in a dependency graph.

Trimble (2014) introduces a Grammar Matrix library for adjectives in attributive and predicative constructions. The customization system's design to provide custom types in addition to the core grammar proved particularly useful for several adjectival phenomena, enabling a more parsimonious analysis for most languages while accommodating several phenomena that were in conflict with the main analysis, particularly constrained argument agreement and switching adjectives (Stassen 2013).

Languages such as German [deu] (Indo-European) show what Trimble (2014, pages 76–79) calls constrained argument agreement. In these languages, a class of adjectives has one set of morphology in the attributive construction and a different set in a predicative construction. In German, agreement morphology is licensed in the attributive construction and is not licensed in the predicative construction.

- (28) a. Der große Hund bellte.
 DET.M.NOM.SG big.M.NOM dog bark.PST
 'The big dog barked.' [deu] (adapted from Hankamer and Lee-Schoenfeld 2005)
- b. Ich bin groß.
 1SG COP.PRES.1SG big
 'I am tall.' [deu] (adapted from Landman and Morzycki 2003)

gClimb (Fokkens 2014) includes an analysis for this construction where a single position class contains both lexical rules for the agreement morphology used in the attributive construction and a non-inflecting lexical rule which allows the adjective to be the complement of a copula (this rule specifies [PRD +] on the adjective). This way, adjectives can either undergo the predicative lexical rule and be licensed as copula complements or undergo one of the agreement rules and be licensed in the attributive position.

Trimble (2014, pages 74–79) finds a related behaviour of adjectival morphology that Stassen (2013) calls *switching* in languages like Maori [mri] (Austronesian). A single adjective class may be licensed with different sets of syntax and morphology under different circum-

stances, such as one set similar to nouns (see (29b)) and one to verbs (see 29d)).

(29) a. **Nominal predicate:**

he kiwi teera manu
DET.INDF kiwi this bird
'This bird is a kiwi.' [mri] (Biggs 1969, page 90, Stassen 2013)

b. **Noun-like adjectival predicate:**

he pai te koorero
INDF good DET.DEF talk
'The talk is good.' [mri] (Biggs 1969, page 14, Stassen 2013)

c. **Verbal predicate:**

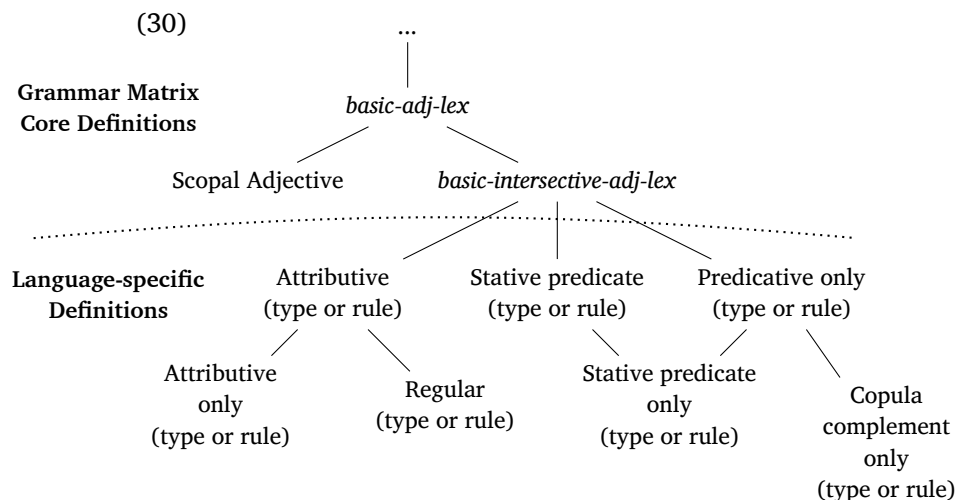
ka oma te kootiro
INCEP run DET.DEF girl
'The girl runs.' [mri] (Biggs 1969, page 18, Stassen 2013)

d. **Verb-like adjectival predicate:**

ka pai te whare nei
INCEP good DET.DEF house this
'This house is good.' [mri] (Biggs 1969, page 6, Stassen 2013)

The cross-linguistic analysis that is fundamental to the Grammar Matrix leads to the juxtaposition of constrained argument agreement in German and switching adjectives in Maori, offering the opportunity for the same analysis to work in both situations. Like in Fokkens 2014, Trimble's (2014, pages 74–79) analyses of adjectives in languages like German and Maori define adjective types without specifics about their syntactic licensing, such as whether or not the types are licensed as a copula complement or as a matrix predicate, and the syntactic behaviour and morphology are defined through lexical rules. A single required position class is defined with two non-inflecting lexical rule types, one which acts as the base analysis (such as tense marking or being licensed as a copula complement) and one for the other behaviour (such as agreement or being licensed in the attributive construction).

Given the need to support languages like German and Maori, as well as any languages which don't have any morphological reflection of the syntactic distribution of adjectives, the question arises: what should go into the cross-linguistic core grammar? Making a position class like the one posited for German and Maori necessary for all languages would push unwanted complexity into languages where it is not necessary. The solution adopted is to leverage the type hierarchy and the possibility of spreading information between cross-linguistically shared supertypes and language-specific subtypes.



In order to accommodate both constrained argument agreement and switching adjectives, the customization system provides a type hierarchy (shown in (30)) with basic adjectival type definitions (HEAD type, scopal versus intersective adjectives, intersective adjectives modify nouns, etc.) in the core grammar and more specific types (attributive versus predicative, copula complement versus stative predicate, etc.) in the language-specific subtypes. Subsequently, the customization system can provide additional lexical types in the language-specific subtypes if neither constrained argument agreement nor switching adjectives are present; and if either of these phenomena are present, it provides lexical rules which specify the correct syntactic behaviour. The source of the robustness of this approach lies in the combination of the expressive power of the Grammar Matrix's

lexical rule infrastructure alongside the basic architecture of splitting the core grammar and the language-specific subtypes to build custom analyses based on the typological facts of the language.

Yup'ik [esu] (Inuit-Yupik-Unangan), Penobscot [aaq-pen] (Algic), and other languages have attributive adjectives that appear as affixes on nouns (see (31a)) usually in addition to adjectives that appear as words (see (31b)) (Miyaoka 2012, page 101, Quinn 2006, pages 28–29). Whereas most morphology either does not add predicates or adds grammatical predicates (such as negation), adjectives are an open class of morpheme with one-to-one morpheme to predicate mapping.

- (31) a. qayar-pa-ngqer-tuq
kayak-big-have-IND.3SG
'He has a big kayak.' [esu] (Miyaoka 2012, page 136)
- b. nutaraq angyaq ang'-uq
new.thing.ABS.SG boat.ABS.SG big-IND.3SG
'The new boat is big.' [esu] (Miyaoka 2012, page 466)

While implementing adjectival lexemes in the Grammar Matrix required reworking of the lexical hierarchy to support adjectives cross-grammatically (Trimble 2014, pages 76–79), the analysis of incorporated adjectives introduced by Trimble (2014, page 79) required neither changes to the core grammar nor the development of new kinds of questionnaire logic. Rather, it was possible to build it by combining already existing functionality. The existing lexical rule infrastructure (O'Hara 2008; Goodman 2013), both in terms of the core grammar and the customization system, was able to handle this added functionality with no extensions required. Existing functionality in the user interface for specifying predicates on lexical items was used to allow specifying predicates for adjectives as affixes on nouns; and existing customization logic and core grammar functionality resulted in grammar fragments that correctly captured the facts of the language (Trimble 2014, pages 128–129).

Curtis (2018a) describes an implementation of valence-changing morphological operations, including passives, causatives, benefactives, and applicatives found in many language families. These analyses were built upon the existing foundation of argument structure in the Grammar Matrix, primarily the separation of syntactic roles in valence lists and semantic roles in argument slots. Starting from the broad typological groupings of valence-changing morphology as described in Haspelmath and Müller-Bardey 2001, pages 4–14, the implementation focused on decomposing the analyses of these operations into fine-grained atomic rule components. For example, in Zulu the benefactive and motive affixes are homonymous and differ only in the semantic predicate contributed and noun class constraint on the added object (Buell 2005):

(32) a. **Benefactive:**

Ngilahlela	uThandi	udoti
Ngi-lahl-el-a	u-Thandi	u-doti
1S.SBJ-dispose.of-APPL-FV	1-1.Thandi	1-1.trash

‘I’m taking out the trash for Thandi.’ [zul] (Buell 2005, page 189)

b. **Motive:**

Ngilahlela	imali	udoti
Ngi-lahl-el-a	i-mali	u-doti
1S.SBJ-dispose.of-APPL-FV	9-9.money	1-1.trash

‘I’m taking out the trash for money.’ [zul] (Buell 2005, page 189)

The library generates distinct, atomic lexical rule components for these operations, for example specifying *only* the PRED value of the added predicates as in (33a) and (33b). These are then assembled by the customization system – along with the common applicative rule (33c) and added argument rule (33d), which together add the new syntactic argument and connect it to the new semantic predicate – into lexical rule hierarchies that lexical rule instances then inherit from directly.

- (33) a.
$$\left[\begin{array}{l} \textit{benefactive-pred-lex-rule} \\ \text{C-CONT} | \text{RELS} | \text{LIST} \left\langle \left[\text{PRED} \textit{benefactive_rel} \right] \right\rangle \end{array} \right]$$
- b.
$$\left[\begin{array}{l} \textit{motive-pred-lex-rule} \\ \text{C-CONT} | \text{RELS} | \text{LIST} \left\langle \left[\text{PRED} \textit{motive_rel} \right] \right\rangle \end{array} \right]$$
- c.
$$\left[\begin{array}{l} \textit{basic-applicative-lex-rule} \\ \text{C-CONT} \left[\begin{array}{l} \text{RELS} | \text{LIST} \left\langle \left[\textit{event-relation} \right] \right\rangle \\ \text{HCONS} | \text{LIST} \langle \rangle \end{array} \right] \\ \text{DTR} | \text{SYNSEM} | \text{LOCAL} | \text{CONT} | \text{HOOK} | \text{INDEX} \quad \boxed{1} \end{array} \right]$$
- d.
$$\left[\begin{array}{l} \textit{added-arg3of3-lex-rule} \\ \text{SYNSEM} | \text{LOCAL} | \text{CAT} | \text{VAL} | \text{COMPS} \left\langle \boxed{2}, \left[\text{LOCAL} \left[\begin{array}{l} \text{CAT} | \text{VAL} \left[\begin{array}{l} \text{SPR} \langle \rangle \\ \text{COMPS} \langle \rangle \end{array} \right] \\ \text{CONT} | \text{HOOK} | \text{INDEX} \quad \boxed{1} \end{array} \right] \right] \right\rangle \\ \text{C-CONT} | \text{RELS} \left\langle \left[\text{ARG2} \quad \boxed{1} \right] \right\rangle \\ \text{DTR} | \text{SYNSEM} | \text{LOCAL} | \text{CAT} | \text{VAL} | \text{COMPS} \quad \langle \boxed{2} \rangle \end{array} \right]$$

Similar decompositions into blocks were implemented for object-removing, subject-adding, and subject-removing operations, supporting implementation of e.g. (respectively) deobjective, causative, and passive morphology. These phenomena were added to the Grammar Matrix customization system without any additional theoretical additions or machinery: valence-changing operations are specified on position classes and lexical rule types using the existing morphotactics library, and the rule components operate on valence lists, semantic predicates, case marking, and scopal arguments using the existing Grammar Matrix mechanisms. Using these existing mechanisms and stored analyses to create new abstractions at intermediate levels of detail enables grammar engineers to more directly model the commonalities and variations in how valence change is expressed.

The Grammar Matrix represents a large and complex system of HPSG analyses accounting for data in a wide typological range. Crucially, by rigorously testing said analyses against data from many languages over the years, the project has demonstrated the robustness of the analyses that it houses, particularly of the implementation of the lexical types and the lexical rule types generally and the analyses of specific phenomena relying on those types. In this section, we talked about how it was possible to elegantly extend the system to include various types of adjectives, question pronouns, and valence-changing morphology, to name just several of the phenomena that were successfully added over the years. Crucially, the analyses for all these phenomena demonstrably work together as a system, even in cases when the system engineers did not deliberately consider some of the possible interactions.

DISCOVERING COMPLEXITY THROUGH USE

Where the data driven methodology described in Section 3.4.1 and the attention to interacting phenomena such as described in Section 4 and Section 5 rigorously test the robustness of the Grammar Matrix's analyses, the use of the Grammar Matrix outside its own development offers the opportunity to vet these analyses and corresponding implementations at scale. External grammars have the potential to extend the vetting of the Grammar Matrix's analyses and reveal further unforeseen interactions, by incorporating more phenomena and larger and more complex lexical and morphological systems. Three examples of such use of the Grammar Matrix are: (i) the AGGREGATION project (Bender *et al.* 2014; Zamaraeva *et al.* 2019a; Howell 2020; Howell and Bender 2022, among others), which uses the Grammar Matrix to generate grammars automatically on the basis of linguistic corpora; (ii) an annually offered graduate-level grammar engineering course at University of Washington, in which students use the Grammar Matrix as a starting point and build out the grammars by hand;

and (iii) the broader use of the Grammar Matrix by the research community to develop larger DELPH-IN grammars for linguistic analysis. The sizes of these grammars and their associated test suites, compared with typical Grammar Matrix development grammars, allow the testing of analyses for more interacting phenomena at once. Furthermore, both AGGREGATION and the grammar engineering course place a focus on under-documented languages; and in doing so, provide further evaluation of the Grammar Matrix's analyses in terms of cross-linguistic generalizability.

The AGGREGATION project

6.1

The configurable nature of the Grammar Matrix's grammar specifications makes this toolkit particularly well-suited to support grammar inference. Grammar inference (Bender *et al.* 2014; Zamaraeva *et al.* 2019a) is the practice of automatically generating grammars from partially annotated text and some external source of linguistic knowledge (Howell 2020, page 6; Howell and Bender 2022, page 2).⁵¹ The AGGREGATION Project leverages two sources of linguistic knowledge – interlinear glossed text (IGT) and the Grammar Matrix customization system – to automatically create machine-readable grammars by inferring grammar specifications for the latter from the former.

Unlike grammars created by linguists using the Grammar Matrix customization system directly, grammars inferred by the AGGREGATION Project's morphological inference system (MOM; Wax 2014; Zamaraeva 2016; Zamaraeva *et al.* 2017) and syntactic inference system (BASIL; Howell 2020; Howell and Bender 2022) are brought to scale much more quickly. The rapid scaling offered by automatic inference allows for a degree of complexity that is much more difficult to reach when defining a grammar by hand.

In particular, the MOM morphological inference system infers lexical entries and morphological rules for each form attested in the corpus of IGT data. These lexical entries and rules are merged into larger

⁵¹ Howell (2020) and Howell and Bender (2022) define the term *grammar inference* in the context of the AGGREGATION project; while earlier work such as Bender *et al.* 2014 and Zamaraeva *et al.* 2019a refers to grammar inference or describes it as a practice but does not provide a definition.

classes, resulting in grammars that account for a variety of previously unseen strings in a language. However, as Howell (2020, pages 123–128) and Howell and Bender (2022, pages 29–30) observe, such a large set of morphological forms can lead to unforeseen sources of ambiguity. Ambiguity in Howell and Bender’s grammars could be traced back both to inference and to the Grammar Matrix and progress towards reducing this ambiguity was made by Conrad (2021). The discovery of these sources of ambiguity was possible because of the AGGREGATION project’s ability to build large-scale grammars quickly and in turn reveal complexity in language or models of language that would otherwise be difficult to find.

6.2 *Grammar Matrix-based grammar development in graduate-level curriculum*

The Grammar Matrix has been used in a graduate-level multilingual grammar engineering course that has been taught annually at the University of Washington since 2004 (see Bender 2007). In this course, students begin by customizing grammars using descriptive resources and the Grammar Matrix customization system; and then continue to build those grammars beyond the customization system’s functionality to achieve greater coverage over the ten week course. Since 2004, over 125 languages have been studied in this course, including many that are the subject of active language documentation projects. The course has served as an important testbed for new Grammar Matrix libraries as they are developed and as a test of the robustness of the system. Each grammar tests at minimum the lexicon and morphological systems; and most exercise the lexicon, morphology, agreement, case, tense-aspect-mood, and coordination libraries. Over the years, each of the other libraries have been tested by multiple languages in connection with the other libraries included in that year’s curriculum. The result has been thorough debugging of Grammar Matrix libraries using combinations of phenomena that were not necessarily considered or tested during library development (such as what we describe in Section 4.2).

To document the analysis that has come from this course, many of the resulting grammar specifications, test suites, and grammar frag-

ments are available via Language CoLLAGE (Bender 2014). In addition, some grammars from the class have resulted in publications focussed on testing specific hypotheses, such as analyzing Kolyma Yukaghir [yux] (isolate) as having focus case (Zamaraeva and Bender 2014) and suggesting a previously unconsidered negation strategy in Nanti [kox̩] (Arawakan) (Inman and Morrison 2014).

Since 2019, students have used grammar specifications produced by the AGGREGATION system (Section 6.1) as input to the customization system (as opposed to creating the specification from scratch using the Grammar Matrix's web-based questionnaire). The incorporation of the AGGREGATION project in the grammar engineering course enables grammars to include both the lexical and morphological complexity offered by inference from corpora and consequently increases the number of phenomena that can be covered in the course. As a result the potential scale of these grammars increases, allowing for even more testing as the Grammar Matrix continues to grow.

Larger DELPH-IN grammars

6.3

In addition to the relatively small grammars generated by AGGREGATION or developed by students, the Grammar Matrix has given rise to a number of larger grammars and resource grammars. Mid-sized grammars have been developed for the following languages using the Grammar Matrix customization system as a starting point and refining and adding analyses for additional phenomena: Bulgarian [bul] (BURGER; Osenova 2010, 2011), Dutch [nld] (gCLIMB Dutch; Fokkens 2011), Hausa [hau] (HaG; Crysmann 2012), Hebrew [heb] (HeGram; Greshler *et al.* 2015), Indonesian [ind] (INDRA; Moeljadi *et al.* 2015), Lushootseed [lut] (Crowgey 2019), Mandarin [cmn] and Cantonese [yue] Chinese (ManGO; Chunlei and Flickinger 2014, Zhong; Fan *et al.* 2015; Fan 2018), Nuuchahnulth [nuk] (Inman 2019), Thai [tha] (Slayden 2011), and Wambaya [wmb] (Bender 2010). The Grammar Matrix has also given rise to even larger grammars that have undergone long-term development, including for German [deu] (gCLIMB; Fokkens 2011), Korean [kor] (KRG; Kim and Yang 2003; Song *et al.* 2010), Modern Greek [ell] (MGRG; Kordoni and Neu 2005), Norwegian [nob] (NorSource; Hellan and Haugereid 2003), Portuguese [por]

(LXGram; Costa and Branco 2010), and Spanish [spa] (SRG; Marimon 2010).

In addition to the benefits of kick-starting development, these grammars have also benefited from standardization to Minimal Recursion Semantics provided by the Grammar Matrix. DELPH-IN grammars generate MRS representations for sentences which can be used in applications which benefit from semantic representations, as well as to create rich, consistent semantic annotations (see Bender *et al.* 2015, among others). The consistency of MRS representations generated by DELPH-IN grammars is useful for such applications as machine translation (Copestake *et al.* 1995; Bond *et al.* 2011). Even some grammars that did not start by using the Grammar Matrix – such as the Japanese resource grammar (Jacy; Siegel *et al.* 2016) – have incorporated analyses from the Grammar Matrix to maintain consistency of MRS representations with other DELPH-IN grammars. Examination of these grammars and how they have diverged from Grammar Matrix analyses during development poses an interesting area for further work to understand how those analyses hold up at scale.

7 CONCLUSION AND FUTURE DIRECTIONS

In this paper, we presented a resource, a system of methodologies, and multiple specific examples of how syntactic hypotheses can be studied rigorously in complex interactions using the HPSG formalism and the Grammar Matrix meta-grammar engineering framework (Bender *et al.* 2002, 2010). We looked back at some of the various contributions made to the Grammar Matrix project since 2010, summarizing the range of phenomena that can currently be modelled and the typological range that the system demonstrably covers.

The first ten years of Grammar Matrix development were characterized by the initial abstraction of the core grammar from the English Resource Grammar, the innovation of the idea of libraries of analyses of cross-linguistically variable phenomena and the orientation to linguistic typology as a key source of data and analyses, and the

development of a regression testing regime for those artifacts. Drellichak's 2009 project began with organizing imagined Grammar Matrix libraries into a (software) dependency tree, leading him to focus on core libraries that others would necessarily depend on (such as case, agreement, and support for the lexicon).

After another decade plus of development, the Grammar Matrix has grown in complexity, with more libraries covering more complex phenomena, allowing for testing more interactions, beginning to realize the promise of linguistic hypothesis testing at a large scale, across languages and ranges of phenomena within those languages. The CLIMB (e.g. Fokkens 2014) and AGGREGATION (e.g. Howell 2020) projects have pushed the boundaries on linguistic hypothesis testing in two further directions: (i) maintaining alternative hypotheses over the course of grammar development and (ii) producing working grammar fragments from the products of field linguistic research.

Where can we expect this project to go in the next decades? There are still many linguistic phenomena to be modelled in Grammar Matrix libraries, including adverbs (other than adverbial clauses and question adverbs), noun compounds, relative clauses, a broader range of valence types (including raising and control phenomena), as well as phenomena which are high frequency in naturally occurring speech collected in language documentation projects such as reported speech, greetings, and discourse markers. There is also near-term work to be done on interactions discovered (and not yet resolved) such as the ones described in Section 4. The addition of computation types to the grammar engineer's toolkit also opens up the possibility for streamlining some existing analyses and making the resulting grammars accordingly easier to maintain. An important direction for future work is seeing how the Grammar Matrix can be informed by its descendant grammars once they are developed beyond the start that the Grammar Matrix provided.

The Grammar Matrix has shown the value of computational implementation to reproducibility in the study of grammar and the concomitant possibility of building directly on the results of others. As the system grows, it also opens up further avenues for study not previously accessible, such as seeking to differentiate which aspects of model complexity are inherent to complexity in the modelled domain and which are consequences of formal or analytical choices.

ACKNOWLEDGMENTS

Olga Zamaraeva's work is supported by Prof. Gómez Rodríguez's funding from the European Research Council (ERC), under the European Union's Horizon 2020 research and innovation programme (FASTPARSE, grant agreement No 714150), from ERDF / MICINN-AEI (TIN2017-85160-C2-1-R, PID2020-113230RB-C21), from Xunta de Galicia (ED431C 2020/11), and from Centro de Investigación de Galicia "CITIC", funded by Xunta de Galicia and the European Union (ERDF – Galicia 2014–2020 Program), by grant ED431G 2019/01.

