Journal of Language Modelling

VOLUME 7 ISSUE 1
JUNE 2019

Institute of Computer Science
Polish Academy of Sciences
Warsaw

Journal of Language Modelling

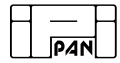
VOLUME 7 ISSUE 1
JUNE 2019

Overviews

Articles

JOURNAL OF
LANGUAGE MODELLING

ISSN 2299-8470 (electronic version)
ISSN 2299-856X (printed version)
http://jlm.ipipan.waw.pl/



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

Circulation: 100 + print on demand

Layout designed by Adam Twardoch.
Typeset in XeLaTeX using the typefaces: *Playfair Display*
by Claus Eggers Sørensen, *Charis SIL* by SIL International,
JLM monogram by Łukasz Dziedzic.

EDITORIAL BOARD

A practitioner's view: a survey and comparison of lemmatization and morphological tagging in German and Latin

Rüdiger Gleim[1], *Steffen Eger*[2], *Alexander Mehler*[1], *Tolga Uslu*[1], *Wahed Hemati*[1], *Andy Lücking*[1], *Alexander Henlein*[1], *Sven Kahlsdorf*[1], *and Armin Hoenen*[1]
[1] Text Technology Lab, Goethe University Frankfurt, Germany
[2] Ubiquitous Knowledge Processing Lab, Technische Universität Darmstadt, Germany

ABSTRACT

The challenge of POS tagging and lemmatization in morphologically rich languages is examined by comparing German and Latin. We start by defining an NLP evaluation roadmap to model the combination of tools and resources guiding our experiments. We focus on what a practitioner can expect when using state-of-the-art solutions. These solutions are then compared with old(er) methods and implementations for coarse-grained POS tagging, as well as fine-grained (morphological) POS tagging (e.g. `case`, `number`, `mood`). We examine to what degree recent advances in tagger development have improved accuracy – and at what cost, in terms of training and processing time. We also conduct in-domain vs. out-of-domain evaluation. Out-of-domain evaluation is particularly pertinent because the distribution of data to be tagged will typically differ from the distribution of data used to train the tagger. Pipeline tagging is then compared with a tagging approach that acknowledges dependencies between inflectional categories. Finally, we evaluate three lemmatization techniques.

Keywords: morphological tagging, lemmatization, morphologically rich languages, NLP evaluation modeling

1 INTRODUCTION

Lemmatization and part-of-speech (POS) tagging are critical preprocessing steps for many natural language processing (NLP) tasks, such as information retrieval, knowledge extraction, and semantic analysis. In morphologically rich languages such as German and Latin, both processes are non-trivial due to the variability of lexical forms. This results in large tagsets for both coarse-grained and fine-grained (morphological) POS tagging – including inflectional categories such as `case`, `gender`, and `degree` in addition to coarse-grained POS labels – and a large number of (potentially unseen) forms associated with each lemma. In this work, we survey tagging and lemmatization techniques for German and Latin, using corpora that allow us to analyse the effects of NLP between genres (Tiger vs. TGermaCorp), and also between periods (Capitularies vs. Proiel). Our survey includes both older tools, such as the TreeTagger (Schmid 1994) and TnT (Brants 2000), and more modern approaches to tagging and lemmatization. Even though we expect technology to improve steadily over time, it is not always easy to quantify the gap between older and more modern approaches, or to rank the most recent generation of systems in order of efficiency. We test our systems under the following conditions and requirements:

- We train lemmatization and tagging independently because the methods studied are designed for separate use.[1] We then focus on the impact of varying parameters, supplementary resources, and a combination of tools.
- Ideally, we want a learned system to perform well on the data on which it has been trained (*in-domain* (ID): a specific text genre, historical language variant, etc.) but also to perform adequately on similar corpora (*out-of-domain* (OD): with similar but different genres, registers, language varieties, etc.).
- Since coarse-grained POS tagging alone may be insufficient for linguistic applications and unsatisfactory for practitioners, we expect a system to perform reasonably well on fine-grained POS tagging.

[1] LemmaTag (Kondratyuk *et al.* 2018) is an exception that natively supports joint lemmatization and tagging.

- As run-times of systems may be of considerable interest for practitioners, we include both training and testing time estimates for each technique.

Section 2 provides a systematization grid for NLP applications and their evaluation. This grid is mapped on to our evaluation objectives, thereby offering both a general evaluation model and a roadmap for the subsequent experimental sections. In Section 3, we describe three approaches to lemmatization, followed in Section 4 by ten tools for tagging, which form the basis of our experiments. Section 5 introduces all resources used in the experiments, primarily the corpora used for training and evaluation, with a discussion of other lexical resources and methods of computing word embeddings. Section 6 describes the experiments used to test lemmatization and POS– as well as fine-grained POS tagging. In Section 7, we discuss our findings, while Section 8 provides a summary of this study and prospects for future work.

2 NLP EVALUATION ROADMAP

Taggers cannot be compared or even applied *in vacuo*; the minimum requirements for NLP taggers are a tagset and a target text of some natural language. Similar dependencies apply to virtually all NLP applications. In order to systematize such relationships and make them transparent for readers of NLP-related work, we provide an interrelationship model in Figure 1. The tree structure on the left-hand side of the model presents NLP tasks and resources, followed by instantiating parameters, which are then mapped on to evaluation objectives. The dashed lines in Figure 1 indicate partial use of a parameter set with respect to an objective. Thus, the systematization grid explicates the requirements to be met in order to perform NLP experimentations. Accuracy values for such experiments assess how adequately relevant objectives have been attained. These objective-to-accuracy mappings constitute the right-hand side of Figure 1 (column "A", values given in %). We chose the maximum accuracy achieved as target value for the scale (which usually lies in the range between 96% and 98%). We also indicate which section presents the relevant evaluation studies. The systematization grid thus presents a general dependency model of NLP tasks and resources, as a roadmap for the current paper.

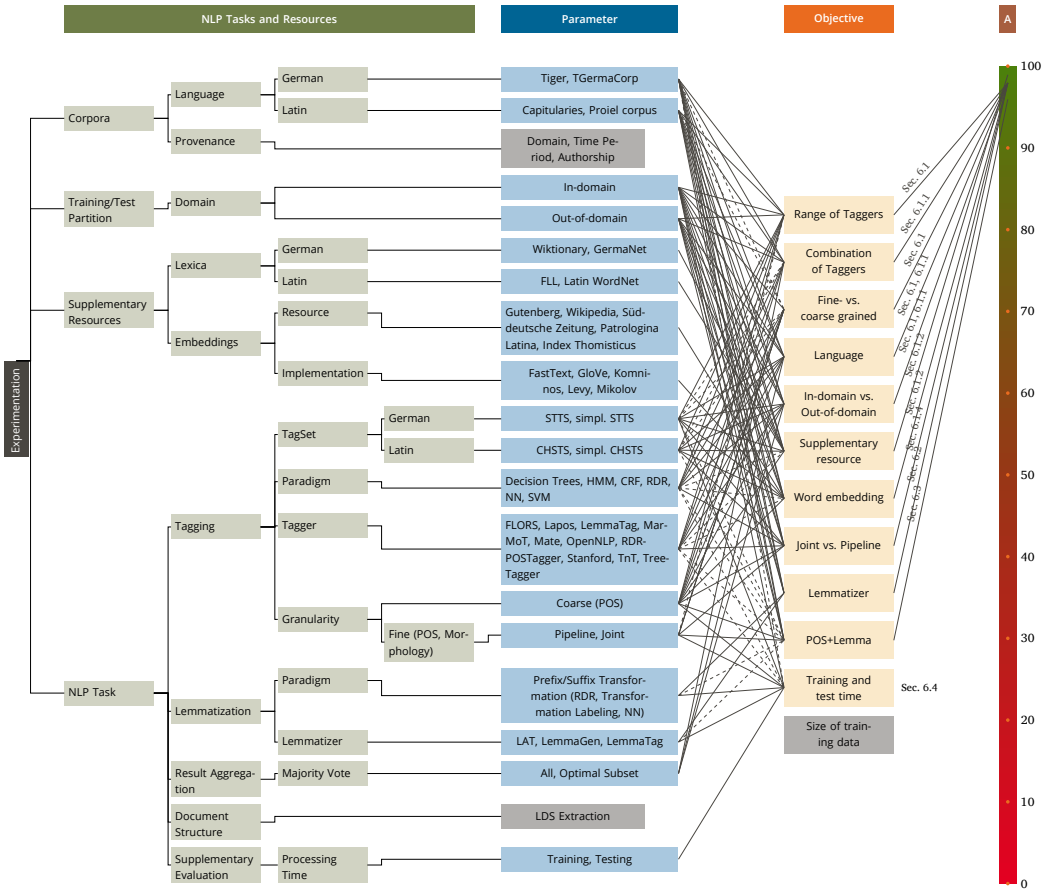


Figure 1: Roadmap for NLP evaluation. See main text for details. The tree on the left-hand side provides a systematization of tasks and resources for NLP. The leaf nodes ("Parameter" column) collect the parameter settings we employ to instantiate the relevant cells of the systematization grid. Parameters that are not part of the current paper are shaded in gray. The parameters are mapped in different ways on to the evaluation objectives, as indicated by connecting lines. Dashed lines indicate that the parameter instantiation in question is not fully exhausted by the target evaluation objective. The scale on the right-hand side depicts the maximum accuracy of the evaluation results for each objective, and indicates the section where the relevant evaluation study is described

3 LEMMATIZATION

We view *lemmatization* as the problem of transforming a word form into its canonical form, or lemma. In a machine-learning context, lemmatization has sometimes been considered as a character-level string transduction process (Dreyer *et al.* 2008; Nicolai *et al.* 2015; Eger 2015; Schnober *et al.* 2016), a prefix and suffix transformation problem (Juršič *et al.* 2010; Gesmundo and Samardzic 2012), or as a pattern-matching task (Durrett and DeNero 2013; Hulden *et al.* 2014).[2] While character-level string transducers may yield excellent results (Nicolai *et al.* 2015), particularly when trained and tested on lists of words randomly extracted from a lexicon (Eger 2015), they tend to be learned slower, and typically do not lemmatize in context, but consider the lemmatization problem in isolation, ignoring contextual word-form cues. In addition, we found in preliminary experiments that, for real-world lemmatization, where distribution is marked by many irregular forms, simpler prefix and suffix transformation systems may be competitive with more sophisticated string transducers.