Grammar Matrix development has been supported by the US National Science Foundation under grants No BCS-0644097, BCS-1160274, and BCS-561833.

We are grateful for the efforts and input of the broader community of Grammar Matrix developers, which includes, beyond the authors of this paper: Elizabeth Conrad, Joshua Crowgey, Laurie Dermer, Scott Drellishak, Chris Evans, Dan Flickinger, Varya Gracheva, Mike Haeger, Scott Halgrim, Joshua Hou, Tom Liu, Daniel P. Mills, Elizabeth Nielsen, Kelly O'Hara, Stephan Oepen, Laurie Poulson, Safiyyah Saleem, Sanghoun Song, David Wax, and Yi Zhang. The work reported here has benefitted greatly from the effort of the past students of Ling 567 at the University of Washington as well as our colleagues in the DELPH-IN Consortium.

We thank the three anonymous reviewers for their constructive criticism and detailed comments.

REFERENCES

Anne ABEILLÉ and Rui P. CHAVES (2021), Coordination, in Stefan MÜLLER, Anne ABEILLÉ, Robert D. BORSLEY, and Jean-Pierre KOENIG, editors, *Head-Driven Phrase Structure Grammar: The handbook*, pp. 723–774, Language Science Press, Berlin, doi:10.5281/zenodo.5599848.

Peter ACKEMA, Patrick BRANDT, Maaïke SCHOORLEMMER, and Fred WEERMAN, editors (2006), *Arguments and agreement*, Oxford University Press, Oxford.

- Gabriel AGUILA-MULTNER and Berthold CRYSMANN (2018), Feature resolution by lists: The case of French coordination, in Annie FORET, Greg KOBELE, and Sylvain POGODALLA, editors, *Proceedings of International Conference on Formal Grammar*, pp. 1–15, Springer, New York.
- Alexandra AIKHENVALD (2004), *Evidentiality*, Oxford University Press, Oxford.
- Jason BALDRIDGE, Sudipta CHATTERJEE, Alexis PALMER, and Ben WING (2007), DotCCG and VisCCG: Wiki and programming paradigms for improved grammar engineering with OpenCCG, in Tracy Holloway KING and Emily M. BENDER, editors, *Proceedings of the 2007 Workshop on Grammar Engineering Across Frameworks*, pp. 5–25, CSLI Publications, Stanford, CA.
- Kent BECK (2003), *Test-driven development: by example*, Addison-Wesley, Boston, MA.
- Emily M. BENDER (2007), Combining research and pedagogy in the development of a crosslinguistic grammar resource, in Tracy Holloway KING and Emily M. BENDER, editors, *Proceedings of the 2007 Workshop on Grammar Engineering Across Frameworks*, pp. 26–45, CSLI Publications, Stanford, CA.
- Emily M. BENDER (2008), Grammar engineering for linguistic hypothesis testing, in Nicholas GAYLORD, Alexis PALMER, and Elias PONVERT, editors, *Computational Linguistics For Less-studied Languages*, pp. 16–36, CSLI Publications, Stanford, CA.
- Emily M. BENDER (2010), Reweaving a grammar for Wambaya: A case study in grammar engineering for linguistic hypothesis testing, *Linguistic Issues in Language Technology*, 3(3):1–34, doi:10.33011/lilt.v3i.1215.
- Emily M. BENDER (2014), Language CoLLAGE: Grammatical description with the LinGO Grammar Matrix, in Nicoletta CALZOLARI, Khalid CHOUKRI, Thierry DECLERCK, Hrafn LOFTSSON, Bente MAEGAARD, Joseph MARIANI, Asuncion MORENO, Jan ODIJK, and Stelios PIPERIDIS, editors, *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC'14)*, pp. 2447–2451, Paris, http://www.lrec-conf.org/proceedings/lrec2014/pdf/639_Paper.pdf.
- Emily M. BENDER, Joshua CROWGEY, Michael Wayne GOODMAN, and Fei XIA (2014), Learning grammar specifications from IGT: A case study of Chintang, in Jeff GOOD, Julia HIRSCHBERG, and Owen RAMBOW, editors, *Proceedings of the 2014 Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pp. 43–53, Baltimore, MD, doi:10.3115/v1/W14-2206, <https://aclanthology.org/W14-2206>.
- Emily M. BENDER, Scott DRELLISHAK, Antske FOKKENS, Laurie POULSON, and Safiyyah SALEEM (2010), Grammar customization, *Research on Language and Computation*, 8:1–50, doi:10.1007/s11168-010-9070-1.
- Emily M. BENDER and Guy EMERSON (2021), Computational linguistics and grammar engineering, in Stefan MÜLLER, Anne ABEILLÉ, Robert D. BORSLEY,

and Jean-Pierre KOENIG, editors, *Head-Driven Phrase Structure Grammar: The handbook*, pp. 1103–1150, Language Science Press, Berlin.

Emily M. BENDER and Dan FLICKINGER (2005), Rapid prototyping of scalable grammars: Towards modularity in extensions to a language-independent core, in *International Joint Conference on Natural Language Processing*, pp. 203–208, Jeju, <https://aclanthology.org/I05-2035>.

Emily M. BENDER, Dan FLICKINGER, and Stephan OEPEN (2002), The Grammar Matrix: An open-source starter-kit for the rapid development of cross-linguistically consistent broad-coverage precision grammars, in John CARROLL, Nelleke OOSTDIJK, and Richard SUTCLIFFE, editors, *Proceedings of the Workshop on Grammar Engineering and Evaluation at the 19th International Conference on Computational Linguistics*, pp. 8–14, Taipei.

Emily M. BENDER, Dan FLICKINGER, Stephan OEPEN, Woodley PACKARD, and Ann COPESTAKE (2015), Layers of interpretation: On grammar and compositionality, in Matthew PURVER, Mehrnoosh SADRADEH, and Matthew STONE, editors, *Proceedings of the 11th International Conference on Computational Semantics*, pp. 239–249, Paris, <https://aclanthology.org/W15-0128>.

Emily M. BENDER, Laurie POULSON, Scott DRELLISHAK, and Chris EVANS (2007), Validation and regression testing for a cross-linguistic grammar resource, in Timothy BALDWIN, Mark DRAS, Julia HOCKENMAIER, Tracy Holloway KING, and Gertjan VAN NOORD, editors, *ACL 2007 Workshop on Deep Linguistic Processing*, pp. 136–143, Prague, doi:10.3115/1608912.1608934, <https://aclanthology.org/W07-1218>.

Emily M. BENDER, Robert SCHIKOWSKI, and Balthasar BICKEL (2012), Deriving a lexicon for a precision grammar from language documentation resources: A case study of Chintang, in Martin KAY and Christian BOITET, editors, *Proceedings of the 24th International Conference on Computational Linguistics*, pp. 247–262, Mumbai, <https://aclanthology.org/C12-1016>.

Manfred BIERWISCH (1963), *Grammatik des deutschen Verbs (Grammar of German verbs)*, Akademie Verlag, Berlin, doi:10.2307/411946.

Bruce BIGGS (1969), *Let's learn Maori*, Reed, Wellington, doi:10.1086/465322.

Francis BOND, Stephan OEPEN, Eric NICHOLS, Dan FLICKINGER, Erik VELLDAL, and Petter HAUGEREID (2011), Deep open-source machine translation, *Machine Translation*, 25(2):87–105.

Irina BORISOVA (2010), *Implementing Georgian polypersonal agreement through the LinGO Grammar Matrix*, Master's thesis, Saarland University.

Gosse BOUMA, Robert MALOUF, and Ivan A. SAG (2001a), Satisfying constraints on extraction and adjunction, *Natural Language and Linguistic Theory*, 19(1):1–65, doi:10.1023/A:1006473306778.

Gosse BOUMA, Gertjan VAN NOORD, and Robert MALOUF (2001b), Alpino: Wide-coverage computational analysis of Dutch, in Walter DAELEMANS, Khalil SIMA'AN, Jorn VEENSTRA, and Jakub ZAVREL, editors, *Selected papers from the Eleventh Meeting for Computational Linguistics in the Netherlands*, pp. 45–59, Rodopi, Amsterdam, doi:10.1163/9789004333901_004.

Ana Paula Barros BRANDÃO (2014), *A reference grammar of Paresi-Haliti (Arawak)*, Ph.D. thesis, University of Texas at Austin.

Leston Chandler BUELL (2005), *Issues in Zulu verbal morphosyntax*, Ph.D. thesis, University of California, Los Angeles.

Miriam BUTT and Tracy Holloway KING (2002), Urdu and the Parallel Grammar Project, in *Proceedings of the 3rd Workshop on Asian Language Resources and International Standardization*, Taipei, doi:10.3115/1118759.1118762.

Miriam BUTT, Tracy Holloway KING, María-Eugenia NIÑO, and Frédérique SEGOND (1999), *A grammar writer's cookbook*, CSLI Publications, Stanford, CA.

Joan L. BYBEE, Revere Dale PERKINS, William PAGLIUCA, et al. (1994), *The evolution of grammar: Tense, aspect, and modality in the languages of the world*, University of Chicago Press, Chicago, IL.

Daniel BÜRING (2009), Towards a typology of focus realization, in Malte ZIMMERMANN and Caroline FÉRY, editors, *Information Structure: Theoretical, Typological, and Experimental Perspectives*, pp. 177–205, Oxford University Press, Oxford, doi:10.1093/acprof:oso/9780199570959.001.0001.

Ulrich CALLMEIER (2000), PET – A platform for experimentation with efficient HPSG processing techniques, *Natural Language Engineering*, 6:99–108.

Marie-Hélène CANDITO (1999), *Organisation modulaire et paramétrable de grammaires électroniques lexicalisées. Application au français et à l'italien (Modular and parameterized organisation of lexicalised electronic grammars. Applications to French and Italian)*, Ph.D. thesis, Université Paris 7.

Robert L. CARPENTER (2005), *The logic of typed feature structures: With applications to unification grammars, logic programs and constraint resolution*, Cambridge University Press, Cambridge.

Yang CHUNLEI and Dan FLICKINGER (2014), ManGO: Grammar engineering for deep linguistic processing, *Data Analysis and Knowledge Discovery*, 30(3):57–64.

Liviu CIORTUZ and Vlad SAVELUC (2012), *Fluid Construction Grammar and feature constraint logics*, pp. 289–311, Springer, Berlin, doi:10.1007/978-3-642-34120-5_12.

Liviu-Virgil CIORTUZ (2002), LIGHT – A constraint language and compiler system for typed-unification grammars, in Matthias JARKE, Jana KOEHLER, and Gerhard LAKEMEYER, editors, *Proceedings of the 25th Annual German Conference on AI*, pp. 3–17, Springer, Berlin, doi:10.1007/3-540-45751-8_1.

Lionel CLÉMENT and Alexandra KINYON (2003), Generating parallel multilingual LFG-TAG grammars from a MetaGrammar, in Erhard W. HINRICHS and Dan ROTH, editors, *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pp. 184–191, Sapporo, doi:10.3115/1075096.1075120.

Bernard COMRIE (1976), *Aspect: An introduction to the study of verbal aspect and related problems*, Cambridge University Press, Cambridge.

Bernard COMRIE (1985), *Tense*, Cambridge University Press, Cambridge.

Elizabeth CONRAD (2021), *Tracing and reducing lexical ambiguity in automatically inferred grammars*, Master's thesis, University of Washington.

Ann COPESTAKE (2002a), Definitions of typed feature structures, in Stephan OEPEN, Dan FLICKINGER, Jun-ichi TSUJII, and Hans USZKOREIT, editors, *Collaborative Language Engineering: A Case Study in Efficient Grammar-Based Processing*, pp. 227–230, CSLI Publications, Stanford, CA.

Ann COPESTAKE (2002b), *Implementing typed feature structure grammars*, CSLI Publications, Stanford, CA.

Ann COPESTAKE (2009), Slacker semantics: Why superficiality, dependency and avoidance of commitment can be the right way to go, in Alex LASCARIDES, Claire GARDENT, and Joakim NIVRE, editors, *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 1–9, Athens, doi:10.3115/1609067.1609167.

Ann COPESTAKE, Dan FLICKINGER, Robert MALOUF, Susanne RIEHEMANN, and Ivan SAG (1995), Translation using minimal recursion semantics, in *Proceedings of the Sixth Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages*, pp. 15–32, Leuven.

Ann COPESTAKE, Dan FLICKINGER, Carl POLLARD, and Ivan A. SAG (2005), Minimal recursion semantics: An introduction, *Research on Language and Computation*, 3(4):281–332.

Greville G CORBETT (2000), *Number*, Cambridge University Press, Cambridge.

Greville G. CORBETT (2006), *Agreement*, Cambridge University Press, Cambridge.

Francisco COSTA and António BRANCO (2010), LXGram: A deep linguistic processing grammar for Portuguese, in António TEIXEIRA, Vera Lúcia Strube LIMA, Luís Caldas OLIVEIRA, and Paulo QUARESMA, editors, *Computational Processing of the Portuguese Language*, volume 6001 of *Lecture Notes in Artificial Intelligence*, pp. 86–89, Springer, Berlin, doi:10.1007/978-3-642-12320-7_11.

Benoît CRABBÉ, Denys DUCHIER, Claire GARDENT, Joseph Le ROUX, and Yannick PARMENTIER (2013), XMG: eXtensible MetaGrammar, *Computational Linguistics*, 39(3):591–629, doi:10.1162/COLI_a_00144.

Joshua CROWGEY (2012), An a priori typology of sentential negation from an HPSG perspective, in Stefan MÜLLER, editor, *Proceedings of the 19th International Conference on Head-Driven Phrase Structure Grammar*, pp. 107–122, CSLI Publications, Stanford, CA, doi:10.21248/hpsg.2012.7.

Joshua CROWGEY (2013), *The syntactic exponence of sentential negation: A model for the LinGO Grammar Matrix*, Master's thesis, University of Washington.

Joshua CROWGEY (2019), *Braiding language (by computer): Lushootseed grammar engineering*, Ph.D. thesis, University of Washington.

Berthold CRYSMANN (2012), HaG: A computational grammar of Hausa, in Michael R. MARLO, Nikki B. ADAMS, Christopher R. GREEN, Michelle MORRISON, and Tristan M. PURVIS, editors, *Selected Proceedings of the 42nd Annual Conference on African Linguistics*, pp. 321–337, Cascadilla Press, Somerville, MA.

Berthold CRYSMANN and Woodley PACKARD (2012), Towards efficient HPSG generation for German, a non-configurational language, in Martin KAY and Christian BOITET, editors, *Proceedings of the 24th International Conference on Computational Linguistics*, pp. 695–710, Mumbai, <http://www.aclweb.org/anthology/C12-1043>.

Chris CURTIS (2018a), *Valence change library for the Grammar Matrix*, Master's thesis, University of Washington.

Chris CURTIS (2018b), Valence-changing morphology via lexical rule composition, in Stefan MÜLLER and Frank RICHTER, editors, *Proceedings of the 25th International Conference on Head-Driven Phrase Structure Grammar*, pp. 36–48, CSLI Publications, Stanford, CA, doi:10.21248/hpsg.2018.3.

Michael CYSOUW (2003), *The paradigmatic structure of person marking*, Oxford University Press, Oxford.

Glauber Romling DA SILVA (2013), *Morfossintaxe da língua Paresi-Haliti (Arawak) (Morphosyntax of the Paresi-Haliti language (Arawakan))*, Ph.D. thesis, Universidade Federal do Rio de Janeiro.

Östen DAHL (1979), Typology of sentence negation, *Linguistics*, 17(1–2):79–106.

Östen DAHL (1985), *Tense and aspect systems*, Basil Blackwell, Oxford.

Matthew DAVIDSON (2002), *Studies in Southern Wakashan (Nootkan) grammar*, Ph.D. thesis, University of New York at Buffalo.

Ferdinand DE SAUSSURE (1916), *Cours de linguistique générale (Course in general linguistics)*, Payot, Lausanne–Paris.

Robert M.W. DIXON (2004), Adjective classes in typological perspective, in Robert M.W. DIXON and Alexandra Y. AIKHENVALD, editors, *Adjective Classes: A Cross-Linguistic Typology*, pp. 1–49, Oxford University Press, Oxford.

Scott DRELLISHAK (2004), *A survey of coordination strategies in the world's languages*, Master's thesis, University of Washington.

- Scott DRELLISHAK (2008), Complex Case Phenomena in the Grammar Matrix, in Stefan MÜLLER, editor, *Proceedings of the 15th International Conference on Head-Driven Phrase Structure Grammar*, pp. 67–86, CSLI Publications, Stanford, CA.
- Scott DRELLISHAK (2009), *Widespread but not universal: Improving the typological coverage of the Grammar Matrix*, Ph.D. thesis, University of Washington.
- Scott DRELLISHAK and Emily M. BENDER (2005), A coordination module for a crosslinguistic grammar resource, in Stefan MÜLLER, editor, *Proceedings of the 12th International Conference on Head-driven Phrase Structure Grammar*, pp. 108–128, CSLI Publications, Stanford, CA, doi:10.21248/hpsg.2005.6.
- Matthew S. DRYER (2013a), Expression of Pronominal Subjects, in Matthew S. DRYER and Martin HASPELMATH, editors, *The World Atlas of Language Structures Online*, Max Planck Institute for Evolutionary Anthropology, Leipzig, <https://wals.info/chapter/101>.
- Matthew S. DRYER (2013b), Negative Morphemes, in Matthew S. DRYER and Martin HASPELMATH, editors, *The World Atlas of Language Structures Online*, Max Planck Institute for Evolutionary Anthropology, Leipzig, <https://wals.info/chapter/112>.
- Matthew S. DRYER and Martin HASPELMATH, editors (2013), *The World Atlas of Language Structures Online*, Max Planck Institute for Evolutionary Anthropology, Leipzig, <https://wals.info/>.
- Guy EMERSON (2017), (Diff)List appends in TDL, <http://www.delph-in.net/2017/append.pdf>, DELPH-IN summit, Oslo, Norway.
- Guy EMERSON (2019), Wrapper types: Relational constraints without relational constraints, <http://users.sussex.ac.uk/~johnca/summit-2019/wrapper-types.pdf>, DELPH-IN summit, Cambridge, UK.
- Guy EMERSON (2021), On the Turing completeness of typed feature structure unification, <https://raw.githubusercontent.com/delph-in/docs/main/summits/2021/turing.pdf>, DELPH-IN summit, online.
- Guy EMERSON (forthcoming), Computation as subtyping: On the Turing completeness of type systems, with applications to formal grammars.
- Zhenzhen FAN (2018), *Building an HPSG Chinese grammar (Zhong)*, Ph.D. thesis, Nanyang Technological University.
- Zhenzhen FAN, Sanghoun SONG, and Francis BOND (2015), An HPSG-based shared-grammar for the Chinese languages: ZHONG [], in Emily M. BENDER, Lori LEVIN, Stefan MÜLLER, Yannick PARMENTIER, and Aarne RANTA, editors, *Proceedings of the 2015 Workshop on Grammar Engineering Across Frameworks*, pp. 17–24, Beijing, doi:10.18653/v1/W15-3303.

Yimai FANG, Haoyue ZHU, Ewa MUSZYŃSKA, Alexander KUHNLE, and Simone TEUFEL (2016), A Proposition-based abstractive summariser, in Yuji MATSUMOTO and Rashmi PRASAD, editors, *Proceedings of the 26th International Conference on Computational Linguistics*, pp. 567–578, Osaka, <https://aclanthology.org/C16-1055>.

Charles J. FILLMORE, Paul KAY, and Mary Catherine O’CONNOR (1988), Regularity and idiomaticity in grammatical constructions: The case of Let Alone, *Language*, 64(3):501–538.

Dan FLICKINGER (2000), On building a more efficient grammar by exploiting types, *Natural Language Engineering*, 6(1):15–28.

Dan FLICKINGER (2011), Accuracy v. robustness in grammar engineering, in Emily M. BENDER and Jennifer E. ARNOLD, editors, *Language from a Cognitive Perspective: Grammar, Usage and Processing*, pp. 31–50, CSLI Publications, Stanford, CA.

Dan FLICKINGER, Carl POLLARD, and Tom WASOW (2021), The evolution of HPSG, in Stefan MÜLLER, Anne ABEILLÉ, Robert D. BORSLEY, and Jean-Pierre KOENIG, editors, *Head-Driven Phrase Structure Grammar: The handbook*, 9, pp. 47–87, Language Science Press, Berlin.

Dan FLICKINGER and Jiye YU (2013), Toward more precision in correction of grammatical errors, in Julia HOCKENMAIER and Sebastian RIEDEL, editors, *Proceedings of the 17th Conference on Computational Natural Language Learning*, pp. 68–73, Sofia.

Antske FOKKENS (2010), Documentation for the Grammar Matrix word order library, https://github.com/delph-in/docs/wiki/MatrixDoc_WordOrder.

Antske FOKKENS (2011), Metagrammar engineering: Towards systematic exploration of implemented grammars, in Dekang LIN, editor, *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pp. 1066–1076, Portland, OR.

Antske FOKKENS (2014), *Enhancing empirical research for linguistically motivated precision grammars*, Ph.D. thesis, Saarland University.

Antske FOKKENS and Tania AVGUSTINOVA (2013), SlaviCLIMB: Combining expertise for Slavic grammar development using a metagrammar, in *Proceedings of High-level Methodologies for Grammar Engineering*, pp. 87–92, Orleans.

Antske FOKKENS, Tania AVGUSTINOVA, and Yi ZHANG (2012), CLIMB grammars: Three projects using metagrammar engineering, in Nicoletta CALZOLARI, Khalid CHOUKRI, Thierry DECLERCK, Mehmet Uğur DOĞAN, Bente MAEGAARD, Joseph MARIANI, Asuncion MORENO, Jan ODIJK, and Stelios PIPERIDIS, editors, *Proceedings of the 8th International Conference on Language Resources and Evaluation*, pp. 1672–1679, Istanbul.

Antske FOKKENS, Yi ZHANG, and Emily M. BENDER (2011), Spring cleaning and grammar compression: Two techniques for detection of redundancy in HPSG grammars, in Minghui Dong HELENA HONG GAO, editor, *Proceedings of the 25th Pacific Asia Conference on Language, Information and Computation*, pp. 236–244, Tokyo.

Caroline FÉRY and Manfred KRIFKA (2008), Information structure: Notional distinctions, ways of expression, in Piet VAN STERKENBURG, editor, *Unity and Diversity of Languages*, pp. 123–136, John Benjamins, Amsterdam, doi:10.1075/z.141.

Gerald GAZDAR, Ewan KLEIN, Geoffrey PULLUM, and Ivan SAG (1985), *Generalized Phrase Structure Grammar*, Harvard University Press, Cambridge, MA.