In this work, we experiment with three approaches to lemmatization, two of which are based on prefix and suffix transformations, and one on neural networks. These experiments are presented in Section 6.2. **LemmaGen** (Juršič *et al.* 2010) learns 'ripple down rules' (Compton and Jansen 1988), that is, tree-like decision structures, from pairs of strings. Rule conditions are suffixes of word forms, and rule consequents are transformations that replace the suffix in question by a new suffix. The second approach we experiment with is the casting of lemmatization as a classification task (Gesmundo and Samardzic 2012), which we call **LAT**: lemmatization is viewed here as a 4-tuple indicating the prefix and suffix transformations involved in the lemmatization process. For example (see Figure 2), the transformation of the German verb form *gespielt* into its lemma *spielen* is encoded by the

[2] Lemmatization can also be implemented with the help of a lexicon. However, lexicons are hard to acquire, and their performance is comparatively low in cases where they do not sufficiently discriminate the distributions of polysemous lemmas in large corpora of real texts. Nevertheless, a lexicon could typically 'assist' a learned system, e.g., via features that trigger if a form occurs in the lexicon (e.g., in a similar manner to that outlined here).

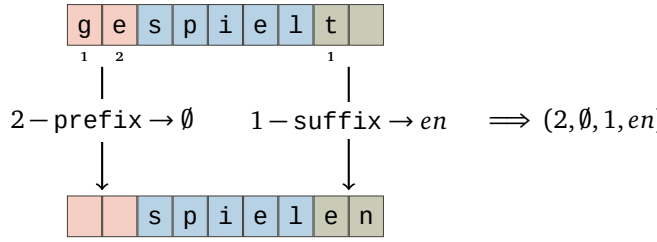


Figure 2: Example of how to represent lemmatization for *gespielt* to *spielen* as a tuple

tuple $(2, \emptyset, 1, en)$, indicating that to derive *spielen*, the first two characters of *gespielt* are replaced by the empty string, and the last character is replaced by *en*. This compact encoding considers lemmatization as a classification problem where the size of the output space is relatively small (some hundreds or thousands of labels, at most). Moreover, lemmatization can then also be treated as a sequence labeling problem, where dependency between subsequent labels may be taken into account. One may argue that inflections in German are rich and that modifications may also include central characters, thus replacing the entire word in extreme cases. Nonetheless, this approach can be applied as long as the tagger can handle the output space.

Finally we include LemmaTag (Kondratyuk *et al.* 2018), which is based on neural networks. It has primarily been included in experiments for POS tagging (see Section 4), but it is also interesting to work with as it supports joint lemmatization and tagging.

4 POS TAGGING

Among the milestones in POS tagging (or sequence labelling) are: including *dependencies* between output labels (as in Markov models such as HMMs or CRFs); the broad use of lexical *features* (Ratnaparkhi 1996; Toutanova *et al.* 2003); and the concept of the *margin* introduced in SVMs. The most recent class of taggers is characterized by several possibilities: that of including *word representations* learned from *unlabeled* data, that of applying feature-rich models to problems with large output spaces, and that of making use of *deep* (rather than *shallow*) models such as neural networks that can in addition function without handcrafted features.

In this work, we consider the following POS tagging systems, which are listed in order of their year of publication. The **TreeTag-**

ger (Schmid 1994), a popular tagging system until recently, is based on decision trees. As such, it cannot account for dependencies between output (tag) labels. **TnT** (Brants 2000) implements a trigram Hidden Markov tagger with a module for handling unknown words. It has been shown to perform as well as maximum entropy models. The **Stanford tagger** (Toutanova *et al.* 2003) implements a bidirectional log-linear model that makes broad use of lexical features. The implementation lets the user specifically activate and deactivate desired features. **Lapos** (Tsuruoka *et al.* 2011) is a 'history-based' tagging model (this model class subsumes maximum entropy Markov models) incorporating a lookahead mechanism into its decision-making process. It has been reported to be competitive with globally optimized models such as CRFs and structured perceptrons. **Mate** (Bohnet and Nivre 2012) has been introduced as a transition-based system for joint POS tagging and dependency parsing. We also include the **OpenNLP** tagger, an official Apache project.[3] For these systems, we refer to the original works for more in-depth descriptions.

Among the most recent generation of taggers, we consider **MarMoT** (Müller *et al.* 2013), which implements a higher order CRF with approximations such that it can deal with large output spaces. In addition, MarMoT can be trained to fire on the predictions of lexical resources as well as on *word embeddings*, real vector-valued representations of words.[4]

The **RDRPOSTagger** (Nguyen *et al.* 2014) implements an error-driven approach to POS tagging by constructing a tree of single classification ripple down rules (SCRDR). It takes a gold standard *and* an automatically tagged version thereof as input in order to generate rules to reflect any differences. By default, RDRPOSTagger uses a built-in lexicon-based tagger, which by itself is not very accurate, but learning exception rules from the initial tagging gives promising results. Note that, since the approach is based on the concept of correcting the output of some initial tagging, the initial tagger is required for both training and testing/tagging. Nguyen *et al.* (2016) show that using an external tagger (e.g. TnT) over the built-in (lexicon-based)

[3] See `https://opennlp.apache.org/`.

[4] Another morphological tagger which is based on conditional random fields is TLT-CRF (vor der Brück and Mehler 2016) which we could not include due to time and space constraints.

tagger yields better results. In this work, we replicate their experiment by optionally using TnT, and extend it by using MarMoT as the initial tagger. Note that, in order to achieve a tagged version of the training set for our experiments, we use 10-fold jack-knifing: each fold is tagged by TnT or MarMoT based on the remaining folds. Taking a practitioner's view, we use the built-in tagger for most of our experiments. However, we also examine the benefits of using an external tagger for POS.

FLORS (Schnabel and Schütze 2014) tags a given word by constructing a feature vector representation of its local context and then classifying this vector by an SVM. The feature vector representation of each word in a context includes distributional, shape, and suffix information, and the feature vector for the entire context is the concatenation of the word vector representations. Note that the implementation of FLORS includes language-specific features (for English). This is expected to decrease its performance on the Latin and German datasets we consider. In principle, the vector representations of words are the same for known and unknown words, thus FLORS is potentially very well-suited for OD tasks. In our work, we use online FLORS (Yin *et al.* 2015), which incrementally updates word representations for each new test sentence encountered.

We also wanted to include **NonLexNN** (Labeau *et al.* 2015), a non-lexicalized neural network architecture for POS tagging. By operating on the subword/character-level, it promises to yield higher performance on OD tasks, similarly to the FLORS tagger. However, we could not make this tagger perform on-par with the other taggers surveyed. One reason for this was its very lengthy runtime – several days for a single training fold – so that we could not sufficiently experiment with its parameters.

A neuronal network approach based on bidirectional long short-term memory recurrent neural networks (Wang *et al.* 2015) was implemented using Deeplearning4j. It consists of one Graves Bidirectional Long-Short Term Memory (BLSTM) Layer, and one RNNOutputLayer with a Softmax activation function and MultiClass Cross Entropy (MCXENT) Loss function. The BLSTM Layer uses a TANH activation function, and we changed the updater to ADAGRAD, to improve learning for rare POS tags. Following Wang *et al.* (2015), we represent each word with its word embedding, and add a three-dimensional binary

	User-def. features	Large output spaces	External resources	Label dependencies
FLORS		✓	✓	
Lapos				✓
LemmaTag		✓		✓
MarMoT	✓	✓	✓	✓
Mate		✓		✓
OpenNLP		✓		✓
RDRPOSTagger		✓		✓
Stanford	✓			✓
TnT		✓		✓
TreeTagger		✓		

Table 1: Systems and selected properties

vector to retain information about capitalization: initial upper case, all upper case, or all lower case. We add another vector of 3 dimensions to identify the word ending. Note that this approach could not be fully exploited during our experiments, because of limited hardware resources and the amount of time needed to train the networks.

Finally, we include **LemmaTag** (Kondratyuk *et al.* 2018), a featureless neural network approach to joint lemma and POS tagging, based on bidirectional memory recurrent neural networks. It uses character- and word-level embeddings. LemmaTag is based on Tensorflow, a library for dataflow programming widely used in machine learning applications. Since it benefits from GPUs, we run LemmaTag on a GPU workstation so as to run all experiments in reasonable time.

In Table 1, we list some key properties of the taggers in the survey. While most models make use of features (except for HMMs as TnT is based on, for which the inclusion of arbitrary features is non-trivial), not all of them allow users to specify user-defined features. Therefore, we had to exclude Lapos and Stanford from some experiments (e.g. joint-tagging) as they do not scale well on large output spaces.

In addition to the list of taggers, we include a **majority vote** POS tagger. By examining the results of at least three taggers, we identify the POS tag for a given token with maximum tagger agreement. In order to solve ties, taggers are ranked by date of publication. In practice, tagger accuracy should be estimated on a held-out set, extracted from the training data, or on a (small) hand-annotated data set. The majority vote tagger is used to assess whether tagging system errors

correlate. The assumption is that majority voting does not work when all systems commit the same types of errors.

5 DATASETS

In this work, we examine the performance of NLP tools on Latin and German texts. We distinguish between in-domain (ID) and out-of-domain (OD) experiments. From a machine learning perspective, ID experiments are more well-defined, because they are generally performed on a corpus of texts from the same era and genre. More importantly, the gold standard of such corpora has usually been created by a closed group of trained annotators who agreed upon a specific annotation manual. Thus, we can expect a high degree of coherency in terms of the primary content as well as the annotation.

For ID experiments the data must be partitioned into distinct training and test sets. In general, we perform a 3-fold random subset validation, with a 90%/10% split on each corpus for each language. In contrast, OD experiments involve two corpora, with one corpus used entirely for training, and the other one for testing. For the LemmaTag tagging and lemmatization tool, we require an additional development set. For these ID experiments, we use a 3-fold 80%/10%/10% split. For the OD experiments, we use 90% of the entire source corpus as the training set and the remaining 10% as the development set. As before, the entire target corpus is used for testing and the 90%/10%-split is performed three times. Since OD experiments bring together corpora that may vary in terms of the era in which they were written, in the genres they cover, and in the standards by which they were annotated, we expect a significant drop in accuracy when we evaluate NLP tools, as has been documented by much previous research (Müller *et al.* 2015; McGillivray *et al.* 2009). Nonetheless, OD scenarios provide a much more realistic perspective on the performance of lemmatization and POS tagging, since a practitioner usually has to rely on pre-trained models. This is not primarily because of the technical skills required to train a model, but rather because of the huge effort required to construct a training set large enough to accurately cover the desired genre.

In the following subsections, we describe the corpora as well as supplementary resources used in this work for German and Latin. Of

Corpus	Language	Sentences	Tokens
Tiger	German	50,472	888,238
TGermaCorp	German	8,941	157,210
Capitularies	Latin	15,572	481,578
Proiel	Latin	1,147	22,280

Table 2:
Statistics of corpora used in the experiments

the taggers we consider, MarMoT can benefit from additional lexical resources as well as word embeddings. In order to examine the impact of supplementary data, we evaluate various lexical resources and different corpora, as well as algorithms to compute word embeddings.

5.1 *Corpora*

For German, we train and test on the Tiger corpus (Brants *et al.* 2004), and on TGermaCorp (Lücking *et al.* 2016). The Tiger corpus consists of newspaper articles from the German "Frankfurter Rundschau" which were semi-automatically lemmatized and tagged for POS and morphology. For Latin, we use the Capitularies corpus (Mehler *et al.* 2015; Eger *et al.* 2015) and the Proiel corpus (Haug and Jøhndal 2008). The Capitularies corpus is based on the "Capitvlaria regvm Francorvm, ed. Alfredus Boretius (Hannover 1897)". The two Latin corpora stem from different genres *and* different epochs, making them interesting candidates for OD tagging experiments. The Capitularies consist of instructions and directives from the Merovingian and Carolingian periods (600–900 AD), whereas Proiel consists of classical and Christian texts (100 BC–500 AD). We use a random subset of Proiel, for which tag labels have been manually synchronized with those of the Capitularies. As the Proiel corpus does not contain punctuation, we expect low accuracy when it participates in OD scenarios. This problem should not occur with the Capitularies corpus, which contains punctuation. Table 2 gives an overview of the corpora used in the experiments.

For this study, the thematic range of TGermaCorp, which is composed of literary texts in standard German, was extended to include language of science, from a diachronic perspective. Extracts were selected from two texts from the second half of the nineteenth century: Friedrich Nietzsche's *Der Antichrist* (1894), representing humanities, and Gregor Mendel's *Versuche über Pflanzenhybriden* (1866), representing the natural sciences. Nietzsche's text was obtained from the *Digitale*

Kritische Gesamtausgabe Werke und Briefe.[5] Mendel's text was obtained from *Project Gutenberg*.[6] The token count for *Der Antichrist* is 30,652 and for *Versuche über Pflanzenhybriden* it is 20,129.

Each text was divided into 12 equal chunks which were paired to form 12 mixed files for annotation. Each annotator had to annotate and occasionally correct the tokenization of a mixed file of 4,831 tokens. Eight annotators performed (coarse-grained) POS tagging and lemmatization, following STTS annotation guidelines (Schiller *et al.* 1999) – see Lücking *et al.* (2016) for further details.

Before annotating the TGermaCorp extension, annotators underwent a five-week training period, supervised by two linguistically trained experts. The objective of the training period was to familiarize annotators with the tagset, so as to achieve consistent annotation. The first part of the training period focused on the rules and labels for lemmatization, using the annotation manuals and comparing results with gold-standard annotations (one week). During the second part of the training period (four weeks), the annotators had to complete two annotation tasks. Their results were inspected on a sample basis by the two supervisors. The findings as well as any examples worthy of discussion encountered by the annotators, were discussed in weekly meetings, in order to clarify annotation rules or agree on conventions in cases not unequivocally covered by the manuals. Annotators who successfully completed the training period then annotated the TGermaCorp extension.[7]

In order to test annotator self-agreement, 300 tokens from each text snippet were extracted and added at the end of the annotation file, so that 600 tokens were annotated twice by each annotator. Intra-rater agreement was calculated on these two annotations for each annotator by means of Cohen's Kappa (Cohen 1960). The results are presented in Table 3. The perfect agreement for ann7 is due to her recognizing the double annotation task and reconstructing her previous choices.

An inter-rater agreement study was also carried out. Four annotators annotated a set of 100 tokens from Nietzsche's *Antichrist*, while the other four annotators annotated a set of 100 tokens from Mendel's

[5] Provided at `http://www.nietzschesource.org` (CC BY-NC 3.0).

[6] `http://www.gutenberg.org/cache/epub/40854/pg40854.txt`

[7] Although twelve annotators were trained, only eight successfully completed the training period.

Annotator	Lemma	POS
annotator 1	0.98	0.98
annotator 2	0.99	0.99
annotator 3	0.98	0.96
annotator 4	0.99	0.99
annotator 5	0.98	0.97
annotator 6	0.96	0.94
annotator 7	1.00	1.00
annotator 8	0.97	0.92

Table 3:
Intra-rater agreement: summary

Level	All	ann1-gs	ann2-gs	ann3-gs	ann4-gs
lemma	0.92	0.91	0.94	0.97	0.88
POS	0.85	0.96	0.94	0.88	0.85

Table 4:
Inter-rater agreement: summary for the Nietzsche extract

Level	All	ann5-gs	ann6-gs	ann7-gs	ann8-gs
lemma	0.95	0.95	0.85	0.94	0.95
POS	0.91	0.90	0.89	0.89	0.90

Table 5:
Inter-rater agreement: summary for the Mendel extract

Versuche über Pflanzenhybriden. These two sets of tokens had previously been annotated and discussed by two linguistically trained annotators, thus providing a gold standard for comparison. Agreement values for the Nietzsche extract are given in Table 4, and those for the Mendel extract are given in Table 5. Column 'all' presents agreement values among all four annotators in terms of Fleiss' generalized Kappa (Fleiss 1971). The remaining columns provide each annotator's compliance with the Gold Standard (Cohen's Kappa). With all agreement scores exceeding the threshold of 0.81, annotations can be regarded as 'sound' (Krippendorff 1980) or 'almost perfect' (Rietveld and van Hout 1993).

As self-consistency (intra-rater) is even higher than mutual consistency (inter-rater) – although both sets of values are satisfactory – and as more annotators were involved in annotating the extension than the original corpus, slightly more diverse annotation values are expected for the extension. We therefore expect a slight decrease in tagger performance for the newly added snippets.

5.2 *Tagsets*

The choice of a tagset depends mainly on the language to be annotated. For German NLP, the Stuttgart-Tübingen TagSet (STTS) is widely recommended. It is used in the Tiger corpus and has been adopted for the annotation of TGermaCorp. Consequently, STTS is used in this work to train and evaluate POS tagging. However, there are some cases in linguistic studies where for a certain type of word, such as a verb, it is not relevant to differentiate between sub-groups, such as finite or non-finite. Another argument in favor of simplification is that training effort for annotators will be reduced. Instead of simply training our taggers for STTS and grouping results into a more coarse-grained tagset, we also investigated the potential accuracy of a simplified tagset. We therefore performed experiments based not only on the STTS but also on the simplified tagset, sSTTS.

Our experiments on Latin texts are primarily based on the Capitularies corpus, which was tagged using the *Computational Historical Semantics TagSet* (CHSTS).[8] Compared to STTS, the CHSTS is coarse grained, and does not distinguish between different types of verbs, adjectives, or pronouns. It does, however, distinguish nouns from named persons and named entities. We therefore expect further simplification not to have as great an impact on the accuracy of POS tagging as that observed with the simplified STTS. Consequently, we also used a simplified tagset sCHSTS in our experiments, which maps nouns (NN), persons (NP), and named entities (NE) to one single POS tag (N). Table 6 maps simplified equivalents to STTS and CHSTS.