Jonathan GINZBURG and Ivan SAG (2000), *Interrogative investigations*, CSLI Publications, Stanford, CA.

Talmy GIVÓN (1994), The pragmatics of de-transitive voice: Functional and typological aspects of inversion, in Talmy GIVÓN, editor, *Voice and Inversion*, pp. 3–44, John Benjamins, Amsterdam.

Michael Wayne GOODMAN (2013), Generation of machine-readable morphological rules from human readable input, *University of Washington Working Papers in Linguistics*, 30, http://depts.washington.edu/uwwpl/vol130/goodman_2013.pdf.

Michael Wayne GOODMAN (2019), A Python library for deep linguistic resources, in Jieh HSIANG and Michael STANLEY-BAKER, editors, *2019 Pacific Neighborhood Consortium Annual Conference and Joint Meetings*, pp. 1–7, Singapore.

Thomas GRAF and Kalina KOSTYSZYN (2021), Multiple wh-movement is not special: The subregular complexity of persistent features in minimalist grammars, in *Proceedings of the Society for Computation in Linguistics*, volume 4, pp. 275–285, ScholarWorks@UMass Amherst, Amherst, MA.

Tali Arad GRESHLER, Livnat HERZIG SHEINFUX, Nurit MELNIK, and Shuly WINTNER (2015), Development of maximally reusable grammars: Parallel development of Hebrew and Arabic grammars, in Stefan MÜLLER, editor, *Proceedings of the 22nd International Conference on Head-Driven Phrase Structure Grammar*, pp. 27–40, CSLI Publications, Stanford, CA, <https://proceedings.hpsg.xyz/article/view/318>.

Michael HAEGER (2017), *An evidentiality library for the LinGO Grammar Matrix*, Master's thesis, University of Washington.

Claude HAGÈGE (2008), Towards a typology of interrogative verbs, *Linguistic Typology*, 12:1–44, doi:10.1515/LITY.2008.031.

Harald HAMMARSTRÖM, Robert FORKEL, Martin HASPELMATH, and Sebastian BANK (2021), Glottolog database 4.5, doi:10.5281/zenodo.5772642, <https://doi.org/10.5281/zenodo.5772642>.

Jorge HANKAMER and Vera LEE-SCHOENFELD (2005), What moves in German VP-fronting, https://www.researchgate.net/publication/254070408_What_Moves_in_German_VP-Fronting, handout of presentation at Berkeley Syntax/Semantics Circle.

Martin HASPELMATH, Michael MEEUWIS, APiCS CONSORTIUM, *et al.* (2013), Position of interrogative phrases in content questions, in Susanne Maria MICHAELISC, Philippe MAURER, Martin HASPELMATH, and Magnus HUBER, editors, *The Atlas of Pidgin and Creole Language Structures*, pp. 44–47, Oxford University Press, Oxford.

Martin HASPELMATH and Thomas MÜLLER-BARDEY (2001), Valence change, in G. BOOIJ, C. LEHMANN, and J. MUGDAN, editors, *A Handbook on Inflection and Word Formation*, Walter de Gruyter, Berlin.

Jeffrey HEATH (2017), *A grammar of Jalkunan (Mande, Burkina Faso)*, Language Description Heritage Library, doi:10.17617/2.2346932.

Bernd HEINE (1997), *Possession: Cognitive sources, forces, and grammaticalization*, 83, Cambridge University Press, Cambridge, doi:10.1017/CBO9780511581908.

Lars HELLAN and Petter HAUGEREID (2003), The NorSource grammar – an exercise in the Matrix grammar building design, in Emily M BENDER, Dan FLICKINGER, Frederik FOUVRY, and Melanie SIEGEL, editors, *Proceedings of Workshop on Multilingual Grammar Engineering, ESSLLI 2003*.

Andreas HÖLZL (2018), *A typology of questions in Northeast Asia and beyond: An ecological perspective*, Language Science Press, Berlin.

Kristen HOWELL (2020), *Inferring grammars from Interlinear Glossed Text: Extracting typological and lexical properties for the automatic generation of HPSG grammars*, Ph.D. thesis, University of Washington.

Kristen HOWELL and Emily M. BENDER (2022), Building analyses from syntactic inference in local languages: An HPSG grammar inference system, *Northern European Journal of Language Technology*, 8(1), doi:10.3384/nejlt.2000-1533.2022.4017, <https://nejlt.ep.liu.se/article/view/4017/3515>.

Kristen HOWELL and Olga ZAMARAEVA (2018), Clausal modifiers in the Grammar Matrix, in Emily M. BENDER, Leon DERCZYNSKI, and Pierre ISABELLE, editors, *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 2939–2952, Santa Fe, NM.

Kristen HOWELL, Olga ZAMARAEVA, and Emily M. BENDER (2018), Nominalized clauses in the Grammar Matrix, in Stefan MÜLLER and Frank RICHTER, editors, *The 25th International Conference on Head-Driven Phrase Structure Grammar*, CSLI Publications, Stanford, CA, doi:10.21248/hpsg.2018.5.

David INMAN (2019), *Multi-predicate constructions in Nuuchahnulth*, Ph.D. thesis, University of Washington.

David INMAN and Ruth MORRISON (2014), Negation in Nanti: Syntactic evidence for head and dependent negators, in Stefan MÜLLER, editor, *Proceedings of the 21st International Conference on Head-Driven Phrase Structure Grammar*, pp. 103–113, CSLI Publications, Stanford, CA, doi:10.21248/hpsg.2014.6, <https://proceedings.hpsg.xyz/article/view/306>.

Aravind K JOSHI and Yves SCHABES (1997), Tree-Adjoining Grammars, in Grzegorz ROZENBERG and Arto SALOMAA, editors, *Handbook of Formal Languages*, pp. 69–123, Springer, New York.

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.

Marina KHASANOVA and Alexander PEVNOV (2002), *Myths and tales of the Negidals*, Nakanishi, Kyoto.

Jong-Bok KIM and Jaehyung YANG (2003), Korean phrase structure grammar and its implementations into the LKB system, in Dong Hong JI and Kim Teng LUA, editors, *Proceedings of the 17th Pacific Asia Conference on Language, Information and Computation*, pp. 88–97, COLIPS Publications, Singapore.

Valia KORDONI and Julia NEU (2005), Deep analysis of Modern Greek, in Keh-Yih SU, Oi Yee KWONG, Jn'ichi TSUJII, and Jong-Hyeok LEE, editors, *Proceedings of the 2004 International Joint Conference on Natural Language Processing*, pp. 674–683, Springer, Berlin.

Hans-Ulrich KRIEGER and Ulrich SCHÄFER (1994), TDL – A type description language for HPSG, in Makoto NAGAO and Yorick WILKS, editors, *Proceedings of the 15th International Conference on Computational Linguistics*, pp. 893–899, Kyoto.

Meredith LANDMAN and Marcin MORZYCKI (2003), Event-kinds and the representation of manner, in Nancy Mae ANTRIM, Grant GOODALL, Martha SCHULTE-NAFEH, and Vida SAMIAN, editors, *Proceedings of the Western Conference in Linguistics*, volume 11, pp. 1–12, Fresno, CA.

Rob MALOUF, John CARROLL, and Ann COPESTAKE (2002), Efficient feature structure operations without compilation, in Stephan OEPEN, Dan FLICKINGER, Jun-ichi TSUJII, and Hans USZKOREIT, editors, *Collaborative Language Engineering: A Case Study in Efficient Grammar-Based Processing*, pp. 105–125, CSLI Publications, Stanford, CA.

Montserrat MARIMON (2010), The Spanish Resource Grammar, in Nicoletta CALZOLARI, Khalid CHOUKRI, Bente MAEGAARD, Joseph MARIANI, Jan

ODIJK, Stelios PIPERIDIS, Mike ROSNER, and Daniel TAPIAS, editors, *Proceedings of the 7th International Conference on Language Resources and Evaluation*, Valletta, http://www.lrec-conf.org/proceedings/lrec2010/pdf/602_Paper.pdf.

Detmar MEURERS, Gerald PENN, and Frank RICHTER (2002), A web-based instructional platform for constraint-based grammar formalisms and parsing, in Dragomir RADEV and Chris BREW, editors, *Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics*, pp. 19–26, Philadelphia, PA.

Yusuke MIYAO and Jun'ichi TSUJII (2008), Feature forest models for probabilistic HPSG parsing, *Computational Linguistics*, 34(1):35–80.

Osahito MIYAOKA (2012), *A grammar of Central Alaskan Yupik (CAY)*, Walter de Gruyter, Berlin, doi:10.1515/9783110278576.

David MOELJADI, Francis BOND, and Sanghoun SONG (2015), Building an HPSG-based Indonesian resource grammar (INDRA), in Emily M. BENDER, Lori LEVIN, Stefan MÜLLER, Yannick PARMENTIER, and Arne RANTA, editors, *Proceedings of the 2015 Workshop on Grammar Engineering Across Frameworks*, pp. 9–16, Beijing, doi:10.18653/v1/W15-3302.

Richard MONTAGUE (1974), *Formal philosophy: Selected papers of Richard Montague*, Yale University Press, New Haven, CT.

Luís MORGADO DA COSTA, Francis BOND, and Xiaoling HE (2016), Syntactic well-formedness diagnosis and error-based coaching in computer assisted language learning using machine translation, in Hsin-Hsi CHEN, Yuen-Hsien TSENG, Vincent NG, and Xiaofei LU, editors, *Proceedings of the 3rd Workshop on Natural Language Processing Techniques for Educational Applications*, pp. 107–116, Osaka.

Luis MORGADO DA COSTA, Roger WINDER, Shu Yun LI, Benedict Christopher Lin Tzer LIANG, Joseph MACKINNON, and Francis BOND (2020), Automated writing support using deep linguistic parsers, in Nicoletta CALZOLARI, Frédéric BÉCHET, Philippe BLACHE, Khalid CHOUKRI, Christopher CIERI, Thierry DECLERCK, Sara GOGGI, Hitoshi ISAHARA, Bente MAEGAARD, Joseph MARIANI, Hélène MAZO, Asuncion MORENO, Jan ODIJK, and Stelios PIPERIDIS, editors, *Proceedings of the 12th Language Resources and Evaluation Conference*, pp. 369–377, Marseille.

Sarah E. MURRAY (2017), *The semantics of evidentials*, Oxford University Press, Oxford, doi:10.1093/oso/9780199681570.001.0001.

Stefan MÜLLER (1999), *Deutsche Syntax deklarativ: Head-Driven Phrase Structure Grammar für das Deutsche (Declarative German Syntax: Head-Driven Phrase Structure Grammar for German)*, Max Niemeyer Verlag, Tübingen, doi:10.1515/9783110915990.

- Stefan MÜLLER (2007), The Grammix CD-ROM: A software collection for developing typed feature structure grammars, in Tracy Holloway KING and Emily M. BENDER, editors, *Proceedings of the 2007 Workshop on Grammar Engineering Across Frameworks*, pp. 259–266, CSLI Publications, Stanford, CA, <http://csli-publications.stanford.edu/GEAF/2007/>.
- Stefan MÜLLER (2015), The CoreGram project: Theoretical linguistics, theory development and verification, *Journal of Language Modelling*, 3(1):21–86, doi:10.15398/jlm.v3i1.91.
- Stefan MÜLLER, Anne ABEILLÉ, Robert D. BORSLEY, and Jean-Pierre KOENIG, editors (2021), *Head Driven Phrase Structure Grammar: The handbook*, Language Science Press, Berlin, doi:10.5281/zenodo.5543318.
- Elizabeth NIELSEN and Emily M. BENDER (2018), Modeling adnominal possession in multilingual grammar engineering, in Stefan MÜLLER and Frank RICHTER, editors, *Proceedings of the 25th International Conference on Head-Driven Phrase Structure Grammar*, pp. 140–153, CSLI Publications, Stanford, CA, doi:10.21248/hpsg.2018.9.
- Elizabeth K NIELSEN (2018), *Modeling adnominal possession in the linGO Grammar Matrix*, Master’s thesis, University of Washington.
- Michael NOONAN (2007), Complementation, in Timothy SHOPEN, editor, *Language Typology and Syntactic Description*, pp. 52–150, Cambridge University Press, Cambridge, 2nd edition, doi:doi.org/10.1017/CBO9780511619434.
- Stephan OEPEN (2002), *Competence and performance profiling for constraint-based grammars: A new methodology, toolkit, and applications*, Ph.D. thesis, Universität des Saarlandes.
- Stephan OEPEN, Emily M. BENDER, Ulrich CALLMEIER, Dan FLICKINGER, and Melanie SIEGEL (2002), Parallel distributed grammar engineering for practical applications, in *Proceedings of the Workshop on Grammar Engineering and Evaluation*, Taipei.
- Stephan OEPEN, Dan FLICKINGER, Kristina TOUTANOVA, and Christopher D. MANNING (2004), LinGO Redwoods: A rich and dynamic treebank for HPSG, *Research on Language and Computation*, 2(4):575–596, doi:10.1007/s11168-004-7430-4.
- Stephan OEPEN and Daniel P. FLICKINGER (1998), Towards systematic grammar profiling: Test suite technology ten years after, *Journal of Computer Speech and Language*, 12(4):411–435, doi:10.1006/csla.1998.0105.
- Stephan OEPEN, Erik VELLDAL, Jan Tore LØNNING, Paul MEURER, Victoria ROSÉN, and Dan FLICKINGER (2007), Towards hybrid quality-oriented machine translation. On linguistics and probabilities in MT, in Andy WAY and Barbara GAWRONSKA, editors, *The 11th International Conference on Theoretical and Methodological Issues in Machine Translation*, Skövde, Sweden, <https://aclanthology.org/2007.tmi-papers>.

Kelly O'HARA (2008), *A morphotactic infrastructure for a grammar customization system*, Master's thesis, University of Washington.

Petya OSENOVA (2010), *BURGER: Bulgarian Resource Grammar – Efficient and Realistic*,

<http://bultreebank.org/en/bulgarian-resource-grammar-burger/>, technical report, Stanford, CSLI.

Petya OSENOVA (2011), Localizing a core HPSG-based grammar for Bulgarian, in Hanna HEDELAND, Thomas SCHMIDT, and Kai WÖRNER, editors, *Proceedings of the Conference of the German Society for Computational Linguistics and Language Technology*, pp. 175–182, Hamburg.

Woodley PACKARD (2015), *Full forest treebanking*, Master's thesis, University of Washington.

Barbara H. PARTEE (1979), Constraining Montague grammar: A framework and a fragment, in S. DAVIS and M. MITHUN, editors, *Linguistics, Philosophy, and Montague Grammar*, pp. 51–101, University of Texas Press, Austin, TX, doi:10.7560/746251-004.

Doris L. PAYNE and Immanuel BARSHI, editors (1999), *External Possession*, John Benjamins Publishing Co., Amsterdam, doi:10.1075/tsl.39.

John R. PAYNE (1985), Complex phrases and complex sentences, in Timothy SHOPEN, editor, *Language typology and syntactic description*, volume 2, pp. 3–41, Cambridge University Press, Cambridge, 1st edition, doi:doi.org/10.1017/CBO9780511619434.

Gerald PENN (2000), Applying constraint handling rules to HPSG, in Thom FRÜHWIRTH, editor, *Proceedings of the Workshop on Rule-Based Constraint Reasoning and Programming*, <http://www.informatik.uni-ulm.de/pm/fileadmin/pm/home/fruehwirth/cl2000r.html>.

Gerald PENN (2004), Balancing clarity and efficiency in typed feature logic through delaying, in Donia SCOTT, editor, *Proceedings of the 42nd Meeting of the Association for Computational Linguistics*, pp. 239–246, Barcelona, doi:10.3115/1218955.1218986.

Carl POLLARD and Ivan A. SAG (1987), *Information-based syntax and semantics*, CSLI Publications, Stanford, CA.

Carl POLLARD and Ivan A. SAG (1994), *Head-Driven Phrase Structure Grammar*, University of Chicago Press, Chicago, IL.

Laurie POULSON (2011), Meta-modeling of tense and aspect in a cross-linguistic grammar engineering platform, in Sanghoun SONG and Joshua CROWGEY, editors, *University of Washington Working Papers in Linguistics*, volume 28, pp. 1–62.

Conor QUINN (2006), *Referential-access dependency in Penobscot*, Ph.D. thesis, Harvard University.

- Aarne RANTA (2011), *Grammatical framework: Programming with multilingual grammars*, CSLI Publications, Stanford, CA.
- Frank RICHTER (2021), Formal Background, in Stefan MÜLLER, Anne ABEILLÉ, Robert D. BORSLEY, and Jean-Pierre KOENIG, editors, *Head-Driven Phrase Structure Grammar: The handbook*, pp. 89–124, Language Science Press, Berlin, doi:DOI:10.5281/zenodo.5599822.
- Ivan A. SAG, Hans C. BOAS, and Paul KAY (2012), Introducing Sign-Based Construction Grammar, in Hans C. BOAS and Ivan A. SAG, editors, *Sign-Based Construction Grammar*, pp. 1–29, CSLI Publications, Stanford, CA.
- Safiyah SALEEM (2010), *Argument optionality: A new library for the Grammar Matrix customization system*, Master's thesis, University of Washington.
- Safiyah SALEEM and Emily M. BENDER (2010), Argument optionality in the LinGO Grammar Matrix, in Chu-Ren HUANG and Dan JURAFSKY, editors, *Proceedings of the 23rd International Conference on Computational Linguistics*, pp. 1068–1076, Beijing.
- Robert SCHIKOWSKI (2012), Chintang morphology, unpublished manuscript, University of Zürich.
- Melanie SIEGEL, Emily M. BENDER, and Francis BOND (2016), *Jacy: An implemented grammar of Japanese*, CSLI Publications, Stanford, CA, <http://web.stanford.edu/group/cslipublications/cslipublications/site/9781684000180.shtml>.
- Anna SIEWIERSKA (2004), *Person*, Cambridge University Press, Cambridge.
- Glenn C SLAYDEN (2011), Thai Grammar, available at <http://agree-grammar.com/src/grammars/thai>. Accessed 05-20-2022.
- Glenn C SLAYDEN (2012), *Array TFS storage for unification grammars*, Master's thesis, University of Washington.
- Sanghoun SONG (2014), *A grammar library for information structure*, Ph.D. thesis, University of Washington.
- Sanghoun SONG, Jong-Bok KIM, Francis BOND, and Jaehyung YANG (2010), Development of the Korean resource grammar: Towards grammar customization, in *Proceedings of the 8th Workshop on Asian Language Resources*, pp. 144–152.
- Edward STABLER (1997), Derivational minimalism, in Christian RETORÉ, editor, *Proceedings of the International Conference on Logical Aspects of Computational Linguistics*, pp. 68–95, Springer, New York.
- Leon STASSEN (1997), *Intransitive predication*, Clarendon Press, Oxford.
- Leon STASSEN (2000), AND-languages and WITH-languages, *Linguistic Typology*, 4:1–54.

Leon STASSEN (2013), Predicative adjectives, in Matthew S. DRYER and Martin HASPELMATH, editors, *The World Atlas of Language Structures Online*, Max Planck Institute for Evolutionary Anthropology, Leipzig, <http://wals.info/chapter/118>.

Mark STEEDMAN (2000), *The syntactic process*, MIT Press, Cambridge, MA.

Patrick SUPPES, Tie LIANG, Elizabeth E. MACKEN, and Dan FLICKINGER (2014), Positive technological and negative pre-test-score effects in a four-year assessment of low socioeconomic status K-8 student learning in computer-based math and language arts courses, *Computers and Education*, 71:23–32, doi:10.1016/j.compedu.2013.09.008.

Sandra A THOMPSON, Robert E LONGACRE, and Shin Ja J HWANG (1985), Adverbial clauses, in Timothy SHOPEN, editor, *Language Typology and Syntactic Description*, pp. 171–234, Cambridge University Press, Cambridge, 1st edition, doi:doi.org/10.1017/CBO9780511619434.

John TORR (2018), Constraining MGBank: Agreement, L-selection and supertagging in Minimalist Grammars, in Iryna GUREVYCH and Yusuke MIYAO, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pp. 590–600, Melbourne, doi:10.18653/v1/P18-1055, <https://aclanthology.org/P18-1055>.

Thomas TRIMBLE (2014), *Adjectives in the LinGO Grammar Matrix*, Master's thesis, University of Washington.

Hans USZKOREIT, Rolf BACKOFEN, Stephan BUSEMANN, Abdel Kader DIAGNE, Elizabeth A. HINKELMAN, Walter KASPER, Bernd KIEFER, Hans-Ulrich KRIEGER, Klaus NETTER, Gunter NEUMANN, Stephan OEPEN, and Stephen P. SPACKMAN (1994), DISCO – An HPSG-based NLP system and its application for appointment scheduling, in Makoto NAGAO and Yorik WILKS, editors, *Proceedings of the 15th International Conference on Computational Linguistics*, Kyoto, <https://aclanthology.org/C94-1072>.

Gertjan VAN NOORD (2006), At Last Parsing Is Now Operational (APLINO), in Piet MERTENS, Cédric FAIRON, Anne DISTER, and Patrick WATRIN, editors, *Actes de la 13ème Conférence sur le Traitement Automatique des Langues Naturelles*, pp. 20–42, l'Association pour Traitement Automatique des Langues, Leuven, <http://talnarchives.atala.org/TALN/TALN-2006/taln-2006-invite-002.pdf>.

David WAX (2014), *Automated grammar engineering for verbal morphology*, Master's thesis, University of Washington.

Olga ZAMARAEVA (2016), Inferring morphotactics from interlinear glossed text: Combining clustering and precision grammars, in Micha ELSNER and Sandra KUEBLER, editors, *Proceedings of the 14th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pp. 141–150, Berlin, doi:10.18653/v1/W16-2021.

Olga ZAMARAEVA (2021a), *Assembling syntax: Modeling constituent questions in a grammar engineering framework*, Ph.D. thesis, University of Washington.

Olga ZAMARAEVA (2021b), Morphological marking of constituent questions: A case for nonlocal amalgamation, in Stefan MÜLLER and Nurit MELNIK, editors, *Proceedings of the 28th International Conference on Head-Driven Phrase Structure Grammar*, pp. 241–261, Frankfurt/Main, doi:10.21248/hpsg.2021.13.

Olga ZAMARAEVA and Emily M. BENDER (2014), Focus case outside of Austronesian: An analysis of Kolyma Yukaghir, in Stefan MÜLLER, editor, *Proceedings of the 21st International Conference on Head-Driven Phrase Structure Grammar*, pp. 176–196, CSLI Publications, Stanford, CA, doi:10.21248/hpsg.2014.10, <https://proceedings.hpsg.xyz/article/view/310>.