The tagsets discussed so far encode morphological information to some extent (e.g. VVFIN vs. VVIMP) but do not explicate morphological categories as a whole. We therefore differentiate between *coarse-grained* and *fine-grained* (morphological) POS tagging in our experiments. Table 7 provides an overview of morphological tags. Categories or tags that are unique either to Latin ($\cdot^{la}$) or German ($\cdot^{de}$) corpora are marked accordingly. The Tiger corpus, the Capitularies and the Proiel corpus provide annotations of `case`, (comparison) `degree`, `gender`, `mood`, `number`, `person` and `tense`. In addition, the Latin texts are an-

[8] This tagset was developed within the framework of the Computational Historical Semantics project (`http://www.comphistsem.org/`) (Jussen *et al.* 2007).

Table 6: Mapping simplified tagsets for STTS and CHSTS

	sSTTS	STTS	sCHSTS	CHSTS
adjective	ADJ	ADJA, ADJD	ADJ	ADJ
adposition	AP	APPO, APPR, APPRART, APZR	AP	AP
adverb	ADV	ADV, PAV	ADV	ADV
article	ART	ART		
card. num.	CARD	CARD	NUM	NUM
conjunction	KON	KOKOM, KON, KOUI, KOUS	CON	CON
dist. num.			DIST	DIST
foreign	FM	FM	FM	FM
interjection	ITJ	ITJ	ITJ	ITJ
non-word	XY	XY	XY	XY
noun	N	NN, NE	N	NN, NE, NP
particle	PTK	PTKZU, PTKNEG, PTKVZ, PTKA, PTKANT	PTC	PTC
pronoun	P	PDAT, PDS, PIAT, PIS, PPER, PPOSAT, PPOSS, PRELAT, PRELS, PRF, PWAT, PWAV, PWS, PIDAT	PRO	PRO
truncation	TRUNC	TRUNC		
verb	V	VAFIN, VAIMP, VAINF, VAPP, VMFIN, VMINF, VMPP, VVFIN, VVIMP, VVINF, VVIZU, VVPP	V	V
pun. term.	$.	$.	$.	$.
comma	$,	$,	$,	$,
other pun.	$(	$(	$(	$(

Category	Tags
case	$*^{de}$, ablative^{la}, accusative, dative, genitive, locative^{la}, nominative, vocative^{la}
degree	$*^{de}$, comparative, positive, superlative
gender	$*^{de}$, feminine, masculine, neuter
mood	gerund^{la}, gerundive^{la}, imperative, indicative, infinitive^{la}, participle^{la}, subjunctive, supine^{la}
number	$*^{de}$, plural, singular
person	1, 2, 3
tense	future^{la}, future perfect^{la}, imperfect^{la}, past^{de}, perfect^{la}, pluperfect^{la}, present
voice^{la}	active^{la}, passive^{la}

Table 7: Morphological tags used for fine-grained POS tagging. The $\cdot^{la}$ tags are only used for Latin texts, whereas $\cdot^{de}$ tags are only used for the German Tiger corpus

notated with either active or passive `voice`. Note that tags common to both corpora can be mapped directly, even if they are not identical in appearance (e.g. “acc” in Tiger and “accusative” in the Latin texts). The morphological annotation of the two Latin corpora is much richer than that of the Tiger corpus.

5.3 *Lexicons*

For German, we extracted a lexicon from the German Wiktionary.[9] Extracting lemmas and syntactic words (including all grammatical categories available) from a Wiktionary instance in a thorough *and* robust way is not a trivial task. Even though guidelines and templates exist, they differ significantly between Wiktionary instances, and also vary in the way they are used within the same language. Our approach parses the HTML code of a Wiktionary instance that has been setup on a local server using the XML-dump from 2015-09-01.[10] This is, in our experience, more accurate than trying to parse MediaWiki sources directly, and it saves bandwidth on the official Wiktionary servers. We also used GermaNet version 11.0 (Hamp and Feldweg 1997; Henrich and Hinrichs 2010) as a lexical resource. In both cases, we are limited to the simplified STTS tagset, since the lexicons extracted to not have sufficient information to differentiate between different types of verbs, as STTS does. Future work may however include an STTS-compliant extraction and mapping of Wiktionary.

For Latin, we made use of the Frankfurt Latin Lexicon (FLL) (Mehler *et al.* 2015). As an equivalent to GermaNet, we included the lexical resources of a Latin WordNet (Minozzi 2008), available under an Attribution-ShareAlike 4.0 license.

For MarMoT, we used information about word forms and their POS. Table 8 gives an overview of the lexical resources used in the experiment.

5.4 *Embeddings*

Besides lexicons, word embeddings can be used as an additional resource to train specific taggers like MarMoT. In order to achieve reliable representations of word forms in a vector space, large corpora

[9] `http://de.wiktionary.org`

[10] `https://dumps.wikimedia.org/dewiktionary/20150901/`

Lexicon	Language	Tagset	Word forms
DE-Wiktionary	German	sSTTS	381,296
GermaNet	German	sSTTS	126,392
GermanAll	German	sSTTS	463,912
FLL	Latin	CHSTS	3,635,245
FLL	Latin	sCHSTS	3,631,179
LatinWordNet	Latin	sCHSTS	9,124
LatinAll	Latin	sCHSTS	3,631,199

Table 8:
Statistics of lexica used in the experiments

are required. We considered three corpora: The Gutenberg-DE Edition 13, the German Wikipedia, and the German newspaper "Süddeutsche Zeitung" (SZ). All corpora were tokenized using the PTBTokenizer contained in StanfordCoreNLP (Manning *et al.* 2014), lemmatized and tagged using MarMoT, and dependency parsed using Mate (Bohnet and Nivre 2012). All these tools and the corresponding processing pipeline are available (also as web-services) via the TextImager system (Hemati *et al.* 2016).

The Gutenberg-DE Edition 13 is a collection of classical German literature ranging from modern works back to a poem by Walther von der Vogelweide written in 1198. In contrast, the German Wikipedia corpus covers articles of the online encyclopedia which were extracted from a dump dating from 2016-02-03. The articles were parsed using Sweble (Dohrn and Riehle 2011) and converted into TEI P5. Finally, the newspaper corpus covers 23 volumes of the "Süddeutsche Zeitung" between 1992 and 2014. In our experiments, we use each corpus separately and all corpora combined (German-All).

For Latin, we use texts ranging from the 2nd to the 14th century. The documents stem from the Patrologia Latina (PL) corpus, the Monumenta Germaniae Historica, and the Central European Medieval Texts Series.[11] [12] [13] Most of the texts are from the 9th to the 12th century and were written by clerics. Since the Patrologia Latina does not contain annotations of dependency structures, it is not suitable for all the word embedding tools examined in this contribution. In order to explore the effect of incorporating dependency structure information

[11] `http://patristica.net/latina`

[12] `http://www.mgh.de/dmgh`

[13] `http://www.ceupress.com/books/html/CentralEuropeanMedievalTexts.htm`

into word embedding nonetheless, we included the Index Thomisticus. This corpus contains texts by Thomas Aquinas (1225–1274 AD). Its size is but a fraction of the Patrologia Latina, but it is annotated with dependency information (Passarotti 2015).

Table 9 gives an outline of the corpus statistics.

Table 9: Statistics of supplementary corpora used in the experiments

Corpus	Language	Sentences	Tokens
Gutenberg	German	24,766,958	440,896,599
Wikipedia	German	85,027,606	1,158,005,656
SZ	German	42,426,628	725,868,505
German-All	German	152,221,192	2,324,770,760
PL	Latin	5,598,592	133,158,974
Index Thomisticus	Latin	21,931	371,824

In order to compute word embeddings, we incorporate five different variants: The Mikolov model (Mikolov *et al.* 2013) optimizes word embeddings such that they can predict other context words occurring in a defined window. The model considers target-context word pairs inside a window of words to the right and to the left of the target word. Among other options, the text model chosen can be either the *continuous bag of words model* (cbow) or the *skip-gram model* (skip). The effect of varying this parameter is examined in the experiments discussed in Section 6.1.2. FastText (Bojanowski *et al.* 2017) is a library and tool to learn word embeddings as well as sentence classifications. Pennington *et al.* (2014) developed GloVe, which we also examined as an alternative to learn word embeddings. Levy and Goldberg (2014) modified the skip-gram model of Mikolov *et al.* (2013). They used dependency contexts instead of a window-based word context. Komninos and Manandhar (2016) introduced a variant of the skip-gram model that combines Mikolov skip-grams with those derived from dependency trees. Each target word optimizes word embeddings such that maximum probabilities of other words within distance one and two in the dependency tree are calculated. A weighting according to distance is applied. Words with distance one from the target word are counted twice. These word-word predictions behave similar to the window model of Mikolov *et al.* (2013). Dependency

parses are also used to filter coincidental co-occurrences (Komninos and Manandhar 2016).

5.5 *Morphological analyzers*

This article focuses on standard tools for natural language processing that are (i) generic, in the sense that they can in principle be applied to any language and genre, and (ii) produce an unambiguous annotation. However, there are resources and tools developed for a specific language that provide detailed information on lexical units and are widely used in the community.

Morphological analyzers provide information about inflected forms regarding morphology and lemmatization. They may also include information regarding segmentation (e.g. Lemlat). All of the following analyzers perform an out-of-context analysis, that is, they process each inflected form individually and may return multiple results. Furthermore, these results do not necessarily cover the actual valid result (when context is considered). Thus we cannot directly compare these resources to the tagging tools analyzed in our experiments. However, we can perform a coverage analysis in order to get an overall impression of the potential of the analyzers. For this purpose, we discard numerals and punctuation. Table 10 summarizes the coverages of the following morphological analyzers. Morpheus is a web service as part of the Perseus project (Crane 1991), which provides morphological analyses for Greek and Latin.[14] Lemlat 3.0 (Passarotti *et al.* 2017) is a morphological analyzer based on a lexical database. LatMor (Springmann *et al.* 2016) is a finite-state morphology for Latin, which, on our corpora, reached the best coverage.