Olga ZAMARAEVA and Guy EMERSON (2020), Multiple question fronting without relational constraints: An analysis of Russian as a basis for cross-linguistic modeling, in Stefan MÜLLER and Anke HOLLER, editors, *Proceedings of the 27th International Conference on Head-Driven Phrase Structure Grammar*, pp. 157–177, CSLI Publications, Stanford, CA, doi:10.21248/hpsg.2020.9.

Olga ZAMARAEVA, Kristen HOWELL, and Emily M. BENDER (2019a), Handling cross-cutting properties in automatic inference of lexical classes: A case study of Chintang, in *Proceedings of the 3rd Workshop on the Use of Computational Methods in the Study of Endangered Languages*, Honolulu, HI.


Olga ZAMARAEVA, Kristen HOWELL, and Emily M. BENDER (2019b), Modeling clausal complementation for a grammar engineering resource, in Gaja JAROSZ, Max NELSON, Brendan O'CONNOR, and Joe PATER, editors, *Proceedings of the 2nd Meeting of the Society for Computation in Linguistics*, pp. 39–49, ScholarWorks@UMass Amherst, Amherst, MA, doi:10.7275/dygn-c796.

Olga ZAMARAEVA, František KRATOCHVÍL, Emily M. BENDER, Fei XIA, and Kristen HOWELL (2017), Computational support for finding word classes: A case study of Abui, in Antti ARPPE, Jeff GOOD, Mans HULDEN, Jordan LACHLER, Alexis PALMER, and Lane SCHWARTZ, editors, *Proceedings of the 2nd Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pp. 130–140, Honolulu, HI, doi:10.18653/v1/W17-0118.

Arnold M. ZWICKY, Joyce FRIEDMAN, Barbara C. HALL, and Donald E. WALKER (1965), The MITRE syntactic analysis procedure for Transformational Grammars, in Robert W. RECTOR, editor, *AFIPS Conference Proceedings*, volume 27, pp. 317–326, Spartan Books, Washington, D.C., doi:10.1145/1463891.1463928.


The Grammar Matrix at 20

Olga Zamaraeva

 0000-0001-9969-058X


Facultade de Informática
Universdade da Coruña
Rúa da Maestranza 9,
15001 A Coruña, Spain

Chris Curtis

 0000-0002-3499-466X


Firemuse Research
Rockville, Maryland, USA

Guy Emerson

 0000-0002-3136-9682


University of Cambridge
Cambridge, UK

Antske Fokkens

 0000-0002-6628-6916


Vrije Universiteit & Eindhoven
University of Technology
Amsterdam & Eindhoven, The
Netherlands

Michael W. Goodman

 0000-0002-2896-5141

LivePerson Inc.
Seattle, Washington, US


Kristen Howell

 0000-0002-4564-055X

LivePerson Inc.
Seattle, Washington, US


University of Washington
Seattle, Washington, USA

T.J. Trimble

 0000-0002-1605-2177


Independent scholar

Emily M. Bender


 0000-0001-5384-6227

University of Washington
Seattle, Washington, USA

Olga Zamaraeva, Chris Curtis, Guy Emerson, Antske Fokkens, Michael Wayne Goodman, Kristen Howell, T.J. Trimble, and Emily M. Bender (2022), *20 years of the Grammar Matrix: cross-linguistic hypothesis testing of increasingly complex interactions*, *Journal of Language Modelling*, 10(1):49–137

 <https://dx.doi.org/10.15398/jlm.v10i1.292>

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

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

Implementing Natural Language Inference for comparatives

Izumi Haruta¹, Koji Mineshima², and Daisuke Bekki¹

¹ Ochanomizu University, Japan

² Keio University, Japan

ABSTRACT

This paper presents a computational framework for Natural Language Inference (NLI) using logic-based semantic representations and theorem-proving. We focus on logical inferences with comparatives and other related constructions in English, which are known for their structural complexity and difficulty in performing efficient reasoning. Using the so-called A-not-A analysis of comparatives, we implement a fully automated system to map various comparative constructions to semantic representations in typed first-order logic via Combinatory Categorical Grammar parsers and to prove entailment relations via a theorem prover. We evaluate the system on a variety of NLI benchmarks that contain challenging inferences, in comparison with other recent logic-based systems and neural NLI models.

Keywords:
comparatives,
compositional
semantics,
theorem proving,
Combinatory
Categorical
Grammar,
Natural Language
Inference

INTRODUCTION

1

Natural Language Inference (NLI), which is also called Recognizing Textual Entailment, is the task of determining whether a text entails a hypothesis. It is a method widely used for evaluating systems in Natural Language Processing (NLP). In recent years, with the development of large datasets such as Stanford Natural Language Inference (SNLI;

Bowman *et al.* 2015) and Multi-Genre Natural Language Inference (MultiNLI; Williams *et al.* 2018), it has been used as one of the methods for evaluating the performance of deep learning (DL) models.

NLI can be characterized as a *black-box* type evaluation in the sense that it does not matter what the internal structure of the evaluated system is (Bos 2008a). Thus, it does not matter whether the system to be evaluated is based on DL or on parsing and logic. In fact, the FraCaS project (Cooper *et al.* 1996), one of the origins of NLI benchmarks, was developed to evaluate a pipeline of syntax, semantics, and inference systems based on linguistic theories. The goal was to make a meaningful comparison and evaluation of various frameworks of formal syntax and semantics (cf. Morrill and Valentín 2016).

How well can current linguistic and logical theories solve NLI benchmarks including FraCaS and others that contain challenging semantic phenomena? The purpose of this paper is to address this question. The question has important implications both in the context of NLP and theoretical linguistics. In the context of NLP, a logic-based approach to NLI can provide a basis for a more explanatory and interpretable alternative to DL-based approaches. In the context of theoretical linguistics, it has the significance of systematically testing and evaluating linguistic theories using NLI benchmarks well-designed by linguists.

In this paper, we introduce a logic-based framework for NLI, focusing on comparatives and other related constructions in English, including adjectives, adverbs, numerals, and generalized quantifiers. Comparative constructions have been actively studied in formal semantics yet still pose a challenge to computational approaches (Pulman 2007). Our system has a pipeline consisting of syntactic parsing based on Combinatory Categorical Grammar (CCG; Steedman 1996, 2000), compositional mapping of parsed trees to logical forms, and theorem-proving in a First-Order Logic (FOL) setting. In this respect, the system is transparent, allowing us to examine what happens at each step of parsing (syntax), semantic analysis (semantics), and theorem proving (logic).

Each linguistic phenomenon we are concerned with in this paper has been largely tackled by a separate semantic theory, for example, event semantics for verbs, degree semantics for adjectives, and theories of generalized quantifier for noun phrases (see Section 2 for the

detail of each theory). What is needed here is to put together these different theories, to formulate the resulting system as a computational model, and to empirically evaluate its prediction. Note also that it is often the case that computational implementation of existing theories is not a trivial task but one that requires additional substantial work, to decide things for which the published papers do not specify the details. In this respect, there is a large gap between formal semantics and its computational implementation. We also emphasize the importance of a fully-automated NLI system for evaluating a linguistic theory: if you throw an inference in natural language to the system, it can immediately compute the logical forms and evaluate the entailment relation, thus facilitating to make a prediction of the theory in an easy and quick way.

Our system is designed to have a reasonable expressive power to represent various comparative constructions without compromising the efficiency of automated theorem proving. The results of the evaluation on various datasets, including FraCaS, show that our system is capable of solving complex logical reasoning with high accuracy. We also compare our system with existing logic-based systems and current state-of-the-art DL models. All code and evaluation results are publicly available.¹

Our contributions are summarized as follow:

- We propose semantic representations (logical forms) for various comparative constructions and related constructions in English, including generalized quantifiers, numerals, and adverbs, using a uniform representation language in typed FOL that is suitable for automated theorem proving (Section 2).
- We implement a compositional semantics for these constructions in the framework of CCG (Section 3).
- We evaluate our system on various NLI datasets including FraCaS that contain complex logical inferences with comparatives and other linguistic phenomena, in comparison with other logic-based systems and DL-based NLI models (Section 4).

¹<https://github.com/izumi-h/ccgcomp>

2 SEMANTIC REPRESENTATIONS

In this section, we first introduce our representation language, in comparison with other approaches (Section 2.1). Then we present the semantic representations of various gradable constructions, in particular, adjectives (Section 2.2), comparatives (Section 2.3), adverbs (Section 2.4), and generalized quantifiers (Section 2.5).

2.1 Representation language: Typed FOL

As a representation language, we use the Typed First-Order Form (TFF) of the Thousands of Problems for Theorem Provers (TPTP) format (Sutcliffe *et al.* 2012; Sutcliffe 2017). TPTP is a library of problems for automated theorem proving systems. TFF is a formal expression in FOL with equality and arithmetic operations. TFF extends the language of FOL with the notion of types. It has predefined basic types for entity (e) and truth-value (t), and arithmetic types for integers, rational numbers, and real numbers.² We use integers as the type of degrees (d), although we can instead use other arithmetic types (rational numbers or real numbers) in the implementation. In addition, we use the type of *events* (v) as a user-defined type. Thus, the semantic type τ of an expression is defined by the following rule:

$$\tau ::= e \mid t \mid v \mid d \mid \tau \rightarrow \tau$$

Here $\tau \rightarrow \tau$ is a function type, where \rightarrow is right-associative. Thus $t \rightarrow t \rightarrow t$ means $t \rightarrow (t \rightarrow t)$.

Note that although we use λ -calculus for semantic composition as will be explained in Section 3, the language of TFF does not allow the use of λ -abstraction. Thus, λ -terms can only appear in the process of a compositional derivation but not in the resulting logical form. Whether this language has a sufficient descriptive capacity is an empirical question, and we will show through evaluation by NLI benchmarks that the language is expressive enough to represent various linguistic phenomena we deal with in this paper.

²TFF uses the notations \$i for individuals and \$o for truth-values (booleans). We instead use e and t in this paper.

Other representation languages used in the logic-based approaches to NLI include (i) Higher-Order Logic (HOL), (ii) FOL, and (iii) Type Theory. Regarding (i), Mineshima *et al.* (2015) and Abzianidze (2015, 2016) propose an NLI system combining CCG parsers with provers specialized for natural languages using a controlled fragment of HOL. Although HOL is expressive enough to handle complex expressions such as generalized quantifiers, provers based on HOL are less efficient than those based on FOL and tend to rely on hand-coded rules, causing scalability issues.

For (ii), Bos (2008b) and Martínez-Gómez *et al.* (2017) present NLI systems based on standard FOL. While theorem provers based on FOL are more efficient than HOL, the expressive power is limited so that there are linguistic phenomena that resist straightforward treatment in FOL. A notable exception is Hahn and Richter (2016), which introduces a method to encode HOL constructions in natural languages in FOL Henkin Semantics. However it is not extended to complex phenomena such as comparatives covered in FraCaS. Perhaps the approach that is closest to ours is that of Pulman (2018), which presents methods to approximate some higher-order inferences with adjectives in a first-order setting. Compared with these previous works, our system has broader coverage, handling a variety of inferences with adjectives, comparatives, generalized quantifiers, numerals, and adverbs from a unified perspective.

For (iii), Chatzikyriakidis and Luo (2014), Bernardy and Chatzikyriakidis (2017) and Chatzikyriakidis and Bernardy (2019) present a type-theoretic system using Coq as a proof assistant for NLI, tackling problems in FraCaS. However they inherit the disadvantages of HOL in that the theorem proving is not computationally efficient; in fact, the theorem-proving component of these type-theoretic systems is not fully automated, due in part to the fact that there is no decision procedure for HOL. Thus, it cannot be used as part of a system that would be comparable to logic-based NLI systems studied in the context of natural language processing (NLP). By contrast, TFF, which is adopted in our approach, has computational efficiency and expressive power in that it can handle equality and arithmetic operations implemented in automated theorem provers. It is a language that suits the purpose of our study. We emphasize the importance of building a fully automated NLI system, which allows us to build a system usable in NLP

applications and to compute the predictions of each formal semantic theory quickly and precisely. This would be an initial step towards establishing a meaningful and systematic way to evaluate each linguistic framework.

2.2 *Adjectives*

We start with the analysis of adjectives in our framework. This serves as a basis for developing computational degree-based semantics for other gradable constructions.

2.2.1 Gradable adjectives

We introduce the phenomenon of GRADABILITY and present an analysis of gradable adjectives in degree-based semantics.³

- (1) My car is *expensive*. (Gradable)
a. My car is very *expensive*.
b. My car is more *expensive* than yours.
- (2) My pet is *four-legged*. (Non-gradable)
a. # My pet is very *four-legged*.
b. # My pet is more *four-legged* than yours.

Expensive and *tall* are gradable adjectives, and can take degree modifiers such as *very* and have comparative form as in (1a) and (1b). On the other hand, *four-legged* is not a gradable adjective; the sentences (2a) and (2b) are not felicitous.

In degree-based semantics, gradable adjectives can be treated as two-place predicates that take entity and degree (Cresswell 1976). For instance, *John is 5 feet tall*, containing the specific numerical expression *5 feet*, is analyzed as $\text{tall}(\text{john}, 5 \text{ feet})$, where $\text{tall}(x, \delta)$ is read as “*x is at least as tall as degree δ* ” (Klein 1991).⁴ For simplicity, we do not consider the internal structure of a measure phrase such as *5 feet* and write as $\text{tall}(\text{john}, 5)$, where 5 is treated as an integer.

³ See Lassiter (2015) and Morzycki (2016) for an overview of degree-based semantics.

⁴ For an explanation of why $\text{tall}(x, \delta)$ is not treated as “*x is exactly as tall as δ* ”, see Section 3.2.

The positive form of a gradable adjective is regarded as involving comparison to some threshold that can be inferred from the context of the utterance. We write $\theta_F(A)$ to denote the contextually specified threshold for a predicate F given a set A , which is called a COMPARISON CLASS (Klein 1980, 1982). When a comparison class is implicit, as in (3a) and (4a), we use the universal set U as a default comparison class.⁵ We often abbreviate $\theta_F(U)$ as θ_F . Thus, (3a) is represented as (3b), which means the height of Mary is more than or equal to the threshold θ_{tall} .

- (3) a. Mary is tall.
b. $\text{tall}(\text{mary}, \theta_{\text{tall}})$

We semantically distinguish the positive adjective *tall* from its antonym *short*, which we call a negative adjective. The logical form of (4a), where a negative adjective *short* appears, is (4b); we take it that (4b) means that the height of Mary is *less than* the threshold θ_{short} .⁶

- (4) a. Mary is short.
b. $\text{short}(\text{mary}, \theta_{\text{short}})$

A threshold can be explicitly constrained by an NP modified by a gradable adjective. Thus, (5a) can be interpreted as (5b) relative to an explicit comparison class, namely, the sets of animals.⁷

- (5) a. Mickey is a small animal. (FraCaS-204)
b. $\text{small}(\text{mickey}, \theta_{\text{small}}(\text{animal})) \wedge \text{animal}(\text{mickey})$

For positive gradable adjectives, if $\text{tall}(x, \delta)$ is true, then x satisfies all heights below δ . By contrast, for negative gradable adjectives,

⁵In this study, we do not consider the context-sensitivity of an implicit comparison class. See Narisawa *et al.* (2013) and Pezzelle and Fernández (2019) for work on this topic in computational linguistics.

⁶We do not claim that this analysis can fully address the subtle inferences about antonyms (cf. Lehrer and Lehrer 1982). A more detailed analysis of antonyms is left for future work.

⁷Here and henceforth, when an example appears in FraCaS dataset (Cooper *et al.* 1996), we refer to the ID of the sentence in the dataset.

if $\text{short}(x, \delta)$ is true, then x satisfies all the heights δ or above. To formalize these properties, we postulate the following axioms for each positive adjective P and negative adjective N :

$$\text{(up)} \quad \forall x \forall \delta_1 (P(x, \delta_1) \rightarrow \forall \delta_2 ((\delta_2 \leq \delta_1) \rightarrow P(x, \delta_2)))$$

$$\text{(down)} \quad \forall x \forall \delta_1 (N(x, \delta_1) \rightarrow \forall \delta_2 ((\delta_1 \leq \delta_2) \rightarrow N(x, \delta_2)))$$

2.2.3

Privative adjectives

Apart from gradable and non-gradable adjectives, *former* and *fake* are classified as **privative** adjectives (Kamp 1975). For a privative adjective Adj and a noun phrase N , the intersection of $\llbracket Adj N \rrbracket$ and $\llbracket N \rrbracket$ is empty. For example, (6) holds for the privative adjective *former* and the noun phrase *student*.⁸

$$(6) \quad \llbracket \text{former student} \rrbracket \cap \llbracket \text{student} \rrbracket = \emptyset$$

(6) can be expressed as an axiom in our system using a predicate variable F in the following way:

$$(7) \quad \forall x (\text{former}(F(x)) \rightarrow \neg F(x))$$

For instance, (8a) is mapped to (8b). By using (7), (8a) contradicts *Peter is a student*.

- (8) a. Peter is a former student.
 b. $\text{former}(\text{student}(\text{peter}))$

2.3

Adjectival comparatives

Next, we consider adjectival comparatives using the analysis of gradable adjectives described in the previous sections.

⁸The truth condition of *former* may involve temporal semantics, which we neglect in order to avoid complicating the whole system.

To begin with, we introduce the so-called A-not-A analysis (Seuren 1973; Klein 1980, 1982, 1991; Schwarzschild 2008) for comparatives in degree-based semantics.⁹

- (9) a. Ann is taller than Bob is.
 b. $\exists \delta (\text{tall}(\text{ann}, \delta) \wedge \neg \text{tall}(\text{bob}, \delta))$



According to this analysis, (9a) is analyzed as (9b), where (9a) is interpreted as saying that there exists a degree δ of height that Ann satisfies but Bob does not. As shown in the figure in (9), together with the Consistency Postulate (CP) explained below, this guarantees that Ann’s height is greater than Bob’s height. More generally, if an adjective F is associated with a degree such as heights and weights, we can say “A is more F than B is” is true if and only if there exists a threshold δ that A satisfies but B does not. A-not-A analysis makes it possible to derive entailment relations between various comparative constructions in a simple way using FOL theorem provers.¹⁰

We show the logical forms for other basic comparative constructions under A-not-A analysis.

- (10) a. Tom is taller than Mary. (Increasing)
 b. $\exists \delta (\text{tall}(\text{tom}, \delta) \wedge \neg \text{tall}(\text{mary}, \delta))$
- (11) a. Harry is less tall than Ken. (Decreasing)
 b. $\exists \delta (\neg \text{tall}(\text{harry}, \delta) \wedge \text{tall}(\text{ken}, \delta))$
- (12) a. Tom is as tall as Mary. (Equatives)
 b. $\forall \delta (\text{tall}(\text{mary}, \delta) \rightarrow \text{tall}(\text{tom}, \delta))$

The sentence (11a) is a construction representing that the height of Harry is less than that of Ken. The sentence (12a) is interpreted as

⁹A version of this analysis is called *delineation analysis*, which goes back to Lewis (1972).

¹⁰This possibility is also suggested by Pulman (2007).

“Tom is *at least* as tall as Mary”, which means the height of Tom is greater than or equal to that of Mary. This reading is captured by mapping (12a) to (12b). The sentence (12a) can also be interpreted as “Tom is *exactly* as tall as Mary”. See Section 3.2 for a discussion on how to derive this strong reading in our setting.

In A-not-A analysis, there is an axiom called Consistency Postulate (CP), which formalizes the relation between the degrees of two entities under A-not-A analysis (Klein 1980, 1982). It asserts that if there is a degree satisfied by x but not by y , then every degree satisfied by y is satisfied by x as well.

(CP) $\forall x \forall y (\exists \delta (A(x, \delta) \wedge \neg A(y, \delta)) \rightarrow \forall \delta (A(y, \delta) \rightarrow A(x, \delta)))$,
 where A is an arbitrary gradable adjective.

The axiom (CP) can be deduced as a derivable rule of (up) and (down):

PROPOSITION 1 (CP) follows from (up) and (down).

PROOF Consider the case where A is a positive adjective. Suppose there exists δ_0 such that $A(x, \delta_0)$ holds but $A(y, \delta_0)$ does not. Also let δ be arbitrary and suppose $A(y, \delta)$. To show $A(x, \delta)$, let us assume $\delta_0 < \delta$ for the sake of contradiction. By (up) and $A(y, \delta)$, we have $A(y, \delta_0)$, but this is the contradiction. Hence, $\delta \leq \delta_0$ holds, and by (up) we have $A(x, \delta)$. Thus, $A(y, \delta) \rightarrow A(x, \delta)$ holds for any δ . When A is a negative adjective, by using (down) instead of (up) we get the same conclusion. Hence we obtain (CP). □

2.3.2

Measure phrases and differential comparatives

The sentence (13a) contains the measure phrase *2 inches* before the comparative form *taller* of the gradable adjective *tall* and mentions the difference in height between Ken and Harry. Such constructions are known as DIFFERENTIAL COMPARATIVES. (13a) means the height of Ken is *2 inches or greater* than the height of Harry. Thus differential comparatives can be handled by extending the analysis of equatives such as the sentence (12a). (13a) is mapped to the logical form (13b).

- (13) a. Ken is 2 inches taller than Harry.
 b. $\forall \delta (\text{tall}(\text{harry}, \delta) \rightarrow \text{tall}(\text{ken}, \delta + 2))$

Note that if (13a) is mapped to $\exists\delta(\text{tall}(\text{ken}, \delta + 2) \wedge \neg\text{tall}(\text{harry}, \delta))$, then the meaning that the difference in height between Ken and Harry is exactly 2 inches is missing.

To derive inferences with measure phrases, we define the axioms (sup) and (inf) that formalize supremum and infimum on degree, respectively.

(sup) $\forall x\exists\delta_1(P(x, \delta_1) \wedge \neg\exists\delta_2((\delta_1 < \delta_2) \wedge P(x, \delta_2)))$

(inf) $\forall x\exists\delta_1(N(x, \delta_1) \wedge \neg\exists\delta_2((\delta_2 < \delta_1) \wedge N(x, \delta_2)))$

The import of (sup) is expressed as follows. Assume we are given some assignment of values to variable x and P . Then there is a value δ_1 that makes $P(x, \delta_1)$ true, but there is no value δ_2 that is more than δ_1 and makes $P(x, \delta_2)$ true. Thus, the inference from (13a) to *Ken is taller than Harry* follows from (sup).

PROPOSITION 2 From $\forall\delta(\text{tall}(\text{harry}, \delta) \rightarrow \text{tall}(\text{ken}, \delta + 2))$, it follows that $\exists\delta(\text{tall}(\text{ken}, \delta) \wedge \neg\text{tall}(\text{harry}, \delta))$.

PROOF By (sup), there exists δ_0 such that $\text{tall}(\text{harry}, \delta_0)$ and there is no δ_1 such that $\delta_0 < \delta_1$ and $\text{tall}(\text{harry}, \delta_1)$. Since $\delta_0 < \delta_0 + 2$, it follows that $\neg\text{tall}(\text{harry}, \delta_0 + 2)$. By the premise, we have $\text{tall}(\text{ken}, \delta_0 + 2)$. Hence, we have $\exists\delta(\text{tall}(\text{ken}, \delta) \wedge \neg\text{tall}(\text{harry}, \delta))$. \square

Finally, consider the construction with a measure phrase in a *than*-clause. The sentence (14a) includes the measure phrase *4 feet* in the *than*-clause. It has the same meaning as “Ken is more than 4 feet tall” and is mapped to (14b). Here, instead of comparing the degree of two entities, we compare the height of Ken with the specific value *4 feet*.

- (14) a. Ken is taller than 4 feet.
 b. $\exists\delta(\text{tall}(\text{ken}, \delta) \wedge (4 < \delta))$

Extensional and intensional comparison classes

2.3.3

Gradable expressions can be divided into extensional and intensional adjectives (Kamp 1975; Partee 2007):

- (15) All dogs are animals.
 a. \Rightarrow All *fat* dogs are *fat* animals. (Extensional)
 b. \nRightarrow All *clever* dogs are *clever* animals. (Intensional)

Fat and *tall* are **extensional** adjectives and license the inference in (15a). In contrast, *clever* and *skillful* are **intensional** adjectives, which do not allow the same pattern of inference. Thus, (15b) does not hold.

The difference between extensional and intensional adjectives also arises in reasoning with comparative expressions. Consider the following:

- (16) a. John is a fatter politician than Bill.
 \Rightarrow John is fatter than Bill. (FraCaS-216)
- b. John is a cleverer politician than Bill.
 $\not\Rightarrow$ John is cleverer than Bill. (FraCaS-217)

The sentences in (16a) involve the comparative form *fatter* of the extensional adjective *fat*. The adjective *fat* is classified as an extensional adjective since *fat as a politician* does not make sense.¹¹ Accordingly, *John is a fatter politician than Bill* can be decomposed into *John is a politician and fatter than Bill*. Thus the inference in (16a) holds. On the other hand, the inference (16b), which contains the comparative form *cleverer* of the intensional adjective *clever*, does not hold. This is because even if John is cleverer than Bill as a politician, we do not know the relation between John and Bill with respect to the cleverness in other domains. For extensional adjectives, the sentence (17a) is mapped to the logical form (17b).

- (17) a. John is a fatter politician than Bill.
 b. $\text{politician}(\text{john}) \wedge \text{politician}(\text{bill})$
 $\wedge \exists \delta (\text{fat}(\text{john}, \delta) \wedge \neg \text{fat}(\text{bill}, \delta))$
- (18) a. John is fatter than Bill.
 b. $\exists \delta (\text{fat}(\text{john}, \delta) \wedge \neg \text{fat}(\text{bill}, \delta))$

For intensional adjectives $\text{clever}(x, \delta)$, we extend its second argument to take an intensional comparison class; in the second argument of the intensional adjectives we use a two-place function for a *noun parameter* $\lambda N \delta. \text{np}(N, \delta)$.¹² The type of $\text{np}(N, \delta)$ is degree. For

¹¹ Note that it is meaningful to say *fat for a politician*, so the adjective *fat* can take a comparison class and is context-sensitive (cf. Partee 2007).

¹² Throughout the paper, we abbreviate $\lambda X_1 \lambda X_2 \dots \lambda X_n. M$ as $\lambda X_1 X_2 \dots X_n. M$.

instance, $\text{clever}(x, \text{np}(\text{politician}, \delta))$ is intended to mean that x is clever as a politician (at least) to degree δ . The sentence (19a) is mapped to the logical form (19b). (19a) means that John is cleverer than Bill as a politician, and thus it does not entail (20a), which means that John is cleverer than Bill for any extension U .

- (19) a. John is a cleverer politician than Bill.
 b. $\text{politician}(\text{john}) \wedge \text{politician}(\text{bill})$
 $\wedge \exists \delta (\text{clever}(\text{john}, \text{np}(\text{politician}, \delta))$
 $\wedge \neg \text{clever}(\text{bill}, \text{np}(\text{politician}, \delta)))$
- (20) a. John is cleverer than Bill.
 b. $\exists \delta (\text{clever}(\text{john}, \text{np}(U, \delta)) \wedge \neg \text{clever}(\text{bill}, \text{np}(U, \delta)))$

Degree modifiers

2.3.4

Consider the case where an adjective appears with degree modifiers such as *very* and *much*. The following two sentences (21a) and (22a) are examples:

- (21) a. Peter is fat.
 b. $\text{fat}(\text{peter}, \theta_{\text{fat}})$
- (22) a. Peter is *very* fat.
 b. $\exists \delta (\text{fat}(\text{peter}, \delta) \wedge (\theta_{\text{fat}} + \delta' \leq \delta))$

The sentence (21a) is represented as (21b), which means that Peter meets the threshold θ_{fat} . In (22a), the degree modifier *very* appears preceding the adjective, which emphasizes the degree that Peter is fat. In this case, we set the lower bound on Peter's weight as $\theta_{\text{fat}} + \delta'$ for a constant δ' such that $0 < \delta'$ and map (22a) to (22b).

As mentioned in Section 2.2.2, we consider not only positive gradable adjectives such as *fat* but also negative gradable adjectives such as *small*. (23a) is interpreted as (23b), where the size of the room satisfies a value less than the threshold θ_{small} . The sentence (24a) emphasizes the small size of this room. In this case, we interpret the size that the room satisfies as being less than $\theta_{\text{small}} - \delta'$, and express it as (24b).

- (23) a. This room is small.
 b. $\exists x (\text{room}(x) \wedge \text{small}(x, \theta_{\text{small}}))$

- (24) a. This room is *very* small.
 b. $\exists x(\text{room}(x) \wedge \exists \delta(\text{small}(x, \delta) \wedge (\delta \leq \theta_{\text{small}} - \delta')))$

A sentence with the degree modifier *much* such as (25a) is interpreted as having a difference of at least a fixed value δ' between the degrees satisfied by the two entities being compared. It is represented as (25b) in a similar way to the analysis of (13).

- (25) a. David is *much* taller than Jim.
 b. $\forall \delta(\text{tall}(\text{jim}, \delta) \rightarrow \text{tall}(\text{david}, \delta + \delta'))$

2.4

Adverbial comparatives

In the previous sections, we analyzed comparative expressions of adjectives using a theory based on degree-based semantics, which was developed for analyzing adjectives and comparatives. In formal semantics, there is another semantic framework, event semantics, used largely to account for the semantics of verb phrases and adverbial modifiers (Davidson 1967; Parsons 1990). To address comparative expressions of adverbs, it is necessary to present a theory that incorporates not only degree semantics but also event semantics. Building on the work in Haruta *et al.* (2020), we combine the two semantic theories and extend the theory of A-not-A analysis with comparative constructions of adverbs.

2.4.1

Adverbs in event semantics

To handle adverbial expressions, we adopt a standard neo-Davidsonian event semantics (Parsons 1990), which analyzes sentences as involving quantification over events. For example, the sentence (26a) is analyzed as (26b), where *subj* is a function term that associates an event to its subject.

- (26) a. John ran.
 b. $\exists e(\text{run}(e) \wedge (\text{subj}(e) = \text{john}))$

A sentence containing an adverb like (27a) is analyzed as (27b), where the adverb *slowly* acts as a predicate of an event.

- (27) a. John ran *slowly*.
 b. $\exists e(\text{run}(e) \wedge (\text{subj}(e) = \text{john}) \wedge \text{slowly}(e))$

This allows us to derive an inference from (27a) to (26a), i.e., an inference to drop adverbial phrases.¹³

Combining event semantics and degree semantics

2.4.2

To correctly derive entailment relations between sentences with gradable adverbials and comparative expressions of adverbs, we apply the same analysis to gradable adverbials such as *slowly* and *fast* as to gradable adjectives. The following examples show logical forms of basic constructions, where adverbs like *loudly* are treated as binary predicates of an event and a degree:

- (28) a. John shouted *loudly*. (Positive)
 b. $\exists e(\text{shout}(e) \wedge (\text{subj}(e) = \text{john}) \wedge \text{loud}(e, \theta_{\text{loud}}))$
- (29) a. Jim sang *better than* Mary. (Comparative)
 b. $\exists e_1 \exists e_2(\text{sing}(e_1) \wedge (\text{subj}(e_1) = \text{jim}) \wedge \text{sing}(e_2) \wedge (\text{subj}(e_2) = \text{mary}) \wedge \exists \delta(\text{good}(e_1, \delta) \wedge \neg \text{good}(e_2, \delta)))$
- (30) a. Bob drove *as carefully as* John. (Equative)
 b. $\exists e_1 \exists e_2(\text{drive}(e_1) \wedge (\text{subj}(e_1) = \text{bob}) \wedge \text{drive}(e_2) \wedge (\text{subj}(e_2) = \text{john}) \wedge \forall \delta(\text{careful}(e_2, \delta) \rightarrow \text{careful}(e_1, \delta)))$

The sentence (28a) contains the adverbial phrase *loudly*, which is analyzed as $\text{loud}(e, \theta_{\text{loud}})$ as in (28b). This means that John's shouting is at least as loud as a certain threshold θ_{loud} , which we take to be the same logical form as the positive form of gradable adjectives. To treat predicates for adverbs in the same way as those for adjectives, we convert a gradable adverb (e.g., *loudly*) to its adjectival form (e.g., *loud*) in the logical form. The sentence (29a) is the adverbial comparative construction with the comparative form *better*. The logical form (29b) means there exists a degree of "goodness" δ such that event e_1 satisfies, but e_2 does not. Similarly, we can assign an appropriate logical form to the sentence (30a) by extending the analyses for adjectival comparatives as described in Section 2.3.

¹³In this study, we do not introduce event variables to adjectives and adverbs themselves. For instance, *Tim is tall* is analyzed as $\text{tall}(\text{tim}, \theta_{\text{tall}})$ not as $\exists e(\text{tall}(e, \theta_{\text{tall}}) \wedge (\text{subj}(e) = \text{tim}))$, where e quantifiers over underlying *states* denoted by *tall*. We do not pursue this alternative analysis here; see Parsons (1990, Chap.10) for some discussion.

2.5

Generalized quantifiers

We extend the analysis of comparatives by the degree semantics described above to generalized quantifiers. In the traditional analysis (Barwise and Cooper 1981), generalized quantifiers such as *many*, *few*, *more than*, and *most* are analyzed as denoting a relation between sets. Alternatively, these quantifiers can be analyzed as adjectives in degree semantics (Partee 1988; Rett 2018) and the proportional quantifier *most* as the superlative form of *many* (Hackl 2000; Szabolcsi 2010). We implement this alternative analysis in our computational framework.

2.5.1

Numerical adjectives

We represent a numerical adjective such as *ten* in *ten orders* by the predicate $\text{many}(x, n)$, which means that the cardinality of x is at least n , where x ranges over pluralities and n is a positive integer (Hackl 2000). The following shows the logical forms of some typical sentences involving numerical adjectives.

- (31) a. Ann won ten orders.
 b. $\exists x(\text{order}(x) \wedge \text{many}(x, 10) \wedge \exists e(\text{win}(e) \wedge (\text{subj}(e) = \text{ann}) \wedge (\text{obj}(e) = x)))$
- (32) a. Ann won many orders.
 b. $\exists \delta \exists x(\text{order}(x) \wedge \text{many}(x, \delta) \wedge (\theta_{\text{many}}(\text{order}) < \delta) \wedge \exists e(\text{win}(e) \wedge (\text{subj}(e) = \text{ann}) \wedge (\text{obj}(e) = x)))$
- (33) a. Ann won more orders than Harry.
 b. $\exists \delta (\exists x(\text{order}(x) \wedge \text{many}(x, \delta) \wedge \exists e(\text{win}(e) \wedge (\text{subj}(e) = \text{ann}) \wedge (\text{obj}(e) = x))) \wedge \neg \exists y(\text{order}(y) \wedge \text{many}(y, \delta) \wedge \exists e(\text{win}(e) \wedge (\text{subj}(e) = \text{harry}) \wedge (\text{obj}(e) = y))))$

As mentioned in the previous section, a sentence like *John is 5 feet tall* is mapped to the logical form $\text{tall}(\text{john}, 5)$ using the binary predicate of the adjective *tall*. In a similar vein, the sentence (31a) is mapped to the logical form (31b), taking the adjective *many* to be hidden between *ten* and *orders* (see Section 3.2 for a compositional derivation). In the case of (32a), we take *many* as the positive form of the adjective and introduce the threshold $\theta_{\text{many}}(\text{order})$ in the logical form (32b). In the

Table 1: Logical forms of some constructions with numerical adjectives

Sentence	Logical form
Mary won at least eleven orders.	$\exists x(\text{order}(x) \wedge \text{many}(x, 11))$ $\wedge \exists e(\text{win}(e) \wedge (\text{subj}(e) = \text{mary}) \wedge (\text{obj}(e) = x))$
Mary sold 20 more books than John.	$\forall \delta(\exists x(\text{book}(x) \wedge \text{many}(x, \delta))$ $\wedge \exists e(\text{sell}(e) \wedge (\text{subj}(e) = \text{john}) \wedge (\text{obj}(e) = x)))$ $\rightarrow \exists x(\text{book}(x) \wedge \text{many}(x, \delta + 20))$ $\wedge \exists e(\text{sell}(e) \wedge (\text{subj}(e) = \text{mary}) \wedge (\text{obj}(e) = x))$
John won twice as many orders than Ann.	$\forall \delta(\exists x(\text{order}(x) \wedge \text{many}(x, \delta))$ $\wedge \exists e(\text{win}(e) \wedge (\text{subj}(e) = \text{john}) \wedge (\text{obj}(e) = x)))$ $\rightarrow \exists x(\text{order}(x) \wedge \text{many}(x, \delta \times 2))$ $\wedge \exists e(\text{win}(e) \wedge (\text{subj}(e) = \text{ann}) \wedge (\text{obj}(e) = x))$
Bob won more orders than Luis lost.	$\exists \delta(\exists x(\text{order}(x) \wedge \text{many}(x, \delta))$ $\wedge \exists e(\text{win}(e) \wedge (\text{subj}(e) = \text{bob}) \wedge (\text{obj}(e) = x)))$ $\wedge \neg \exists x(\text{order}(x) \wedge \text{many}(x, \delta))$ $\wedge \exists e(\text{lost}(e) \wedge (\text{subj}(e) = \text{luis}) \wedge (\text{obj}(e) = x))$
More than five campers caught a cold.	$\exists x \exists \delta(\text{camper}(x) \wedge \text{many}(x, \delta) \wedge (\delta > 5))$ $\wedge \exists y(\text{cold}(y) \wedge \exists e(\text{catch}(e) \wedge (\text{subj}(e) = x)$ $\wedge (\text{obj}(e) = y)))$

case of (33a), *more* is analyzed as the comparative form of *many*; the logical form (33b) says that there exists a positive integer δ such that Ann won (at least) δ -many orders but Harry did not. Table 1 shows some more examples of logical forms of constructions with numerical adjectives.

Comparative quantificational determiners

2.5.2

We also use the predicate $\text{many}(x, n)$ to analyze proportional quantifiers such as *most* and *at least half of*. For example, the sentence (34a) is analyzed as meaning “More than half of *A* is *B*”, following the standard truth-condition (Barwise and Cooper 1981), and can be represented as (34b). The logical form in (34b) implies that there are more red apples than non-red apples. The sentence (35a) with *at most half of* is ana-

lyzed as meaning “Less than or equal to half of A is B ”, and is mapped to the logical form with the negation in (35b).¹⁴

- (34) a. *Most* apples are red.
 b. $\exists\delta(\exists x(\text{apple}(x) \wedge \text{red}(x) \wedge \text{many}(x, \delta)) \wedge \neg\exists x(\text{apple}(x) \wedge \neg\text{red}(x) \wedge \text{many}(x, \delta)))$
- (35) a. *At most half of* apples are red.
 b. $\neg\exists\delta(\exists x(\text{apple}(x) \wedge \text{red}(x) \wedge \text{many}(x, \delta)) \wedge \neg\exists x(\text{apple}(x) \wedge \neg\text{red}(x) \wedge \text{many}(x, \delta)))$

This analysis correctly captures the monotonicity property of *most*, according to which *most* is right-upward monotone;¹⁵ thus (34a) entails *Most apples are red or green*. Likewise, *at most half of* in (35a) is right-downward monotone, which is captured in the logical form (35b). Similarly, the sentence (36a) can be analyzed as meaning “More than or equal to half of A is B ” and is represented as (36b). The sentence (37a) with *less than half of* is mapped to (37b). Since *less than half of* is also a downward quantifier, we give it the logical form with negation.

- (36) a. *At least half of* apples are red.
 b. $\forall\delta(\exists x(\text{apple}(x) \wedge \neg\text{red}(x) \wedge \text{many}(x, \delta)) \rightarrow \exists x(\text{apple}(x) \wedge \text{red}(x) \wedge \text{many}(x, \delta)))$
- (37) a. *Less than half of* apples are red.
 b. $\neg\forall\delta(\exists x(\text{apple}(x) \wedge \neg\text{red}(x) \wedge \text{many}(x, \delta)) \rightarrow \exists x(\text{apple}(x) \wedge \text{red}(x) \wedge \text{many}(x, \delta)))$

¹⁴Since we assume each variable can stand for pluralities, $\text{red}(x)$ should be interpreted as distributive, meaning that each atomic part of x satisfies the predicate *red* (Link 1983). Similarly, $\neg\text{red}(x)$ should be interpreted as meaning that each atomic part of x does not satisfy *red*, where the negation is treated as a predicate modifier. However, it is beyond the scope of this paper to implement the distinction between collective and distributive predication, so we leave a full treatment of the semantics of pluralities to future work.

¹⁵Let Q be a quantifier and A and B be its restrictor and nuclear scope, respectively. The quantifier Q is *right-upward monotone* if $Q(A, B)$ and $B \subseteq C$ entail $Q(A, C)$; Q is *right-downward monotone* if $Q(A, B)$ and $C \subseteq B$ entail $Q(A, C)$. For the classification of generalized quantifiers and monotonicity properties, see e.g., Barwise and Cooper (1981) and Westerstaahl (2007).

ID	Premises and hypothesis	Gold label
253	<i>P</i> : At most half of the students take the class. <i>H</i> : Less than half of the students take the class.	Unknown
254	<i>P</i> : Most students take the class. <i>H</i> : None of the students take the class.	No
255	<i>P</i> : Less than half of the students take the class. <i>H</i> : Most students take the class.	No
256	<i>P</i> : More than half of the students take the class. <i>H</i> : Most students take the class.	Yes
257	<i>P</i> : Most students take the class. <i>H</i> : At least half of the students take the class.	Yes

Table 2:
Examples
of entailment
problems
for generalized
quantifiers
from CAD

The above analysis shows that monotonicity inferences with proportional quantifiers can be handled in typed FOL with arithmetic by assigning logical forms based on A-not-A analysis. Table 2 shows some examples of entailment relations with sentences containing the expressions described above. These are extracted from CAD dataset we will use for evaluation (see Section 4.2).

Comparatives and quantifiers

2.5.3

When determiners such as *all* or *some* appear in *than*-clauses, we need to consider the scope of the corresponding quantifiers (Larson 1988). As examples, (38a) and (39a) are assigned the logical forms in (38b) and (39b), respectively.

- (38) a. Mary is taller than every student.
b. $\forall y(\text{student}(y) \rightarrow \exists \delta(\text{tall}(\text{mary}, \delta) \wedge \neg \text{tall}(y, \delta)))$
- (39) a. Mary is taller than some student.
b. $\exists y(\text{student}(y) \wedge \exists \delta(\text{tall}(\text{mary}, \delta) \wedge \neg \text{tall}(y, \delta)))$

Conjunction (*and*) and disjunction (*or*) appearing in a *than*-clause show different behaviors in scope taking, as pointed out in Larson (1988). For instance, in (40a), the conjunction *and* takes wide scope over the main clause, whereas in (41a), the disjunction *or* can take

narrow scope. Thus, we can infer *Mary is taller than Harry* from both (40a) and (41a). These readings are represented as in (40b) and (41b), respectively.

- (40) a. Mary is taller than Harry and Bob.
 b. $\exists \delta(\text{tall}(\text{mary}, \delta) \wedge \neg \text{tall}(\text{harry}, \delta))$
 $\wedge \exists \delta(\text{tall}(\text{mary}, \delta) \wedge \neg \text{tall}(\text{bob}, \delta))$
- (41) a. Mary is taller than Harry or Bob.
 b. $\exists \delta(\text{tall}(\text{mary}, \delta) \wedge \neg(\text{tall}(\text{harry}, \delta) \vee \text{tall}(\text{bob}, \delta)))$

The quantifiers in the *than*-clause as in the sentences (38a), (39a), and (40a) need to take wide scope, while that in (41a) needs to take narrow scope. To derive this kind of scope ambiguity is not the focus of the current study and remains unsolved in our implementation. We use a fixed scope relation for quantifiers in *than*-clauses and take the wide scope reading as in (38a), (39a), and (40a) as a default reading.

3

COMPOSITIONAL SEMANTICS

In this section, we present an overview of compositional semantics that maps various comparative constructions in English to logical forms. We use CCG as a syntactic framework, a lexicalized grammar formalism that provides a transparent syntax-semantics interface (Steedman 1996, 2000). To implement a fully automated system, we use off-the-shelf CCG parsers (Clark and Curran 2007; Lewis and Steedman 2014; Yoshikawa et al. 2017), which are based on English CCGBank (Hockenmaier and Steedman 2007). Though it has been pointed out that there is room to improve English CCGBank with respect to the analysis of comparative constructions (Honnibal et al. 2010), it provides a reasonably fine-grained and rich syntactic structure that derives the type of logical forms suitable for our purposes, as we will show below. A point of using existing resources such as CCGBank is to make explicit what can be done in currently available treebanks and parsers. This would make clear the potentials and limitations of the current English CCGBank, thereby contributing to the acceleration of the study of computational semantics based on treebanks.