Analyzer	Capitularies	Proiel
LatMor	99.527	99.426
Lemlat	97.616	99.495
Morpheus	92.371	96.297

Table 10:
Coverage of morphological analyzers on Latin corpora in percent

[14] To evaluate the coverage of Morpheus, we used a Python project (`https://github.com/tmallon/morpheus`) by Timothy Mallon, which wraps and caches requests on the Morpheus web service.

6 EXPERIMENTS

In this section, we evaluate the performance of various NLP tools. We do not seek to find only one perfect combination of tool and resources in order to proclaim a winner. To attempt to do so would require extensive hyperparameter experiments and tuning of each tool. Our experiments showed that this is almost impossible since some tools take hours to train a model on our compiled training/test partitions *and* have a wide parameter space to explore. And even then the results would be valid for the examined corpora but might differ significantly for other corpora. This is because the process of optimizing parameters to maximize accuracy for a given data set might lead to over-fitting for that specific task.

So our focus is rather on the practitioner: What accuracy can one generally expect from a given NLP approach – used off-the-shelf and using default hyperparameters – and how much time do you need to invest for training and predicting? Furthermore, we are interested in how significant the differences are between older and more modern approaches. Does "old" automatically need to stand for "no longer relevant"?

We start by examining POS and fine-grained POS tagging, followed by lemmatization. Finally, we evaluate what level of accuracy we can expect when putting the pieces together.

6.1 *Tagging*

Table 11 shows POS tagging accuracy, achieved without using any optimization or adding any supplementary resources for training. LemmaTag performed best on almost all German corpora, followed by MarMoT and FLORS. The picture becomes more diverse when considering the results for the Latin texts. While LemmaTag still performs best on the Capitularies, accuracy drops significantly for Proiel as well as (to a minor degree) for the two OD scenarios. On average, MarMoT and the tagging veteran TreeTagger are only 1.40 % apart. Another outlier worth noticing is FLORS, trained on Tiger and tested on TGermaCorp, which is 0.47 % above LemmaTag (second in place for this setting), followed by Mate and Lapos. As expected, we observed a significant drop in accuracy for OD experiments, compared to ID experiments. For example, MarMoT achieves 98.02 % accuracy on the

Table 11: POS accuracy for STTS and CHSTS in %. Lines written in italics indicate tagging results of out-of-vocabulary tokens only

	FLORS	Lapos	LemmaTag	MarMoT	Mate	OpenNLP	RDR	Stanford	TnT	TreeTagger
TG	92.33	93.12	**93.44**	93.34	92.89	91.64	91.09	91.67	91.63	92.22
TG	*82.07*	*84.69*	***86.02***	*86.01*	*84.54*	*80.73*	*69.41*	*76.05*	*81.30*	*79.85*
Tiger	97.69	97.84	**98.58**	98.02	97.88	96.84	96.72	97.17	97.23	97.09
Tiger	*91.59*	*93.00*	***93.93***	*93.74*	*93.05*	*90.18*	*81.20*	*85.99*	*90.78*	*88.49*
TG→Tiger	89.70	85.69	**90.69**	90.59	88.94	87.09	85.59	85.62	88.41	88.33
TG→Tiger	*80.57*	***80.77***	*79.95*	*80.61*	*75.34*	*72.09*	*64.13*	*65.65*	*74.08*	*71.73*
Tiger→TG	**90.59**	89.19	90.12	89.10	89.22	87.75	86.70	87.52	88.52	88.26
Tiger→TG	***78.44***	*69.40*	*72.42*	*68.83*	*69.74*	*66.37*	*57.44*	*60.07*	*65.70*	*64.95*
Capit	95.44	96.08	**96.18**	96.10	95.79	94.83	95.43	94.83	95.47	95.17
Capit	*78.72*	***83.54***	*80.28*	*83.22*	*81.79*	*77.04*	*72.90*	*62.95*	*78.30*	*74.95*
Proiel	93.11	**95.62**	92.00	95.49	95.56	93.51	94.11	92.96	94.36	94.32
Proiel	*79.84*	*84.83*	*76.76*	***85.10***	*83.82*	*80.13*	*73.32*	*66.03*	*76.74*	*76.64*
Capit→Proiel	84.72	**87.73**	86.20	87.58	86.80	82.62	84.66	82.12	85.79	85.50
Capit→Proiel	*70.65*	*77.32*	*70.91*	***77.72***	*75.41*	*65.22*	*63.85*	*51.91*	*69.79*	*68.33*
Proiel→Capit	62.44	63.48	61.54	63.05	**63.85**	62.31	61.42	54.95	62.94	61.22
Proiel→Capit	*42.79*	*43.39*	*42.01*	*44.00*	***44.56***	*42.42*	*39.51*	*28.43*	*42.08*	*38.60*
average	88.25	88.59	88.60	**89.16**	88.87	87.07	86.97	85.86	88.04	87.76
average	*75.59*	*77.12*	*75.28*	***77.41***	*76.03*	*71.77*	*65.22*	*62.13*	*72.35*	*70.44*

Tiger corpus (ID). But when Tiger is used as the training corpus to tag TGermaCorp, results drop by about 9%. The problem becomes even more apparent when a relatively small corpus such as Proiel is used to tag a larger corpus like the Capitularies. In this case, accuracy drops from 95.62 % (Proiel ID) to 63.48 % (Proiel→Capitularies) when using Lapos. However, this is partly because Proiel does not contain punctuation, so a model trained on that corpus is bound to produce more errors when applied to the Capitularies. Since a practitioner is often limited to using pre-trained models, OD experiments reveal a much more realistic view of what can be expected. In Section 6.1.2, we will examine how and to what degree accuracies in OD scenarios can be improved by supplementary resources. Our results show that given a specific tagger, accuracy may vary heavily across different resources. We also observe that the taggers perform quite differently for the same resource. This finding suggests, that it may be worth the effort to try more than one tagger on a given POS tagging task.

Taggers may vary considerably in the way they can cope with words which have not been part of the training set (out-of-vocabulary items). Table 11 lists the accuracy for out-of-vocabulary items in italics. Figure 3 shows a visualization of the differences compared to all tokens as heatmap. The more saturated the cell, the higher the delta. The maximum delta of 31.88% is reached by Stanford tagger on the Capitularies. Lapos and MarMoT perform best from the perspective of how well they can cope with out-of-vocabulary items. In contrast Stanford tagger, RDR and TreeTagger mark the other end of the spectrum. Figure 3 also reveals that the delta for Tiger→TG is much higher than for TG→Tiger which is in contrast to Capit→Proiel vs. Proiel→Capit: In the German out-of-domain scenario accuracy drops significantly more when Tiger, being the much larger corpus, is used to tag TGermaCorp. For Latin it is the other way around: Accuracy drops much more for Proiel being used as training set to tag the significantly larger Capitularies corpus.

What are the main reasons for tagging errors? Figure 4 depicts in a bipartite graph how often POS-tagging errors have occurred, using LemmaTag on Tiger. The arrows from the gold-standard (top) to the test results (bottom) represent the range of variation in POS-tagging errors, and their relative frequency of occurrence. For better readability, the figure only shows the ten most frequent nodes for gold

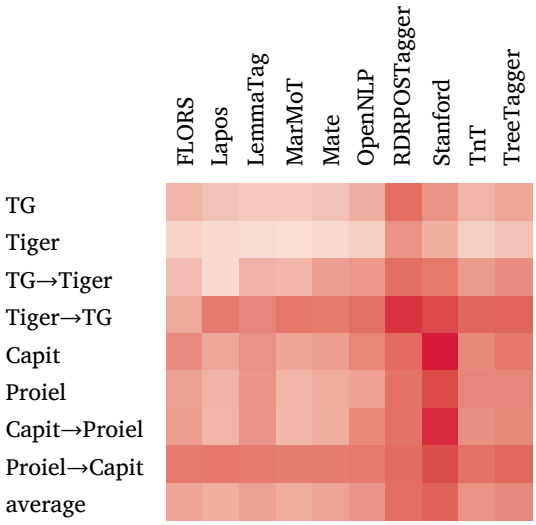


Figure 3: Heatmap depicting the degree to which accuracy drops when only out-of-vocabulary tokens are considered. The more saturated the cell color, the higher the delta

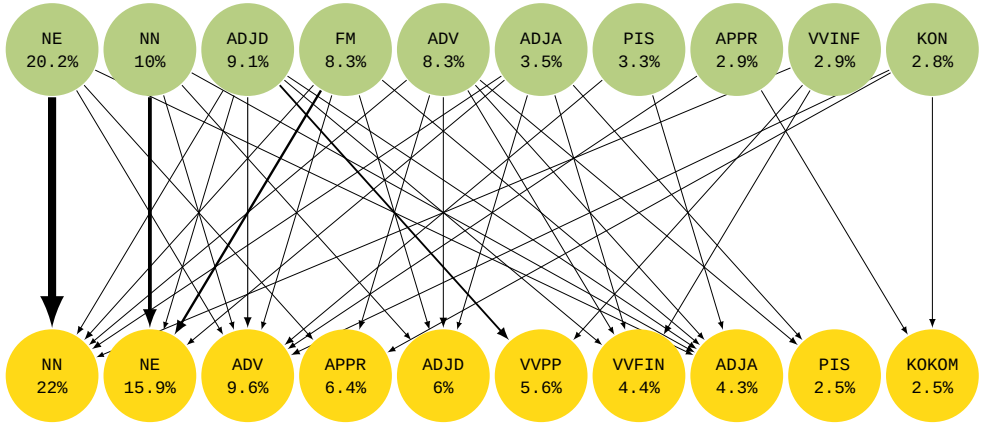


Figure 4: Depiction of the distribution and extent of erroneous POS tagging, comparing gold-standard tags (top) with test-results (bottom), using STTS for LemmaTag on Tiger

standard, and for test results, while arrows indicate at least ten mismatches. Percentages represent the ratio of all incoming or outgoing arrows with respect to the overall total. Apparently, most errors are caused by named entities recognized as nouns and vice versa. Like-