Table 3: Lexical entries for basic categories

Category	Logical form	Example
N	ann	<i>Ann</i>
N	$\lambda x. \text{boy}(x)$	<i>boy</i>
NP/N	$\lambda F G. \exists x. (F(x) \wedge G(x))$	<i>a</i>
NP/N	$\lambda F G. \forall x. (F(x) \rightarrow G(x))$	<i>every</i>
$S \backslash NP$	$\lambda Q. Q(\lambda x. \exists e. (\text{run}(e) = x))$	<i>run</i>
$S \backslash NP / NP$	$\lambda Q_1 Q_2. Q_1(\lambda y. Q_2(\lambda x. \exists e. \text{love}(e) \wedge (\text{subj}(e) = x) \wedge (\text{obj}(e) = y)))$	<i>love</i>

CCG-style Compositional semantics for comparatives

3.1

In CCG-style compositional semantics, the mapping from syntax to semantics is defined by assigning a syntactic category to each word. The logical form of a sentence is then compositionally derived using the standard λ -calculus. In CCGBank, major basic (ground) syntactic categories consist of N (noun), NP (noun phrase), and S (sentence). Functional categories are of the form $X \backslash Y$ and X / Y , which derives an expression of category X when combined with an expression of category Y to its left and right, respectively. Thus, category $S \backslash NP$ expects an expression of category NP to its left and produces an expression of category S , which plays the role of intransitive verbs. Similarly, $S \backslash NP / NP$ is a category for a transitive verb.¹⁶

There is a correspondence between syntactic categories and semantic types: if E_1 and E_2 are expressions assigned the same category, then the semantic types of E_1 and E_2 necessarily become the same. Table 3 shows a list of major lexical entries with semantic representations.¹⁷

To see how to derive a logical form from a CCG parsing tree based on English CCGBank, let us start with a simple example:

(42) Ann saw a boy.

¹⁶ \backslash and $/$ are left-associative; $S \backslash NP / NP$ means $(S \backslash NP) / NP$.

¹⁷ In CCGBank, a proper noun such as *Ann* is assigned the category N and shifted to NP by the unary rule *lex*, to which we assign the semantics $N : \text{ann} \Rightarrow NP : \lambda F.F(\text{ann})$.

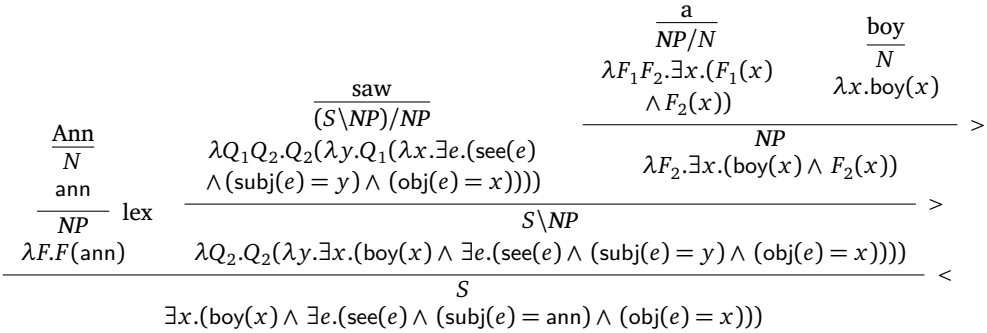


Figure 1: Parsing tree of *Ann saw a boy*

The parsing tree with logical forms looks as in Figure 1.¹⁸ Here to accommodate our compositional semantics to English CCGBank, it is convenient to use Argument Raising (Hendriks 1993), which assigns a λ -term of the quantifier type $(e \rightarrow t) \rightarrow t$ to an expression of category *NP*. Thus a transitive verb is assigned a lambda term of type $((e \rightarrow t) \rightarrow t) \rightarrow ((e \rightarrow t) \rightarrow t) \rightarrow t$.

Given this background, let us see how to derive a suitable logical form to adjectival and comparative constructions. Here are three basic constructions with their logical form under our A-not-A analysis.

- (43) a. Ann is tall. $\text{tall}(\text{ann}, \theta_{\text{tall}})$
 b. Ann is taller than Bob. $\exists \delta (\text{tall}(\text{ann}, \delta) \wedge \neg \text{tall}(\text{bob}, \delta))$
 c. Ann is as tall as Bob. $\forall \delta (\text{tall}(\text{bob}, \delta) \rightarrow \text{tall}(\text{ann}, \delta))$

To derive these logical forms compositionally, there are two main questions to be addressed: (i) which constituent introduces a degree variable and (ii) how to “saturate” the degree variables in terms of a threshold value as in (43a), existential closure as in (43b), or universal quantification as in (43c). For (i), we take it that adjectives themselves

¹⁸The variable convention for major semantic types we adopt throughout the paper is as follows. Each variable can be attached subscripts like x_1, x_2 .

Variable	Type	Description
x, y, z	e	entities
δ	d	degrees
F, G	$e \rightarrow t$	predicates
Q	$(e \rightarrow t) \rightarrow t$	quantifiers

introduce degree variable.¹⁹ Thus, under the argument raising analysis we adopt, the basic semantic representation for the adjective *tall* is $\lambda Q\delta.Q(\lambda x.tall(x, \delta))$, though a more complicated form will be needed as explained below. For (ii), we introduce an empty category into the adjunct position (i.e., a position where a measure phrase appears as in *4 feet tall*), to control the compositional derivations of the three types of logical forms.²⁰ Since English CCGBank does not support this type of empty categories, we insert them in the post-processing process of syntactic parsing. That is, we rewrite each tree in the following way.

- Empty category *pos* for positive form

$$\frac{\frac{\text{is}}{(S \backslash NP) / (S_{adj} \backslash NP)} \quad \frac{\text{tall}}{S_{adj} \backslash NP}}{S \backslash NP} > \quad \dashrightarrow \quad \frac{\text{pos}}{(S_{adj} \backslash NP) / (S_{adj} \backslash NP)} \quad \frac{\text{tall}}{S_{adj} \backslash NP}}{S \backslash NP} >$$

$$\frac{\text{is}}{(S \backslash NP) / (S_{adj} \backslash NP)} \quad \frac{\frac{\text{pos}}{(S_{adj} \backslash NP) / (S_{adj} \backslash NP)} \quad \frac{\text{tall}}{S_{adj} \backslash NP}}{S_{adj} \backslash NP}}{S \backslash NP} >$$

- Empty category *dgr* for comparative form

$$\frac{\frac{\text{taller}}{S_{adj} \backslash NP} \quad \frac{\text{than Bob}}{(S_{adj} \backslash NP) \backslash (S_{adj} \backslash NP)}}{S_{adj} \backslash NP} < \quad \dashrightarrow \quad \frac{\text{dgr}}{(S_{adj} \backslash NP) / (S_{adj} \backslash NP)} \quad \frac{\text{taller}}{S_{adj} \backslash NP}}{S_{adj} \backslash NP} > \quad \frac{\text{than Bob}}{(S_{adj} \backslash NP) \backslash (S_{adj} \backslash NP)}}{S_{adj} \backslash NP} <$$

- Empty category *dgr2* for equative

$$\frac{\frac{\text{as tall}}{S_{adj} \backslash NP} \quad \frac{\text{as Bob}}{(S_{adj} \backslash NP) \backslash (S_{adj} \backslash NP)}}{S_{adj} \backslash NP} < \quad \dashrightarrow \quad \frac{\text{dgr2}}{(S_{adj} \backslash NP) / (S_{adj} \backslash NP)} \quad \frac{\text{as tall}}{S_{adj} \backslash NP}}{S_{adj} \backslash NP} > \quad \frac{\text{as Bob}}{(S_{adj} \backslash NP) \backslash (S_{adj} \backslash NP)}}{S_{adj} \backslash NP} <$$

¹⁹ See Klein (1991), among others. See also Klein (1980, 1982) for views against this type of analysis.

²⁰ Instead we could introduce type-shifting rules that correspond to the empty categories.

The parsing tree for each sentence in (43) is shown in Figures 2, 3, and 4, respectively.²¹ We assign a uniform semantic representation to each adjective, following the strategy of *generalizing to the worst case* (Montague 1970). An adjective (e.g., *tall*) and its comparative form (e.g., *taller*) of category $S_{adj}\backslash NP$ are uniformly assigned the following term:

$$(44) \quad \lambda Q\delta HI.Q(I(\lambda x.tall(x, \delta), H(tall, \delta)))$$

This term is combined with the other terms including empty elements to form the relevant logical form as illustrated in Figures 2, 3, and 4. For comparison, Figure 5 shows the parsing tree for the case where the explicit degree modifier *4 feet* appears in the adjunct position.

We introduce two variables H and I in the semantic representation in (44). H can be filled in different ways to control the meaning of a *than*-clause, as illustrated in Figure 2 where there is no *than*-clause or Figure 3 where there is a noun phrase in the *than*-clause. I is used to determine whether the entire logical form is of existential type as in (43b) or of universal type as in (43c). We ascribe the negation in A-not-A analysis to *than*, following the analysis of *than*-clauses as introducing negative contexts as presented in the categorial grammar literature (Hendriks 1995).

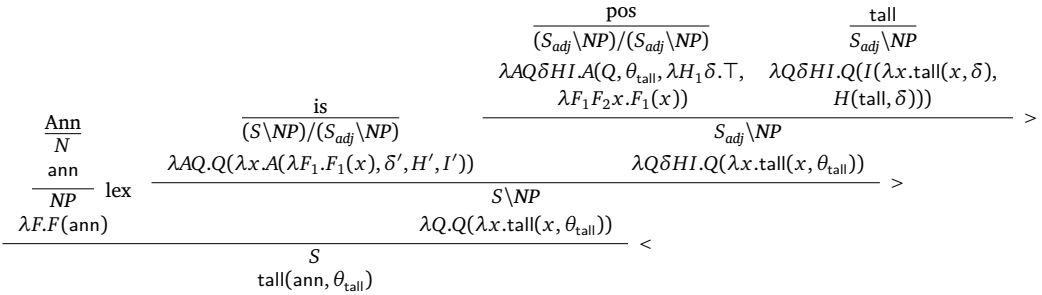


Figure 2: Parsing tree of *Ann is tall*

²¹ In these semantic representations, δ' , H' , and I' are constants to be applied to the vacuous λ -abstraction appearing in the term of category $S_{adj}\backslash NP$.

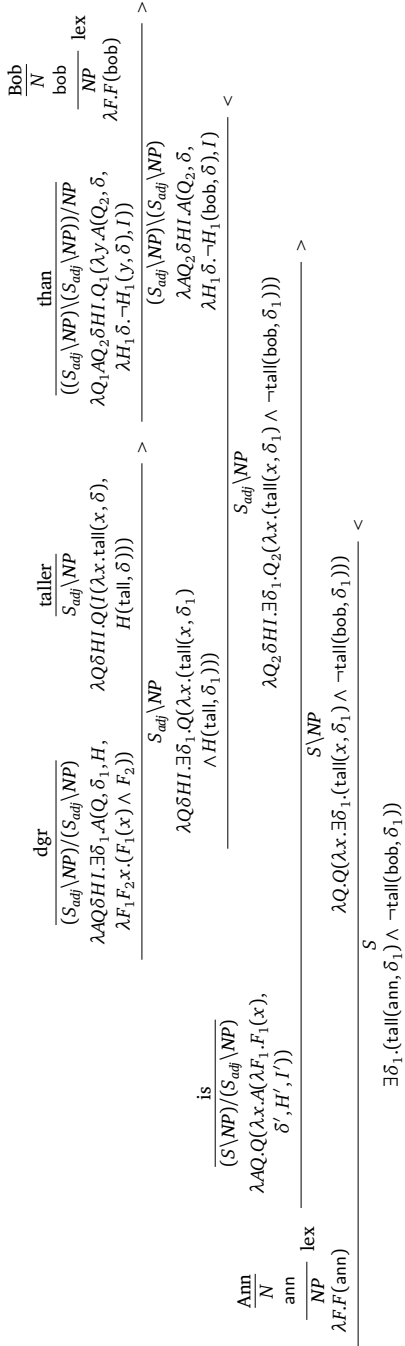
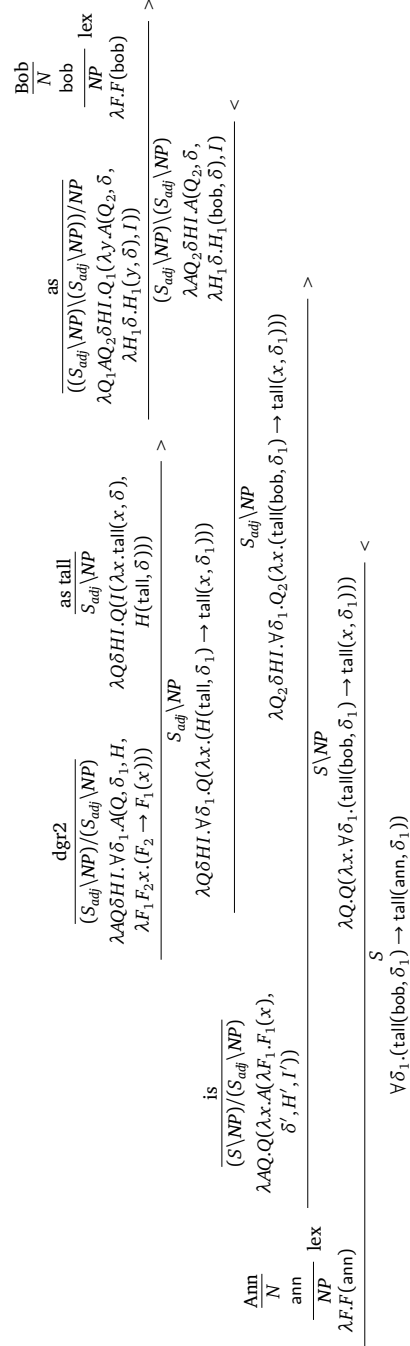

 Figure 3: Parsing tree of *Ann is taller than Bob*

 Figure 4: Parsing tree of *Ann is as tall as Bob*

Table 5: Lexical entries for monotonicity

Expression	Syntactic category	Logical form
2	<i>Num</i>	2
<i>at least</i>	$(NP/N)/Num$	$\lambda\delta F_1 F_2. \exists x(F_1(x) \wedge F_2(x) \wedge \text{many}(x, \delta))$
<i>at most</i>	$(NP/N)/Num$	$\lambda\delta F_1 F_2. \neg \exists x(F_1(x) \wedge F_2(x) \wedge \text{many}(x, \delta + 1))$
<i>exactly</i>	$(NP/N)/Num$	$\lambda\delta F_1 F_2. (\exists x(F_1(x) \wedge F_2(x) \wedge \text{many}(x, \delta))$ $\wedge \forall \delta_1 (\exists x(F_1(x) \wedge F_2(x) \wedge \text{many}(x, \delta_1)) \rightarrow (\delta_1 \leq \delta)))$
ϕ_{exactly}	$(NP/N)/Num$	$\lambda\delta F_1 F_2. (\exists x(F_1(x) \wedge F_2(x) \wedge \text{many}(x, \delta))$ $\wedge \forall \delta_1 (\exists x(F_1(x) \wedge F_2(x) \wedge \text{many}(x, \delta_1)) \rightarrow (\delta_1 \leq \delta)))$

- (45) a. Mary read two books. (Upward)
 b. $\exists x(\text{book}(x) \wedge \text{many}(x, 2) \wedge \exists e(\text{read}(e) \wedge (\text{subj}(e) = \text{mary}) \wedge (\text{obj}(e) = x)))$

For numeral modifiers such as *at least*, we give the category $(NP/N)/Num$. Figure 6 shows an example derivation. The following is an example of a sentence involving a downward monotonic modifier *less than*.

- (46) a. Mary read less than two books. (Downward)
 b. $\neg \exists x(\text{book}(x) \wedge \text{many}(x, 2) \wedge \exists e(\text{read}(e) \wedge (\text{subj}(e) = \text{mary}) \wedge (\text{obj}(e) = x)))$

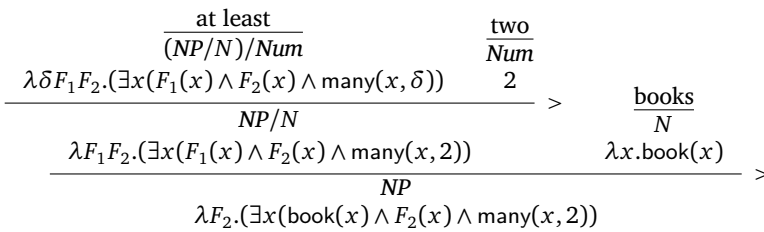


Figure 6:
 Parsing tree of
at least two books

Similarly, we assign syntactic categories like $(NP/N)/Num$ to non-monotonic quantifiers such as *exactly* and *only*. This allows the sentence (47a) to be assigned the complex logical form (47b), which adds the meaning “the number of books Mary read is less than or equal to two” to (45b).

- (47) a. Mary read exactly two books. (Non-monotonicity)
 b. $\exists x(\text{book}(x) \wedge \text{many}(x, 2) \wedge \exists e(\text{read}(e) \wedge (\text{subj}(e) = \text{mary}) \wedge (\text{obj}(e) = x))) \wedge \forall x \forall \delta(\text{book}(x) \wedge \text{many}(x, \delta) \wedge \exists e(\text{read}(e) \wedge (\text{subj}(e) = \text{mary}) \wedge (\text{obj}(e) = x)) \rightarrow (\delta \leq 2))$

Here (45a) has the *at least* reading glossed as “Mary read *at least* two books”. However, it is often natural to interpret (45a) as “Mary read *exactly* three books”. This *exactly* reading is usually derived pragmatically as scalar implicature (SI) (Horn 1973; Gazdar 1979; van Rooij and Schulz 2004). To account for this reading, as an initial attempt, we implement the mechanism of scalar implicature in our system. For this purpose, we use empty category ϕ_{exactly} , which derives the same interpretation as in (47b) for (45a). Thus the system can distinguish two logical forms for a sentence involving a bare numeral, depending on the environment in which it appears.²²

This type of pragmatic ambiguity is related to the fact that $\text{tall}(x, \delta)$ is not interpreted as “ x is *exactly* as tall as δ ” but as “ x is *at least* as tall as δ ”, as mentioned in Section 2.3.1. Thus by inserting the ϕ_{exactly} operator we can uniformly derive SI readings for sentences with numerical expressions as in (45), equatives as in (48), measure phrases as in (49) and (50).

- (48) a. Tom is as tall as Mary.
 \rightsquigarrow Tom is *exactly* as tall as Mary.
 b. $\forall \delta(\text{fast}(\text{mary}, \delta) \leftrightarrow \text{fast}(\text{tom}, \delta))$
- (49) a. John is 5 cm shorter than Bob.
 \rightsquigarrow John is *exactly* 5 cm shorter than Bob.
 b. $\forall \delta(\text{short}(\text{bob}, \delta) \leftrightarrow \text{short}(\text{john}, \delta - 5 \text{ cm}))$
- (50) a. Bob is 170 cm tall.
 \rightsquigarrow Bob is *exactly* 170 cm tall.
 b. $\text{tall}(\text{bob}, 170 \text{ cm}) \wedge \forall \delta(\text{tall}(\text{bob}, \delta) \rightarrow (\delta \leq 170 \text{ cm}))$

On the other hand, negative sentences from (51) to (53) have *at least* reading (see Spector (2013) for an overview). Thus, we do not insert the empty categories in the following constructions.

²²This strategy is similar to the grammatical encoding of scalar implicature proposed by Chierchia (2004).

- (51) a. Peter didn't solve ten problems.
b. $\neg\exists x(\text{problem}(x) \wedge \text{solve}(\text{peter}, x) \wedge \text{many}(x, 10))$
- (52) a. Tom is not as tall as Mary.
b. $\neg\forall\delta(\text{tall}(\text{mary}, \delta) \rightarrow \text{tall}(\text{tom}, \delta))$

(51a) can be interpreted to mean that Peter solved no more than nine problems, i.e., the number of problems Peter solved is less than ten. To derive the reading in (51b), we need to assign the *at least* reading to the numeral adjective *ten*. Similarly, the equative construction with the negation in (52a) has the *at least* reading as in (52b).

Such differences in interpretation occur not only in negation but also more generally in downward environments triggered by negative adjectives such as *fewer than five* and *few*, as well as in the antecedent of a conditional and the restrictor of a universal quantifier.²³

- (53) a. Fewer than five children play in the park.
b. Few boys had three cookies.
c. If Andy is 5 feet tall, he is taller than Bob.
d. Every student who solved 10 problems passed.

We apply the same technique to derive two readings of the determiner *any* (Kadmon and Landman 1993), the existential reading as in (54a) and the universal reading as in (54b).

- (54) a. Bob did not take any exams. (Existential reading)
b. Any owl hunts mice. (Universal reading)

The existential reading is known to be allowed only if *any* appears within the range of DOWNWARD ENTAILING (DE) operators (DE environments) that reverse the direction of entailment, such as negative expressions (Ladusaw 1979). We assume that there is lexical ambiguity in that *any* as an NPI has an existential meaning (Horn 1973; Ladusaw 1979), while *any* as free choice has a universal meaning (Carlson 1981).

To derive two interpretations, we determine from the CCG parsing trees whether *any* appears in the DE environment. Specifically, when

²³Note that there is disagreement as to whether hypothetical clauses are truly SI-free; see the discussion in Breheny (2008) and Spector (2013).

any appears in a non-DE environment, we assign a universal meaning (any_{\forall}), and when *any* appears in a DE environment, we assign an existential meaning (any_{\exists}). This is accomplished in the same way as the process for deriving SIs as described before.

3.3 *Compositional event semantics and adverbial comparatives*

For the compositional account of adverbs and adverbial comparatives, we basically follow the implementation of compositional event semantics presented in Martínez-Gómez *et al.* (2017), which derives the logical form (55b) from the sentence (55a). The compositional derivation is shown in Figure 7.

- (55) a. Tim ran fast.
b. $\exists e(\text{run}(e) \wedge (\text{subj}(e) = \text{tim}) \wedge \text{fast}(e, \theta_{\text{fast}}))$

To derive the logical form in (55b) compositionally, we follow Champollion (2015) to use a continuation variable K which is to be filled in by an adverbial element; If there is no adverbial element as in the root of the parsing tree, it is filled by the constant \top (meaning “true”). We also need to introduce an empty category pos that sets the threshold value to θ_{tall} , in a similar way to the treatment of positive adjectives.

4 EXPERIMENTS

We implemented our system and evaluated it on various NLI datasets. All code and data, including visualized CCG parsing trees with logical forms obtained for each dataset, are made publicly available at <https://github.com/izumi-h/ccgcomp>.

4.1 *System architecture*

Figure 8 shows the pipeline of the proposed system. First, the input consists of a set of premises P_1, \dots, P_n and a hypothesis H , which are mapped to CCG parsing trees. The trees are converted so that they

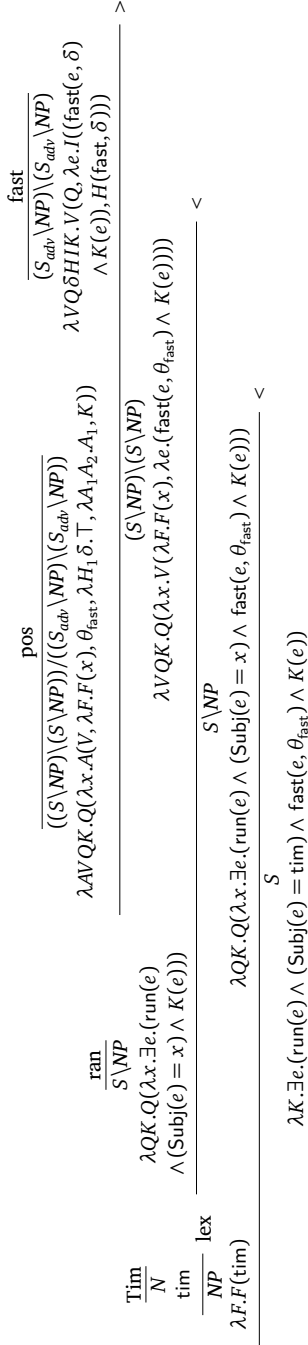


Figure 7: Parsing tree for *Tim ran fast*

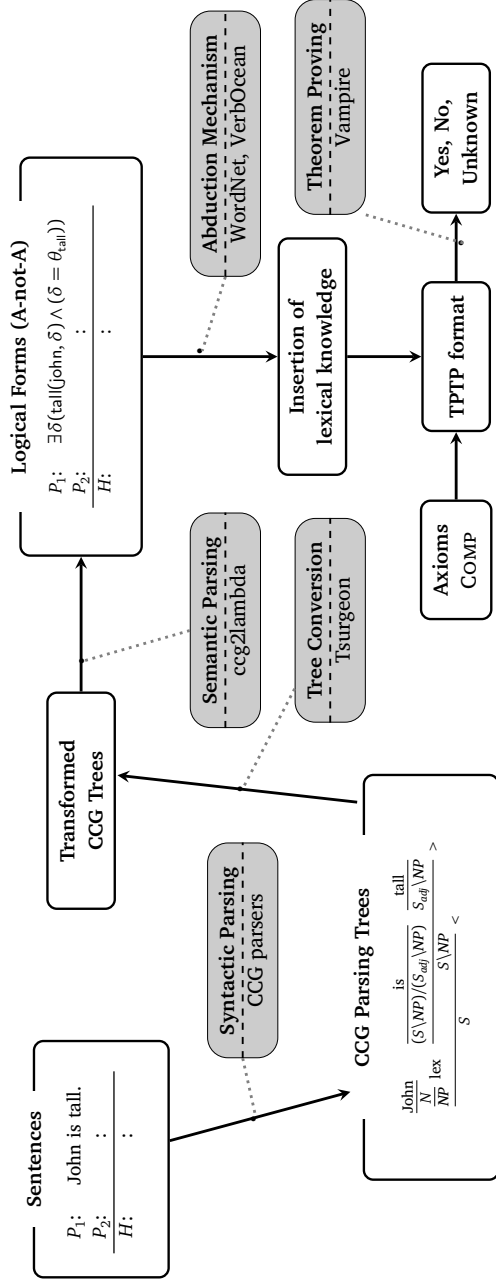


Figure 8: Overview of the proposed system

are suitable for our compositional semantics described in Section 3. The modified trees are mapped to logical forms. Before the process of theorem-proving, the abduction mechanism searches for lexical relations holding on the predicates in the mapped logical forms and introduces them as axioms. Then, a theorem prover checks whether $P_1 \wedge \dots \wedge P_n \rightarrow H$ holds, potentially with the aid of the axioms. The system outputs *yes* (entailment) if $P_1 \wedge \dots \wedge P_n \rightarrow H$ can be proved by a theorem prover, and outputs *no* (contradiction) if the negation of the hypothesis (i.e., $P_1 \wedge \dots \wedge P_n \rightarrow \neg H$) can be proved. If both fail, it tries to construct a counter-model and outputs *unknown* (neutral) if a counter model is found or a timeout occurs.

We build the system on top of off-the-shelf CCG parsers and a theorem prover. To these existing tools, we mainly add three components, (1) rules to transform CCG derivation trees, (2) rules to map CCG derivation trees to logical forms, and (3) axioms for comparatives to derive theorems. We will explain each step in the pipeline in detail.

1. Syntactic parsing To obtain CCG parsing trees we use three CCG parsers to mitigate parsing errors: C&C (Clark and Curran 2007), Easy-CCG (Lewis and Steedman 2014), and depccg (Yoshikawa et al. 2017). For all parsers, we use the standard model trained on the original CCG-Bank. We also use POS tagging to supplement the information available from CCG trees. For example, CCG categories do not distinguish positive and comparative forms of adjectives. To remedy this, we use POS tags *JJ* and *JJR* for positive and comparative forms. For POS tagging, we use the C&C POS tagger for C&C and spaCy²⁴ for depccg.

2. Tree conversion To modify CCG parsing trees, we use Tsurgeon (Levy and Andrew 2006). We use 125 entries (regex rewriting rules) in the Tsurgeon script. In addition to modifying trees, we use the following rules to add information needed to derive logical forms in our compositional semantics. There are five types of rewriting rules.

- **Multiword Expression.** We add rules to join multiword expressions for determiners; e.g. *a lot of* to $a \sim lot \sim of$ and *a few* to $a \sim few$.
- **Empty category.** We insert empty categories and add syntactic features to CCG categories as described in Section 3.

²⁴<https://github.com/explosion/spaCy>

- Adjective type. Based on the analysis presented in Section 2, we classify adjectives into six types: extensional positive (*POS*), extensional negative (*NEG*), intensional positive (*POS-INT*), intensional negative (*NEG-INT*), non-gradable (*PRE*), or non-subjective (*N-SUB*). To classify positive and negative adjectives, we use SentiWordNet (Baccianella *et al.* 2010). For the other types, we prepare hand-written rules for a set of the adjectives appearing in the FraCaS dataset.
- Negative Polarity *any*. We distinguish any_{\forall} and any_{\exists} according to its environment as described in Section 3.2.
- Lemmatization. Comparative forms of adverbs are converted to positive forms (e.g., *faster* to *fast*), and positive forms of adverbs are converted to corresponding adjectives (e.g., *slowly* to *slow*). We use the WordNet (Miller 1995) library in NLTK²⁵ for this conversion.

3. Semantic parsing To implement compositional semantics, we use the semantic parsing platform *ccg2lambda* (Martínez-Gómez *et al.* 2016), which uses λ -calculus to obtain logical forms. We extend the schematic lexical entries (called semantic templates) for FOL event semantics proposed in Martínez-Gómez *et al.* (2017) to handle linguistic phenomena based on degree-based semantics. In this system, semantic parsing is performed using two different semantic templates to manipulate the scope of negation in logical forms. If input sentences contain the negation *not* or *n't*, the proof is attempted in two different logical forms with negation taking wide scope or narrow scope. The total number of lexical entries assigned to CCG categories is 551, and the number of entries directly assigned to particular words (e.g., *than* and *as* for comparatives and items for quantifiers) is 151.

4. Abduction mechanism To handle basic lexical inferences, we adapt an abduction mechanism presented in Martínez-Gómez *et al.* (2017) to our framework. Given logical forms for premises, the abduction mechanism searches lexical relations from two lexical knowledge bases: WordNet (Miller 1995) and VerbOcean (Chklovski and Pantel 2004). Following Martínez-Gómez *et al.* (2017), we use seven rela-

²⁵<https://www.nltk.org/>

tionships such as antonym and hypernym and add the corresponding axioms. The acquisition of antonym relations of gradable adjectives such as *tall* and *short* is also based on the use of this mechanism.

5. Theorem proving For theorem proving, we use a resolution-based FOL prover Vampire 4.4 (Kovács and Voronkov 2013),²⁶ which accepts TFF forms with arithmetic operations. The proof runs in the automatic modes *casc* and *casc_sat*, which automatically select a series of strategies that attempt to prove a particular problem. While *casc* is aimed at solving theorems, *casc_sat* is aimed at solving satisfiable or non-theorem problems, that is, those problems where there is a model in which the premises are true but the conclusion is false (i.e., there is a counter-model for the inference). In our system, we first try to prove the problem in *casc* mode and then try to prove it again in *casc_sat* mode for any problems that are labeled *unknown*. We set the timeout at 7 sec in *casc* mode and 1 sec in *casc_sat*. We add the four axiom schemata described in Section 2, which we call the axiomatic system COMP, before starting the process of theorem proving. Each axiom scheme is instantiated by gradable adjectives appearing in the target sentences.

We run a process of theorem proving for each of the three parsers and obtain three outputs. If the three outputs are different, we choose the system answer in the following way: if two answers are *yes* (resp. *no*), then the system answer is *yes* (resp. *no*), no matter what the other answer is; if one answer is *yes* (resp. *no*) and the others are *unknown*, the system answer is *yes* (resp. *no*); if all answers are different, then the system answer is *unknown*.

4.2

Datasets

For evaluation, we use five NLI datasets containing linguistically challenging problems with quantifiers, adjectives, adverbs, comparatives, and lexical knowledge. Table 6 shows some examples in each dataset. **FraCaS** FraCaS (Cooper et al. 1996) is a dataset comprising nine sections, each of which contains semantically challenging inferences related to various linguistic phenomena. In this study, we target four

²⁶<https://github.com/vprover/vampire>

Implementing natural language inference for comparatives

Table 6: Examples of entailment problems from the FraCaS, MED, SICK, HANS, and CAD datasets. They are solved by our system but not by the DL models

Dataset	Label	ID	Example (premises and hypothesis)	Gold label
FraCaS	<i>Adj</i>	209	P_1 : Mickey is a small animal. P_2 : Dumbo is a large animal. H : Mickey is larger than Dumbo.	No
	<i>Com</i>	241	P_1 : ITEL won more orders than APCOM lost. P_2 : APCOM lost ten orders. H : ITEL won at least eleven orders.	Yes
MED	<i>gq</i>	485	P : Exactly 12 aliens threw some tennis balls. H : Exactly 12 aliens threw some balls.	Unknown
		1021	P : More than five campers have had a sunburn or caught a cold. H : More than five campers have caught a cold.	Unknown
	<i>gqlx</i>	176	P : Few aliens saw birds. H : Few aliens saw doves.	Yes
SICK	-	1357	P : A puppy is repeatedly rolling from side to side on its back. H : A dog is rolling from side to side.	Yes
		4789	P : There is no woman riding on an elephant. H : A woman is opening a soda and drinking it.	Unknown
HANS	-	16005	P : Happy authors advised the artists. H : Authors advised the artists.	Yes
		23990	P : The student recommended the author, or the presidents believed the managers. H : The student recommended the author.	Unknown
CAD	-	001	P_1 : John is 5 cm taller than Bob. P_2 : Bob is 170 cm tall. H : John is 175 cm tall.	Yes
		103	P_1 : Bob is not tall. P_2 : John is not tall. H : John is taller than Bob.	Unknown
		115	P : Exactly seven students smiled. H : At most nine students smiled.	Yes
		157	P_1 : Ann runs as fast as Luis does. P_2 : Ann runs slowly. H : Luis runs fast.	No

sections: Generalized Quantifiers (*GQ*: 73 problems), Adjectives (*Adj*: 22 problems), Comparatives (*Com*: 31 problems), and Attitudes (*Att*: 13 problems). The Comparative section contains a complex inference that requires arithmetic operation, such as ID-241 in Table 6.

MED MED (Yanaka et al. 2019) collects problems with monotonicity inferences with generalized quantifiers and lexical knowledge via crowdsourcing. We use a portion of the dataset tagged with *gqlex* and *gq*, those inferences that require lexical knowledge (*gqlex*: 691 problems) and those that do not (*gq*: 498 problems).

SICK We use the 2014 version of SemEval (Marelli et al. 2014) of SICK dataset. The dataset contains 4,927 problems for test set. SICK is designed to evaluate compositional inferences involving lexical knowledge and logical operations such as negation and quantifiers.

HANS HANS (McCoy et al. 2019) is a dataset containing problems that DL-based systems tend to erroneously output *yes* for cases in which they rely on simple heuristics, for example, problems where the hypothesis is a constituent or a sub-string of the premise, such as disjunctive sentences (e.g., HANS-23990 in Table 6), and problems related to those concerning adjectives and adverbs (e.g., ID-16005 in Table 6). The entire test set contains 30,000 problems, which are divided into entailment (*yes*) and non-entailment (*unknown*) problems.

CAD The above four datasets do not cover linguistically interesting inferences such as ones concerned with adverb phrases (e.g., dropping adverbial phrases and comparative forms of adverbs). Accordingly, we created a new dataset containing 257 inference problems concerning adjectives, comparatives, adverbs, and quantifiers. The dataset also includes problems related to SI (29 problems), to which both gold labels for semantic interpretation and pragmatics interpretation (i.e., those considering SIs) are annotated. We collected a set of inferences (13 problems) from linguistics papers (Klein 1982; Lasersohn 2006) and created more problems by adding negation and degree modifiers (e.g., *very*), changing numerical expressions, replacing positive and negative adjectives (e.g., *large* to *small*), or swapping the premise and hypothesis of an inference. Of the 257 problems, 137 are single-premise problems, and 120 are multi-premise problems. The distribution

of gold answer labels is (*yes/no/unknown*) = (110/70/77). All of the gold labels were checked by an expert in linguistics.

Results and discussion

4.3

Tables 7, 8, 9, 10, and 11 show the results of the evaluation. We will describe the details of each result from Section 4.3.1 to Section 4.3.5 below. Since MED and HANS use binary labels (*yes* and *unknown*), for these two datasets we modify the system so that it outputs *yes* if the hypothesis can be proved from the premise; otherwise, the output is *unknown*. *Majority* is the accuracy of the majority baseline. Before looking at the details of the results, let us explain the setting of an ablation analysis and the systems being compared.

Ablation analysis To gain insights into the impact of each component, we performed an ablation analysis on overall performance.

- *Plain* is the accuracy of the system with the transformation of CCG parsing trees only.
- *+ abduction* is the accuracy achieved by the insertion of lexical knowledge through the implementation of the abduction mechanism, as described in Section 4.1.
- *+ rule* is the accuracy achieved by the addition of hand-coded rules. Some errors were caused by failing to assign correct POS tags and lemmas to comparatives. For example, *cleverer* is wrongly assigned *NN* rather than *JJR* (FraCaS-217). To estimate the upper

FraCaS					
Section		<i>GQ</i>	<i>Adj</i>	<i>Com</i>	<i>Att</i>
#All		73	22	31	13
Majority		.49	.41	.61	.62
DL	RB	.73	.45	.52	.69
Logic	MN	.77	.68	.48	.77
	LP	.93	.73	–	.92
Ours	plain	.96	.82	.90	.92
	+ abduction	.97	.82	.90	.92
	+ abduction + rule	.99	.95	.90	.92

Table 7:
Accuracy on FraCaS dataset

Table 8:
Accuracy on MED dataset

MED			
Label		<i>gq</i>	<i>gqlx</i>
#All		498	691
Majority		.58	.63
DL	BERT	.56	.58
	BERT+	.54	.68
	RB	.57	.55
Ours	plain	.97	.67
	+ abduction	.97	.91
	+ abduction + rule	.97	.92

Table 9:
Accuracy on SICK dataset

SICK		
#All		4,927
Majority		.57
DL	RB	.56
Logic	LP	.81
	MG	.83
Ours	plain	.76
	+ abduction	.82
	+ abduction + rule	.82

bound on the accuracy of our system by reducing error propagation, we added hand-coded rules to assign correct POS tags and lemmas (23 words). We also added two rules to join multiword expressions to derive correct logical forms (*law lecturer* and *legal authority* in FraCaS-214, 215).

- For CAD, we also experimented with an implementation for SI, as described in Section 3.2. We use 23 rules in Tsurgeon scripts. The accuracy is shown in *+ implicature*.

Comparison of existing NLI systems We compare our system with other logic-based systems and recent DL-based systems. For logic-based systems, we mainly compare three systems based on CCG parsers and theorem proving:

- MN (Mineshima *et al.* 2015) uses a CCG parser (C&C; Clark and Curran 2007) and implements a theorem prover for NLI based on HOL. This system uses Coq (Castéran and Bertot 2004), an

HANS			
Gold		<i>yes</i>	<i>unknown</i>
#All		15,000	15,000
Majority		.50	.50
DL	BF	.87	.61
	RB	1.0	.56
Symbolic		GKR4	.84 .59
DL & Symbolic	HNB	.84	.54
	HNX	.83	.25
Ours	plain	.98	.83
	+ abduction	.98	.83
	+ abduction + rule	.98	.83

Table 10:
Accuracy on HANS dataset

CAD		
#All		257
Majority		.43
DL	RB	.58
Ours	plain	81
	+ abduction	.81
	+ abduction + rule	.82
	+ abduction + rule + implicature	.92

Table 11:
Accuracy on CAD dataset

interactive natural deduction theorem prover in a fully automated way.

- LP (Abzianidze 2015, 2016) is a system that uses two CCG parsers (C&C and EasyCCG) and implements a natural logic inference system based on semantic tableau. The system uses the theorem prover for HOL (Abzianidze 2015) based on *natural logic* (Lakoff 1970; van Benthem 1986).
- MG (Martínez-Gómez *et al.* 2017) is a system based on two CCG parsings (C&C and EasyCCG) with compositional event semantics and theorem proving, an updated version of MN.

Table 12 summarizes the characteristics of the logic-based systems, including ours.

For DL-based systems, we compare our system with the following.

Table 12: Existing logic-based NLI systems

System	Proof strategy	Logic	Prover	Abduction	Arithmetic
MN	natural deduction	HOL	Coq		
LP	tableau	Natural Logic/HOL	NLogPro	✓	
MG	natural deduction	FOL	Coq	✓	
Ours	resolution	Typed FOL	Vampire	✓	✓

- BERT shows the performance of a BERT model fine-tuned with MultiNLI, and BERT+ shows that of a BERT model with data augmentation for approximately 36,000 monotonicity inferences in addition to the MultiNLI training set. Both models were tested and reported in Yanaka *et al.* (2019).
- BF is a BiLSTM model trained on MultiNLI, which is a state-of-the-art model on HANS. The model was tested and reported in Yaghoobzadeh *et al.* (2019).
- RB shows that we use a state-of-the-art model RoBERTa (Liu *et al.* 2019) trained on MultiNLI (Williams *et al.* 2018) using the implementation provided in AllenNLP.²⁷ The accuracies in the table represent those we tested.

In addition, for HANS dataset (see Table 10) we refer to the accuracy of a hybrid system with a symbolic component and a DL component reported in Kalouli *et al.* (2020), where three systems, HNB, HNX, and GKR4 are distinguished.

- HNB uses the Graphical Knowledge Representation (GKR) context graphs (Kalouli and Crouch 2018) to determine whether a given inference is semantically complex or not; for a complex problem, it uses a symbolic component that makes use of multiple graphs to represent sentence information, while for a simple problem, it uses a BERT model for determining the entailment label.
- HNX is a system that uses an XLNet model as the DL-model.
- GKR4 is a system that only uses the symbolic component.

²⁷<https://github.com/allenai/allennlp>

Table 7 shows the results on FraCaS. For comparison, we use the two logic-based systems (MN and LP) and the DL-based system (RB). Our system achieved very high accuracy and outperformed the DL-system by a large margin. Table 6 shows examples that were solved by our system but not by the DL-system. Our system successfully solved inferences such as FraCaS-209 that involve antonyms, which the DL-system found particularly difficult to solve. FraCaS-241 is a complex inference with numerical expressions and clausal comparatives. This problem is solved by our system but by neither of the other logic-based systems, nor by the DL-system.

One problem that our system was not yet able to solve is concerned with comparative ellipsis. The sentence *APCOM has a more important customer than ITEL* (FraCaS-244, 245) can have two interpretations (56H) or (57H).

- (56) *P*: APCOM has a more important customer than ITEL.
H: APCOM has a more important customer than ITEL is.
 (FraCaS-244, gold label: *yes*)
- (57) *P*: APCOM has a more important customer than ITEL.
H: APCOM has a more important customer than ITEL has.
 (FraCaS-245, gold label: *yes*)

Our system does not have a component to handle this type of comparative ellipsis and can only derive the interpretation in (56H), thus failing to provide the correct judgement for FraCaS-245.

Table 8 shows the results on MED. Our system outperformed the DL-based systems. MED-176 and MED-485 in Table 6, which involve a downward quantifier (*few*) and a non-monotonic quantifier (*exactly 12*), respectively, are examples that our system correctly solved but the DL-models did not. For the problems containing lexical inferences in *gqlx*, our system achieved a high improvement in accuracy (67% to 91%) by implementing the abduction mechanism, showing that our system is compatible with lexical knowledge.

Table 9 shows the results on SICK. Our system outperformed the DL-based system (RB) and achieved comparable results with the logic-based systems (LP and MG). SICK-1357 in Table 6 is an example involving the lexical inference from *puppy* to *dog*. Our system correctly predicted the *yes* label for this problem, while the DL-based system (RB) predicted the *no* label. SICK-4789 in Table 6 contains negation *no*; our system can represent what information is negated by the scope of the negation in the logical form, but DL-based systems tend to answer *no* to such inferences.

One problem that was solved by MG but not by our system is the following.

(58) *P*: Someone is on a black and white motorcycle and is standing on the seat.

H: A motorcycle rider is standing up on the seat of a white motorcycle. (SICK-199, gold label: *unknown*)

In the case of MG, which implements *on-demand* abduction (an axiom is added during the process of constructing a natural deduction proof), the premise sentence does not generate any axioms, while in our system, the axiom $\forall x(\text{black}(x) \rightarrow \neg\text{white}(x))$ based on the antonym is added before the proof process, making the premise inconsistent with the same entity being white and not white at the same time. Thus, our system incorrectly predicts *yes* by the principle of explosion (i.e., any proposition can be derived from the contradiction).

Another type of error is found in the following problem.

(59) *P*: A man is holding a small animal in one hand.

H: A man is holding an animal, which is small, in one hand. (SICK-4690, gold label: *yes*)

The gradable adjective *small* in *P* is a nominal adjective, generating the threshold $\theta_{\text{small}}(\text{animal})$, while that in *H* is a predicate adjective, generating the threshold $\theta_{\text{small}}(U)$ with the universal set U . Due to this mismatch in the comparison class, the system failed the proof.

Overall, our system achieved performance comparable to that of MG based on event semantics, thus showing the compatibility of event semantics and degree semantics.

Table 10 shows the results on HANS. We compared our system with the following systems: BF, RB, GKR4, HNB, and HNX.