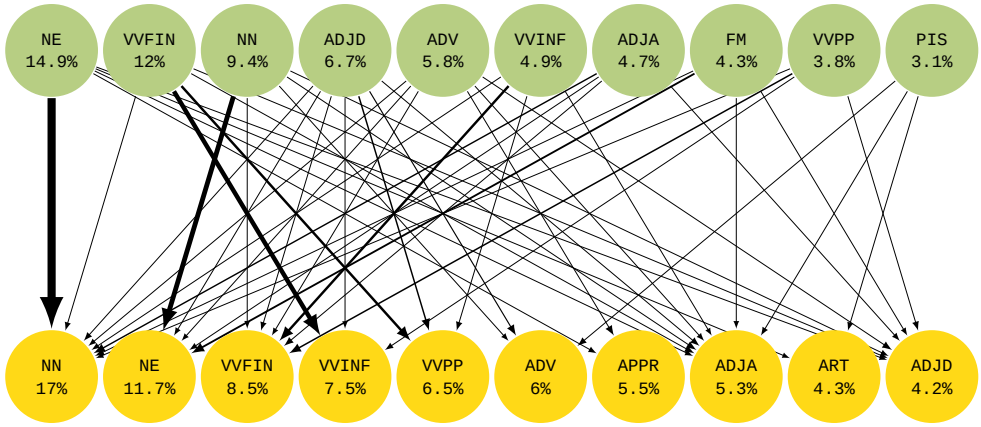


Figure 5: Depiction of the distribution and extent of erroneous POS tagging, comparing gold-standard tags (top) with test-results (bottom), using STTS for all taggers on Tiger

wise, finite verbs are often erroneously tagged as non-finite verbs and vice versa. Are these observations specific to LemmaTag or do other taggers show similar behavior? Compare Figure 4 with Figure 5, which shows the sum of all mismatches over all taggers combined. Percentages in the upper row code the fraction of errors where the corresponding POS was tagged erroneously; percentages in the lower line code the fraction of errors where this target POS was chosen. The figure is based on the same filter criteria. As before discriminating named entities and nouns as well as finite and non-finite verbs are the main sources of error.

The POS-tagging accuracy results shown in Table 11 for ID experiments are the average of a three-fold cross-validation. The choice of how to partition a corpus into a training set and a test set may have a significant impact, and thus make comparision between POS tagging results from the literature more difficult. The distribution of the deltas between the minimum and maximum value encountered for POS tagging in cross-validations (Figure 6) clearly shows that delta values depend on the size of the corpus. The smaller the corpus to be partitioned, the higher the probability that the training set will fail to cover words or patterns that are part of the test set.

In Section 5.2, we introduced a mapping from STTS to a simplified version that abstracts from variants of verbs, pronouns, etc.

	FLORS	Lapos	LemmaTag	MarMoT	Mate	OpenNLP	RDRPOSTagger	Stanford	TnT	TreeTagger
TG	0.36	0.55	0.16	0.47	0.41	0.19	0.06	0.22	0.63	0.42
Tiger	0.09	0.01	0.08	0.03	0.08	0.08	0.02	0.03	0.10	0.12
Capit	0.30	0.26	0.17	0.28	0.27	0.28	0.10	0.36	0.45	0.33
Proiel	1.07	0.75	0.54	0.35	0.39	1.19	0.81	0.77	1.03	0.57

Figure 6: Distribution of deltas $acc_{max} - acc_{min}$ for POS-taggers in %, based on a three-fold cross-validation. The more saturated the cell color, the higher the delta

The Latin tagset CHSTS is similar to sSTTS already, but still discriminates nouns from named entities and named persons. Consequently, we also examined the impact of an abstraction of the Latin tagset, sCHSTS. As some use cases in computational linguistics do not require an explicit distinction for verbs, pronouns, and other POS, the question is whether this abstraction results in significantly higher accuracy. Table 12 contrasts the STTS/CHSTS-based POS results for Tiger and the Capitularies with the simplified tagsets, sSTTS and sCHSTS.[15] All taggers show improvement after the abstraction. On average Tiger gains 1.09% while accuracy for Capitularies is increased by 0.32%. Since sCHSTS only abstracts nouns, the improvement is understandably less.

We already noted that mismatches between named entities and nouns, as well as variants of verbs, are a major source of POS tagging errors. Shifting to a simplified tagset brings into focus other errors that were already present in the original tagsets. Figure 7 shows the percentage of erroneous POS using sSTTS for all taggers on Tiger. As before, the figure shows only the ten most frequent types for gold standard and for test results, with arrows to indicate at least ten mis-

[15] For two out of three train/test partitions of Tiger, Lapos frequently tagged commas not as "$," but as "$(", "V" and other forms. As this error was not observed for other corpora nor other taggers, we presumed that this was an error in Lapos and therefore explicitly set the correct tag for commas in this specific case.

Table 12:
POS accuracy for all tagsets on Tiger and Capitularies in %

	Tiger STTS	Tiger sSTTS	Capit CHSTS	Capit sCHSTS
FLORS	97.69	**98.74**	95.44	**95.65**
Lapos	97.84	**98.69**	96.08	**96.31**
LemmaTag	98.58	**99.18**	96.18	**96.51**
MarMoT	98.02	**98.88**	96.10	**96.35**
Mate	97.88	**98.90**	95.79	**96.09**
OpenNLP	96.84	**98.11**	94.83	**95.18**
RDRPOSTagger	96.72	**97.92**	95.43	**95.88**
Stanford	97.17	**98.37**	94.83	**95.18**
TnT	97.23	**97.93**	95.47	**95.74**
TreeTagger	97.09	**97.86**	95.17	**95.60**
average	97.51	98.46	95.53	95.85

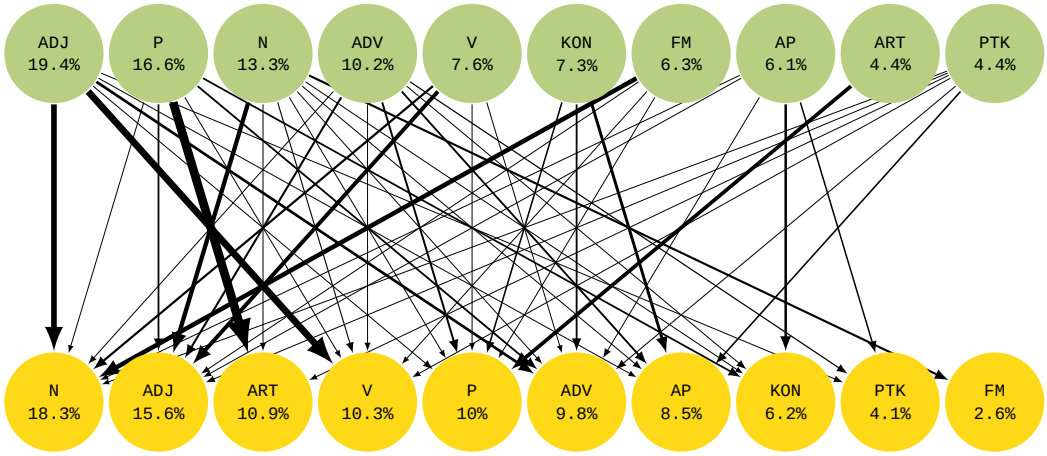


Figure 7: Percentage of POS tagging errors using sSTTS for all taggers on Tiger

matches. The most dominant errors are adjectives erroneously tagged as nouns or verbs. Another frequent error stems from pronouns beings tagged as articles. These errors were present in STTS as well but were dominated there by the much more frequent problem of distinguishing nouns from verbs.

So far we have only considered coarse-grained POS tagging. Now, we shed light on what accuracy can be achieved for fine-grained POS tagging. Instead of lining up result tables as we did for POS, we propose a graphical representation to depict tagging accuracy for the

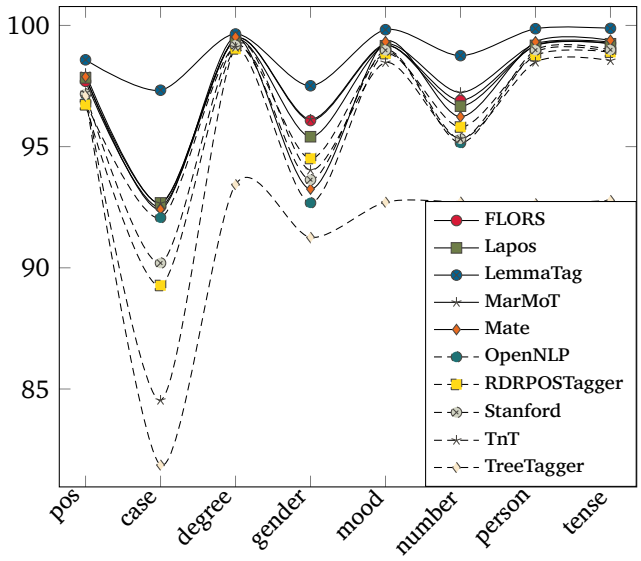


Figure 8:
Fine-grained POS tagging of the Tiger corpus (accuracy in %)

Tiger corpus (Figure 8) and for the Capitularies (Figure 9). For both corpora, results are relatively good for `degree`, `mood`, `person`, and `tense`, whereas `case`, `gender`, and `number` appear to be more challenging. All tools show similar behavior with respect to the accuracy for a given morphological category. LemmaTag performs exceptionally well on all categories, while TreeTagger marks the lower end of the spectrum.