McCoy *et al.* (2019) reported that DL-based systems tend to erroneously output *yes* for cases in which the hypothesis was a constituent or a substring of the premise, such as disjunctive sentences (e.g., HANS-23990 in Table 6). To see how a system performs in these cases, we present the accuracy for each gold answer label (*yes* and *unknown*). While accuracy whose gold label is *yes* was close to 100% in both our system and the DL-based system (RB), the accuracy of our system was higher than that of RB when the label is *unknown* (83% vs. 56%).

One reason for the relatively low accuracy (83%) of our system in comparison with its performance on the other datasets is parse error. HANS contains syntactically complex sentences such as *The author who advised the lawyer supported the athlete* (HANS-12182, *subsequence*), for which the CCG parsers output incorrect parses. For example, in the case of C&C parser, the substring of the sentence, *The author who advised*, is parsed as *NP*, separated from the object noun phrase *the lawyer*. The rest of the sentence, *the lawyer supported the athlete*, is parsed as *S* and shifted to *NP\NP*. For *depccg*, the sentence *The athletes presented in the library* (HANS-13002) is parsed as *NP* instead of *S*.

Another type of error is concerned with an inference involving a modal adverb, e.g., the inference from *Probably the secretary admired the athlete* to *The secretary admired the athlete* (HANS-24034). The gold label is *unknown*, but our system predicts *yes* since any adverb can be dropped in the current implementation. A more fine-grained classification of adverbs will be needed to handle this type of inference.

Table 11 shows the results on CAD. Our system outperformed the DL-based system (RB). Our system was able to solve inference involving numerical computations (CAD-001,115) and antonym conversion for adverbs (CAD-157) shown in Table 6, while RB incorrectly predicted *unknown* for CAD-001, *no* for CAD-115, and *yes* for CAD-157.

Table 13 shows some example problems from CAD where the gold label changes between semantics and pragmatics. In the setting shown

Table 13:
Examples
of entailment
problems for SI
of gradable
expressions
from CAD

ID	Premises and hypothesis	Gold label	
		Semantics	Pragmatics
002	P_1 : John is 5 cm shorter than Bob. P_2 : Bob is 170 cm tall. H : John is 165 cm tall.	Unknown	Yes
052	P_1 : Bob is much taller than John. P_2 : Bob is a 5 feet tall boy. H : John is shorter than 5 feet.	Unknown	Yes
112	P : Bob saw four students. H : Bob saw three students.	Yes	No
145	P : Ann runs as fast as Luis. H : Ann runs faster than Luis.	Unknown	No
245	P : There are a few books. H : There are many books.	Unknown	No

in + *implicature*, our system was able to solve problems involving SIs, which led to the improvement in accuracy. Our system also solved complex inferences (CAD-002,052) that involve antonyms and numerical expressions.

There are still problems that need to be addressed. For example, the sentence *Jones drives more carefully today than yesterday* (CAD-183) conjoins two adverbs *today* and *yesterday* by *than*. The current system does not derive the correct logical form for this type of complex coordinate structure formed by *than*-clauses. Also, in the case of the sentence *Chris is more happy than Alex is sad* (CAD-013), which is an instance of COMPARATIVE SUBDELETION (Bresnan 1975), the clause *Alex is sad* is simply parsed as S and mapped to $\text{sad}(\text{alex}, \theta_{\text{sad}})$, making it impossible to compare it the degrees introduced by the main clause. Further improvement to CCG parsing is needed to handle complex coordinate constructions and comparative subdeletion.

4.3.6

Comparison of CCG parsers

For a comprehensive comparison, Table 14 shows accuracies for each CCG parser at its best performances in our system. It shows that our system achieved the best accuracy with *depccg* in most datasets. One of the reasons for this is that the tree conversion is designed based on the outputs of *depccg*. It is also noted that as described in Section 4.1,

Table 14: Accuracy for each CCG parser at the best performances

Parser	FraCaS				MED		SICK	HANS		CAD
	<i>GQ</i>	<i>Adj</i>	<i>Com</i>	<i>Att</i>	<i>gq</i>	<i>gqlx</i>		<i>yes</i>	<i>unknown</i>	
Multi	.99	.95	.90	.92	.97	.92	.82	.98	.83	.92
C&C	.82	.86	.61	.69	.93	.88	.76	.80	.85	.52
EasyCCG	.97	.86	.55	.92	.97	.89	.77	.93	.98	.53
depccg	.96	.95	.90	.92	.96	.91	.77	.97	.95	.92

our system prioritizes *yes* (or *no*) rather than *unknown* among the answers given by the three parsers. For this reason, parse errors caused by C&C led to a decrease in overall accuracy in the case of *unknown* problems, as shown in Table 14. It would be necessary to refine the system’s answer selection mechanism when multiple parsers are used.

General discussion

4.3.7

FraCaS and CAD are datasets manually constructed by experts; their size is small (FraCaS: 139 , CAD: 257) but contains linguistically challenging inferences. The evaluation of FraCaS and CAD shows that the proposed system can handle the various types of complex inferences discussed in formal semantics, including adjectives, comparatives, and generalized quantifiers.

MED, SICK, and HANS are crowdsourced or automatically generated datasets that are larger in size than FraCaS and CAD (MED: 1,189, SICK: 4,297, HANS: 30,000). The inferences in MED, SICK, and HANS are single-premise inferences, simpler than FraCaS and CAD but containing lexical inferences (MED, SICK) and logical phenomena such as quantification, disjunction, and negation (MED, SICK, HANS). The experimental results for MED, SICK, and HANS indicate that our system can successfully handle these types of inferences.

The ablation study aimed to estimate the effects of three additional mechanisms: (1) abduction (lexical inference) mechanism, (2) hand-written rules for error correction, and (3) mechanisms for handling implicature. The results of the ablation study for each dataset show that the system improved accuracy for the datasets that include lexical inference (indicated by +abduction in MED and SICK) and for the dataset containing implicature (indicated by +implicature in

CAD). These results were more or less expected, but still seem to be meaningful enough to show the effectiveness of the additional components.

5

CONCLUSION

We presented a CCG-based compositional semantics and inference system for comparatives and other related constructions. The logical forms used are based on A-not-A analysis in formal semantics and the inference system is combined with the axioms of COMP based on TFF forms acceptable in efficient FOL provers. The entire system is transparently composed of multiple modules and can solve complex inferences in an explanatory manner. The system can handle gradable expressions such as comparatives and adjectives, which are a weakness of conventional logic-based systems. The system can also be extended to handle generalized quantifiers, adverbs, and numerals while maintaining the advantages of the original system for adjectival comparatives. For adverbs in particular, by combining two semantic theories, degree semantics and event semantics, we were able to assign appropriate logical forms to solve complex inferences.

For evaluation, we used various NLI datasets containing linguistically challenging problems. The results showed that our system works well on complex logical inferences for which standard DL-based systems show poor performance. In addition, our system has the advantage that it does not require large amounts of training data, such as SNLI or MultiNLI, as opposed to DL-based systems.

It might be objected that the results on the DL models in Section 4.3 were not surprising, because these models were trained on SNLI and MultiNLI that do not target the logical and numerical inferences we are concerned with in this study. However, it is fair to say that it is challenging to generate effective training data for handling various complex inferences with comparatives, numerals, and generalized quantifiers. This study can also contribute to the study of computational modeling and to the evaluation of formal semantic

theories, as well as to the creation of challenging NLI problems that DL-based models need to address.

In addition to the problems we have already mentioned, there are still some unresolved issues in this study. For example, we need to extend our analysis to cover more challenging comparative constructions such as GAPPING (Ross 1970; Hendriks 1995). It would also be interesting to modify CCGbank, which is the training data for CCG parsers, based on the proposed transformation of parsing trees. These are left for future work.

ACKNOWLEDGEMENT

We thank the three reviewers for their valuable comments. This work was supported by JSPS KAKENHI Grant Number JP21K00016, JSPS KAKENHI Grant Number JP18H03284, JST CREST Grant Number JP-MJCR2114, and JST CREST Grant Number JPMJCR20D2, Japan.

REFERENCES

- Lasha ABZIANIDZE (2015), A tableau prover for natural logic and language, in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 2492–2502.
- Lasha ABZIANIDZE (2016), Natural solution to FraCaS entailment problems, in *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics (*SEM)*, pp. 64–74.
- Stefano BACCIANELLA, Andrea ESULI, and Fabrizio SEBASTIANI (2010), SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining, in *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, pp. 2000–2004.
- Jon BARWISE and Robin COOPER (1981), Generalized quantifiers and natural language, *Linguistics and Philosophy*, 4(2):159–219.
- Jean-Philippe BERNARDY and Stergios CHATZIKYRIAKIDIS (2017), A type-theoretical system for the FraCaS test suite: Grammatical Framework meets Coq, in *IWCS 2017 – 12th International Conference on Computational Semantics*.

- Johan BOS (2008a), Let's not argue about Semantics, in Nicoletta CALZOLARI, Khalid CHOUKRI, Bente MAEGAARD, Joseph MARIANI, Jan ODIJK, Stelios PIPERIDIS, and Daniel TAPIAS, editors, *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, pp. 2835–2840.
- Johan BOS (2008b), Wide-coverage semantic analysis with Boxer, in *Proceedings of the 2008 Conference on Semantics in Text Processing (STEP)*, pp. 277–286.
- Samuel R. BOWMAN, Gabor ANGELI, Christopher POTTS, and Christopher D. MANNING (2015), A large annotated corpus for learning natural language inference, in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 632–642.
- Richard BREHENY (2008), A new look at the semantics and pragmatics of numerically quantified noun phrases, *Journal of Semantics*, 25(2):93–139.
- Joan W. BRESNAN (1975), Comparative deletion and constraints on transformations, *Linguistic Analysis*, 1:25–74.
- Greg CARLSON (1981), Distribution of free-choice Any, in Masek HENDRICK and MILLER, editors, *Papers from the Seventeenth Regional Meeting of the Chicago Linguistics Society*, 17, pp. 8–23.
- Pierre CASTÉRAN and Yves BERTOT (2004), *Interactive theorem proving and program development. Coq'Art: The Calculus of inductive constructions*, Springer.
- Lucas CHAMPOLLION (2015), The interaction of compositional semantics and event semantics, *Linguistics and Philosophy*, 38(1):31–66.
- Stergios CHATZIKYRIAKIDIS and Jean-Philippe BERNARDY (2019), A wide-coverage symbolic natural language inference system, in *Proceedings of the 22nd Nordic Conference on Computational Linguistics (NoDaLiDa)*, pp. 298–303.
- Stergios CHATZIKYRIAKIDIS and Zhaohui LUO (2014), Natural language inference in Coq, *Journal of Logic, Language and Information*, 23(4):441–480.
- Gennaro CHERCHIA (2004), Scalar implicatures, polarity phenomena and the syntax/pragmatics interface, in Adriana BELLETTI, editor, *Structures and Beyond*, pp. 39–103, Oxford University Press.
- Timothy CHKLOVSKI and Patrick PANTEL (2004), VerbOcean: Mining the web for fine-grained semantic verb relations, in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 33–40.
- Stephen CLARK and James R CURRAN (2007), Wide-coverage efficient statistical parsing with CCG and log-linear Models, *Computational Linguistics*, 33(4):493–552.
- Robin COOPER, Richard CROUCH, Jan VAN EIJCK, Chris FOX, Josef VAN GENABITH, Jan JASPERS, Hans KAMP, Manfred PINKAL, Massimo POESIO, Stephen G. PULMAN, et al. (1996), FraCaS – A framework for computational semantics, *Deliverable D6*.

Implementing natural language inference for comparatives

- Max J. CRESSWELL (1976), The semantics of degree, in Barbara PARTEE, editor, *Montague Grammar*, pp. 261–292, Academic Press.
- Donald DAVIDSON (1967), The logical form of action sentences, in Nicholas RESCHER, editor, *The Logic of Decision and Action*, pp. 81–95, University of Pittsburgh Press.
- Gerald GAZDAR (1979), *Pragmatics: Implicature, Presupposition, and Logical Form*, Academic Press.
- Martin HACKL (2000), *Comparative Quantifiers*, Ph.D. thesis, Massachusetts Institute of Technology.
- Michael HAHN and Frank RICHTER (2016), Henkin semantics for reasoning with natural language, *Journal of Language Modelling*, 3(2):513–568.
- Izumi HARUTA, Koji MINESHIMA, and Daisuke BEKKI (2020), Combining event semantics and degree semantics for natural language inference, in *Proceedings of the 28th International Conference on Computational Linguistics (COLING)*, pp. 1758–1764.
- Herman HENDRIKS (1993), *Studied Flexibility: Categories and Types in Syntax and Semantics*, Ph.D. thesis, ILLC, University of Amsterdam.
- Petra HENDRIKS (1995), *Comparatives and Categorical Grammar*, Ph.D. thesis, University of Groningen.
- Julia HOCKENMAIER and Mark STEEDMAN (2007), CCGbank: A corpus of CCG derivations and dependency structures extracted from the Penn Treebank, *Computational Linguistics*, 33(3):355–396.
- Matthew HONNIBAL, James R. CURRAN, and Johan BOS (2010), Rebanking CCGbank for improved NP interpretation, in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 207–215.
- Laurence Robert HORN (1973), *On the Semantic Properties of Logical Operators in English*, Ph.D. thesis, University of California.
- Nirit KADMON and Fred LANDMAN (1993), Any, *Linguistics and Philosophy*, 16(4):353–422.
- Aikaterini-Lida KALOULI and Richard CROUCH (2018), GKR: the graphical knowledge representation for semantic parsing, in *Proceedings of the Workshop on Computational Semantics beyond Events and Roles (SemBEaR)*, pp. 27–37.
- Aikaterini-Lida KALOULI, Richard CROUCH, and Valeria DE PAIVA (2020), Hy-NLI: a hybrid system for natural language inference, in *Proceedings of the 28th International Conference on Computational Linguistics (COLING)*, pp. 5235–5249.
- Hans KAMP (1975), Two theories about adjectives, in Edward L KEENAN, editor, *Formal Semantics of Natural Language*, pp. 123–155, Cambridge University Press.

- Ewan KLEIN (1980), A semantics for positive and comparative adjectives, *Linguistics and Philosophy*, 4(1):1–45.
- Ewan KLEIN (1982), The interpretation of adjectival comparatives, *Journal of Linguistics*, 18(1):113–136.
- Ewan KLEIN (1991), Comparatives, in Arnim VON STECHOW and Dieter WUNDERLICH, editors, *Semantics: An International Handbook of Contemporary Research*, pp. 673–691, de Gruyter.
- Laura KOVÁCS and Andrei VORONKOV (2013), First-order theorem proving and Vampire, in *Proceedings of the 25th International Conference on Computer Aided Verification*, volume 8044, pp. 1–35.
- William A. LADUSAW (1979), *Polarity Sensitivity as Inherent Scope Relations*, Ph.D. thesis, University of Texas.
- George LAKOFF (1970), Linguistics and natural logic, *Synthese*, 22(1-2):151–271.
- Richard K. LARSON (1988), Scope and comparatives, *Linguistics and Philosophy*, 11(1):1–26.
- Peter N. LASERSON (2006), Event-based semantics, in Keith BROWN, editor, *Encyclopedia of Language and Linguistics*, volume 4, pp. 316–320.
- Daniel LASSITER (2015), Adjectival modification and gradation, in *The Handbook of Contemporary Semantic Theory*, pp. 141–167, John Wiley & Sons, Ltd.
- Adrienne LEHRER and Keith LEHRER (1982), Antonymy, *Linguistics and Philosophy*, 5(4):483–501.
- Roger LEVY and Galen ANDREW (2006), Tregex and Tsurgeon: tools for querying and manipulating tree data structures, in *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, pp. 2231–2234.
- David LEWIS (1972), General semantics, in Donald DAVIDSON and Gilbert HARMAN, editors, *Semantics of Natural Language*, pp. 169–218, Springer.
- Mike LEWIS and Mark STEEDMAN (2014), A* CCG parsing with a supertag-factored model, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 990–1000.
- Godehard LINK (1983), The logical analysis of plurals and mass terms: A lattice-theoretic approach, in Paul PORTNER and Barbara H. PARTEE, editors, *Formal Semantics – the Essential Readings*, pp. 127–147, Blackwell.
- Yinhan LIU, Myle OTT, Naman GOYAL, Jingfei DU, Mandar JOSHI, Danqi CHEN, Omer LEVY, Mike LEWIS, Luke ZETTLEMOYER, and Veselin STOYANOV (2019), RoBERTa: A robustly optimized BERT pretraining approach, *arXiv preprint arXiv:1907.11692*.

Marco MARELLI, Stefano MENINI, Marco BARONI, Luisa BENTIVOGLI, Raffaella BERNARDI, and Roberto ZAMPARELLI (2014), A SICK cure for the evaluation of compositional distributional semantic models, in *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pp. 216–223.

Pascual MARTÍNEZ-GÓMEZ, Koji MINESHIMA, Yusuke MIYAO, and Daisuke BEKKI (2016), ccg2lambda: A compositional semantics system, in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL), System Demonstrations*, pp. 85–90.

Pascual MARTÍNEZ-GÓMEZ, Koji MINESHIMA, Yusuke MIYAO, and Daisuke BEKKI (2017), On-demand injection of lexical knowledge for recognising textual entailment, in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pp. 710–720.

Tom MCCOY, Ellie PAVLICK, and Tal LINZEN (2019), Right for the wrong Reasons: diagnosing syntactic heuristics in natural language inference, in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 3428–3448.

George A. MILLER (1995), WordNet: A lexical database for English, *Communications of the ACM*, 38(11):39–41.

Koji MINESHIMA, Pascual MARTÍNEZ-GÓMEZ, Yusuke MIYAO, and Daisuke BEKKI (2015), Higher-order logical inference with compositional semantics, in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 2055–2061.

Richard MONTAGUE (1970), Universal grammar, *Theoria*, 36(3):373–398.

Glyn MORRILL and Oriol VALENTÍN (2016), Computational coverage of type logical grammar: The Montague test, *Empirical Issues in Syntax and Semantics*, 11:1–30.

Marcin MORZYCKI (2016), *Modification*, Cambridge University Press.

Katsuma NARISAWA, Yotaro WATANABE, Junta MIZUNO, Naoaki OKAZAKI, and Kentaro INUI (2013), Is a 204 cm man tall or small? Acquisition of numerical common sense from the web, in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 382–391.

Terence PARSONS (1990), *Events in the Semantics of English*, MIT Press.

Barbara H. PARTEE (1988), Many quantifiers, in *Proceedings of the 5th Eastern States Conference on Linguistics (ESCOL)*, pp. 383–402.

Barbara H. PARTEE (2007), Compositionality and coercion in semantics: The dynamics of adjective meaning, in Gerlof BOUMA *et al.*, editors, *Cognitive Foundations of Interpretation*, pp. 145–161.

- Sandro PEZZELLE and Raquel FERNÁNDEZ (2019), Is the red square big? MALeViC: Modeling adjectives leveraging visual contexts, in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2865–2876.
- Stephen G. PULMAN (2007), Formal and computational semantics: A case study, in Jeroen GEERTZEN, Elias THIJSSSE, Harry BUNT, and Amanda SCHIFFRIN, editors, *Proceedings of the Seventh International Workshop on Computational Semantics: IWCS-7, Tilburg, The Netherlands, 2007*, pp. 181–196.
- Stephen G. PULMAN (2018), Second order inference in natural language semantics, *Journal of Language Modelling*, 6(1):1–40.
- Jessica RETT (2018), The semantics of *many*, *much*, *few*, and *little*, *Language and Linguistics Compass*, 12(1):e12269.
- John Robert ROSS (1970), Gapping and the order of constituents, in Manfred BIERWISCH and Karl E. HEIDOLPH, editors, *Progress in Linguistics*, pp. 249–259, De Gruyter Mouton.
- Roger SCHWARZSCHILD (2008), The semantics of comparatives and other degree constructions, *Language and Linguistics Compass*, 2(2):308–331.
- Pieter A. M. SEUREN (1973), The comparative, in Ferenc KIEFER and Nicolas RUWET, editors, *Generative Grammar in Europe*, pp. 528–564, Riedel.
- Benjamin SPECTOR (2013), Bare numerals and scalar implicatures, *Language and Linguistics Compass*, 7(5):273–294.
- Mark STEEDMAN (1996), *Surface Structure and Interpretation*, MIT Press.
- Mark STEEDMAN (2000), *The Syntactic Process*, MIT Press.
- Geoff SUTCLIFFE (2017), The TPTP problem library and associated infrastructure, *Journal of Automated Reasoning*, 59(4):483–502.
- Geoff SUTCLIFFE, Stephan SCHULZ, Koen CLAESSEN, and Peter BAUMGARTNER (2012), The TPTP typed first-order form with arithmetic, in Nikolaj BJØRNER and Andrei VORONKOV, editors, *Logic for Programming, Artificial Intelligence, and Reasoning*, pp. 406–419, Springer.
- Anna SZABOLCSI (2010), *Quantification*, Cambridge University Press.
- Johan VAN BENTHEM (1986), *Essays in Logical Semantics*, Springer.
- Robert VAN ROOIJ and Katrin SCHULZ (2004), Exhaustive interpretation of complex sentences, *Journal of Logic, Language and Information*, 13(4):491–519.
- Dag WESTERSTAÅHL (2007), Quantifiers in formal and natural languages, in Dov M. GABBAY and Franz GUENTHNER, editors, *Handbook of Philosophical Logic*, volume 14, pp. 223–338, Springer.

Implementing natural language inference for comparatives

Adina WILLIAMS, Nikita NANGIA, and Samuel BOWMAN (2018), A broad-coverage challenge corpus for sentence understanding through inference, in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp. 1112–1122.

Yadollah YAGHOOBZADEH, Remi TACHET, Timothy J. HAZEN, and Alessandro SORDONI (2019), Robust natural language inference models with example forgetting, *arXiv preprint arXiv:1911.03861*.

Hitomi YANAKA, Koji MINESHIMA, Daisuke BEKKI, Kentaro INUI, Satoshi SEKINE, Lasha ABZIANIDZE, and Johan BOS (2019), Can neural networks understand monotonicity reasoning?, in *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 31–40.

Masashi YOSHIKAWA, Hiroshi NOJI, and Yuji MATSUMOTO (2017), A* CCG parsing with a supertag and dependency factored model, in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 277–287.

Izumi Haruta

Ochanomizu University, Japan

Daisuke Bekki

0000-0002-9988-1260
bekki@is.ocha.ac.jp

Ochanomizu University, Japan

Koji Mineshima

0000-0002-2801-9171
minesima@abelard.flet.keio.ac.jp

Keio University, Japan

Izumi Haruta, Koji Mineshima, and Daisuke Bekki (2022), *Implementing Natural Language Inference for comparatives*, *Journal of Language Modelling*, 10(1):139–191

<https://dx.doi.org/10.15398/jlm.v10i1.294>

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

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