6.1.1 Majority voting

When more than two annotations are generated for the same task, a majority-vote approach can be applied. Assuming that the majority is more likely to be right when tagging a specific token, erroneous outliers can be compensated for, thus leading to better results. We distinguish three groups of taggers: By "top3" and "top5" we denote the three or five most recently published (see Section 4): LemmaTag, FLORS, RDRPOSTagger, MarMoT, and Mate. By "all" we denote the ten taggers examined in our study. We also rely on (descending) order of publication of the taggers to resolve tie situations in majority votes.

As Table 13 shows, applying a majority vote improves performance in most cases. Unexpectedly, using the three most recently published taggers (LemmaTag, FLORS, and RDRPOSTagger, according to

Figure 9:
Fine-grained POS tagging of the Capitularies (accuracy in %)

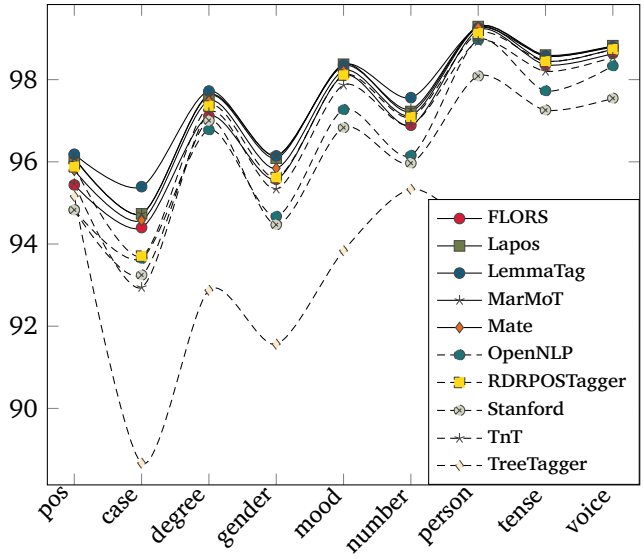


Table 13:
Majority vote tagging of POS (STTS/CHSTS) (accuracy in %)

Corpora	Best	Top3	Top5	All
TG	93.44	93.63	**93.91**	93.86
Tiger	**98.58**	98.30	98.37	98.29
TG→Tiger	90.69	90.47	91.37	**91.39**
Tiger→TG	**90.59**	90.41	90.44	90.08
Capit	96.18	96.20	**96.29**	96.28
Proiel	95.62	94.70	95.78	**96.04**
Capit→Proiel	87.73	86.49	87.82	**87.92**
Proiel→Capit	63.85	63.51	64.61	**64.76**

our list) cannot be recommended, as it improves accuracy in only two out of eight cases. However, by adding MarMoT and Mate to this list, accuracy exceeds the baseline in six out of eight cases. A similar improvement is observed when using all taggers.

We then went a step further and evaluated the performance of majority votes for any combination of taggers, in any possible order (to solve ties). With 10 taggers available, 9,864,000 combinations were computed. Table 14 lists the best combinations for fine-grained (STTS) POS tagging of the Tiger corpus, using a fixed number of taggers. In this scenario, the best result (98.44%) is achieved when using LemmaTag, Mate, TnT, and FLORS (in that order), which is 0.14% above

#	Acc.	Taggers (ordered chronologically, most recent first)
3	98.39	LemmaTag, FLORS, Mate
4	**98.44**	LemmaTag, Mate, TnT, FLORS
5	98.37	LemmaTag, RDRPOSTagger, FLORS, MarMoT, Mate
6	98.41	LemmaTag, RDRPOSTagger, Mate, FLORS, MarMoT, TreeTagger
7	98.33	LemmaTag, RDRPOSTagger, Mate, TnT, MarMoT, Lapos, FLORS
8	98.36	LemmaTag, RDRPOSTagger, Mate, Lapos, MarMoT, TreeTagger, TnT, FLORS
9	98.29	LemmaTag, Mate, MarMoT, TnT, TreeTagger, FLORS, Stanford, RDRPOSTagger, Lapos
10	98.31	LemmaTag, Mate, FLORS, RDRPOSTagger, Stanford, MarMoT, TnT, Lapos, TreeTagger, OpenNLP

Table 14: Best results for majority vote POS-tagging (STTS) of Tiger by number of taggers (accuracy in %)

the best single performer (LemmaTag). Our results suggest that, in practice, the relatively small gain in performance does not justify the great effort for training the individual tools in order to use majority vote tagging.

6.1.2 Supplementary resources

The first experiment presented in this article examined the performance of various tools on POS tagging, with no hyperparameter tuning and no additional resources whatsoever. We observed that, even though all tools produced reasonable results for ID scenarios, switching to OD settings had a severe impact on accuracy: The training data cannot cover the morphological and grammatical diversity, or the specific characteristics of the test domain. This problem becomes even more severe when only small training corpora are available. Creating well-annotated corpora as gold standard for tagging is a tedious and complex task. How can additional resources that are easier to produce be used to augment models, and what impact can be expected on the accuracy of our scenarios?

Lexical resources as well as word embeddings can be used with MarMoT to support the training of POS tags, while FLORS can benefit from additional unstructured texts. Because of the time required to train FLORS models, we focused on MarMoT to examine the impact of different resources on overall performance. We started by examining POS tagging of Latin corpora, using different combinations of lexicons

Table 15: POS (CHSTS) accuracy in % using MarMoT, for different types of word embedding, based on two Latin corpora

Emb. corp.	Corpora	No em.	Mik.	F.Text	GloVe	Kom.	Levy
PL	Capit	96.10	**96.26**	96.23	96.01	–	–
Thom.	Capit	**96.10**	96.08	96.09	96.06	96.03	96.01
PL	Pro	95.49	**96.85**	96.75	95.90	–	–
Thom.	Pro	95.49	95.67	95.56	95.75	**95.83**	95.78
PL	Cp→Pro	87.58	88.68	**88.69**	86.75	–	–
Thom.	Cp→Pro	**87.58**	87.28	87.35	86.40	86.33	86.14
PL	Pro→Cp	63.05	**69.61**	68.77	64.91	–	–
Thom.	Pro→Cp	63.05	63.65	63.47	62.92	**64.25**	64.24

and word embeddings. The word-embedding approaches of Komninos and Levy can only be applied to the Index Thomisticus, because the Patrologia Latina corpus lacks dependency information.

We used default values for Mikolov, FastText, GloVe, Komninos, and Levy, while aiming to keep results for the different approaches comparable. We therefore used a feature vector size of 100 and 5 iterations (Levy used 1 by default, which we changed to 5). As the default number of iterations for GloVe is 25, we did not limit the number of iterations to 5. For German, we set the minimum word frequency to 10 in order to compute embeddings within reasonable timeframe. For Latin, we used a minimum word frequency of 1.

Table 15 shows that using word embeddings based on the Patrologia Latina improved tagging results. Mikolov performs best in most cases, followed by FastText and GloVe. The most significant boost in terms of accuracy is achieved for the OD setting Pro→Cp. In this case, training POS tags based on the rather small Proiel corpus benefits best from the additional information provided by word embeddings. In two cases, GloVe has a negative impact compared to the baseline of using no word embeddings.

In contrast, the results of using Index Thomisticus as a basis for word embeddings are rather inconclusive. In about half the cases the baseline is not met. When Proiel is examined in-domain or out-of-domain, Komninos yields the best results, but in the other scenarios, it does not reach baseline accuracy either. Using dependency information may therefore have a positive impact, but a great deal depends on the corpora being processed. The Index Thomisticus consists only of texts from the 13th century, whereas the Capitularies date from

Corpora	Lexicon	Tagset	No em.	PL skip	PL cbow
Capit	none	CHSTS	96.10	96.26	95.02
Capit	FLL	CHSTS	96.52	**96.60**	95.52
Capit	none	sCHSTS	96.35	96.49	95.44
Capit	FLL	sCHSTS	96.67	**96.69**	95.77
Capit	La-WN	sCHSTS	96.37	96.49	95.45
Capit	all	sCHSTS	96.68	96.69	95.78
Cp→Pro	none	CHSTS	87.58	88.68	85.29
Cp→Pro	FLL	CHSTS	88.59	**88.79**	86.03
Pro→Cp	none	CHSTS	63.05	69.61	68.65
Pro→Cp	FLL	CHSTS	69.88	**72.82**	71.18

Table 16:
POS accuracy in %, using MarMoT, for combinations of lexicons and Mikolov-embeddings on Capitularies as well as OD-settings of Capitularies and Proiel

the Merovingian and Carolingian periods, while Proiel is based on ancient and early Christian Latin. Furthermore, the Index Thomisticus is rather small, at least compared to the Patrologina Latina. Finally, the Index Thomisticus treebank is completely set in lower-case, which may also contribute to the bad overall results.

Table 16 reveals that including lexical resources to train MarMoT consistently improves accuracy in ID as well as OD experiments. The most significant improvement can be observed when the relatively small Proiel corpus is used to tag the Capitularies. By including the Frankfurt Latin Lexicon (FLL), accuracy can be increased by 6.83% to 69.88%. Even a small lexicon like that extracted from the Latin WordNet, which is only a fraction the size of the FLL, has a positive impact on the results. As already noted, Proiel does not contain punctuation, and neither do the lexicons: thus the improvements result from the additional lexical resources, and are not merely a fix for a single shortcoming. Table 16 also shows that skip-gram clearly outperforms the continuous bag of words (cbow) model when using Mikolov. Finally, results confirm that using the simplified tagset sCHSTS over CHSTS also provides better results when supplementary resources are used. The best improvement can be achieved by incorporating skip-gram and FLL for the Proiel→Capit scenario, which raises accuracy from 63.05% to 72.82%.

In order to study the impact of supplementary resources on German, we rely on two lexica: one extracted from the German Wiktionary and the other from GermaNet (see Section 5.3). We used five

methods (Mikolov, FastText, GloVe, Komninos, and Levy) to compute word embeddings on four different corpora: Gutenberg, Süddeutsche Zeitung (SZ), Wikipedia, and a merged corpus thereof (see Section 5.4). Combined with four different ID and OD scenarios, for two different tagsets, there are too many options to be shown in detail. We rather focus on specific aspects, and then describe the whole.

We first examined the impact of different types of word embeddings, based on the four corpora available for this experiment. Table 17 shows the results for POS tagging, using MarMoT, on a selection of ID and OD settings for Tiger and TGermaCorp, using STTS. As we already saw for Latin, our findings indicate that using word embeddings can have a positive or negative effect on accuracy, depending on the method used and the parameters applied. Mikolov and Komninos yielded the best accuracy, even for OD scenarios. FastText was mostly on a par with Mikolov, whereas using GloVe gave slightly worse results, but was still better than the baseline of no embeddings at all. The expected accuracy gain when including dependency information for training with Komninos and Levy was much more obvious in the experiments on German in comparison with the Latin corpora. However, we assume that this result cannot be generalized to the language examined, but rather to the specific characteristics of the Latin corpus used.

Table 17: POS (STTS) accuracy in % using MarMoT, for different types of word embedding, based on four German corpora

Emb. corp.	Corpora	No em.	Mik.	F.Text	GloVe	Kom.	Levy
Gutenberg	Tiger	98.02	98.12	98.11	98.06	98.19	98.16
SZ	Tiger	98.02	98.20	98.20	98.11	98.29	98.22
Wikipedia	Tiger	98.02	98.17	98.16	98.05	98.25	98.19
Merged	Tiger	98.02	98.22	98.22	98.11	**98.32**	98.28
Gutenberg	Tig→TG	89.10	90.17	90.10	89.88	90.11	90.07
SZ	Tig→TG	89.10	89.81	89.67	89.92	89.67	89.54
Wikipedia	Tig→TG	89.10	89.79	89.72	89.84	90.07	89.92
Merged	Tig→TG	89.10	**90.45**	90.17	90.21	90.37	90.33
Gutenberg	TG→Tig	90.59	91.98	90.72	91.60	91.61	91.62
SZ	TG→Tig	90.59	92.51	92.18	91.19	93.10	92.84
Wikipedia	TG→Tig	90.59	92.37	92.64	91.54	93.43	93.44
Merged	TG→Tig	90.59	92.96	93.02	91.83	**93.80**	93.44

Corpora	Tagset	No em.	Mik.	F.Text	GloVe	Kom.	Levy
TG	STTS	93.34	94.03	94.05	93.64	**94.08**	93.79
Tiger	STTS	98.02	98.22	98.22	98.11	**98.32**	98.28
TG→Tiger	STTS	90.59	92.96	93.02	91.83	**93.80**	93.44
Tiger→TG	STTS	89.10	**90.45**	90.17	90.21	90.37	90.33
TG	sSTTS	96.02	**96.51**	96.43	96.27	96.47	96.40
Tiger	sSTTS	98.88	99.00	98.98	98.90	**99.04**	99.02
TG→Tiger	sSTTS	94.83	95.75	95.55	94.85	**96.15**	96.13
Tiger→TG	sSTTS	92.41	93.22	93.10	93.10	**93.35**	93.07

Table 18: POS accuracy in % using MarMoT, for different types of word embedding, based on all German corpora (merged)

Do we get a different picture when we consider the simplified version of STTS, which abstracts from different types of verbs, etc.? Table 18 presents the results for POS tagging, comparing STTS with sSTTS. Apart from the general improvement of accuracy by switching to sSTTS, we observe the same behavior as with plain STTS when embeddings are introduced. Using embeddings based on Komninos yields the best results. The accuracy for Tiger reaches the 99% mark.

In Section 4, we mentioned an approach for POS tagging based on Bidirectional Long Short-Term Memory Recurrent Neural Networks (BLSTMRNN; Wang *et al.* (2015)). Because of limited hardware resources and the extensive time needed to train the networks, we did not fully include it in our experiments. However, we did perform a POS tagging experiment on Tiger (ID), in order to estimate the performance of this class of taggers compared to others. We include it in the section discussing supplementary resources because our implementation strictly requires word embeddings (rather than using them as an option as MarMoT does). We used Mikolov and the merged German corpus for this task. Using BLSTMRNN achieves an accuracy of 98.29%, which is comparable to MarMoT (98.22%), using the same word embedding.

So far we have only considered word embeddings as a supplementary resource for German POS tagging. In the following, we examine the impact of the two lexicons based on Wiktionary and GermaNet. Table 19 shows the accuracy for combinations of lexicons and embeddings on the Tiger corpus. Since we observed that Levy was outperformed by Komninos based on the merged German corpus, we only considered one option (Komninos) as word embedding. Wik-

Table 19: POS (sSTTS) accuracy in % for combinations of German lexica and word embeddings, using MarMoT with Komninos, based on the merged German corpus

	None	GNet	Wikt.	AllLex	Emb	All&Emb
TG	96.02	96.11	**96.26**	96.24	96.47	96.46
Tiger	98.88	98.90	**98.95**	98.94	99.04	99.04
TG→Tig	94.83	94.96	95.14	**95.16**	96.15	96.16
Tig→TG	92.41	92.50	92.56	**92.61**	93.35	93.33

tionary and, to a lesser extent, GermaNet provided a slight gain in performance. Merging both resources (AllLex) had no significant effect. Combining lexical resources and embeddings did not produce the expected improvement in accuracy. In fact, our findings indicate that, at least in this setting, using both types of resources may even lead to a slight drop in accuracy. This scenario suggests that it is better to rely solely on embeddings rather than lexical resources or a combination of both.

We therefore conclude that including lexical resources for POS tagging is consistently beneficial. Incorporating word embeddings may improve results in many cases, especially in OD scenarios. However, depending on the corpus being processed, as well the method and parameters being used to compute the embeddings, the effect may also be negative, as seen in our Latin experiments. For Mikolov, the skip model outperforms the continuous bag of words model.

6.1.3 RDR-based POS tagger on external taggers

Our experiments have shown that the RDR-based POS tagging approach produced only modest results. So far, however, we have simply applied the out-of-the-box procedure, using the internal, lexicon-based tagger for initial tagging, as a practitioner will most likely do. As Nguyen *et al.* (2016) have demonstrated, using an external tagger can yield better results.[16] They report an accuracy of 96.28 using the internal tagger for initial tagging and 97.46 using TnT, which constitutes an improvement of 1.18. In our experiment, we measured 96.72 as the baseline when using the internal tagger. By integrating TnT as the initial tagger, we achieve an accuracy of 97.62, an improve-

[16] According to personal communication with the authors, the data published in Nguyen *et al.* (2016) is not based on 10-fold jack-knifing, which they nonetheless recommend to achieve better results. Following their recommendation, we applied 10-fold jack-knifing in our experiments.

ment of 0.9 compared to the baseline and 0.39 compared to TnT. Can this approach also improve the results for the MarMoT tagger, among the most successful taggers on the Tiger corpus in our experiments, with an accuracy of 98.02? When using MarMoT as the initial tagger for the RDR-based POS tagging approach, accuracy reaches 98.05, a slight improvement. This shows that the method can significantly improve the results of mediocre taggers, and may even have a positive impact on the best performing taggers. However this improvement massively increases computation costs in both training and testing.

6.1.4 Pipeline vs joint tagging

So far, we have examined each category of tagging independently. But as soon as morphological tagging is included, the question of maximum accuracy arises, when all tags have to fit the gold standard. We examined two different approaches: pipeline-learning and (what we call) "joint-learning". In pipeline-learning, we train and test each category independently. Then we combine all results to compute the overall accuracy. The advantage of this approach is that it can be performed by virtually any tagger. One of its major disadvantages, however, is that we cannot integrate any dependencies between categories into the learning process. Thus, it is theoretically possible for a noun to be tagged with a tense because each category is learned and tested independently.

A joint approach (in our sense) integrates all grammatical categories into one tag, which helps to capture dependencies. However, the tagset space becomes considerably larger: the Tiger corpus requires 53 STTS tags, but when the tags required to capture morphology are added, the total rises to 694 joint tags. Likewise, the Capitularies use 19 CHSTS tags, and 913 joint tags. In our experiments, taggers like Lapos and Stanford could not handle such a vast tagset space.

Table 20 shows accuracy results for joint and pipeline tagging on Tiger, Capitularies, and Proiel (ID/OD). As Figures 8 and 9 have already shown, accuracy varies considerably from category to category. While results for POS are relatively stable across all taggers, results for `case` and `gender` are significantly worse. The results achieved by pipeline learning are unlikely to be better than the individual results for each category.

Table 20: Accuracy in % for joint and pipeline tagging, on Tiger, Proiel, and Capitularies (STTS/CHSTS tagset)

	Tiger		Capitularies		Proiel		Capit→Proiel		Proiel→Capit		Average	
	Pipe	Joint	Pipe	Joint	Pipe	Joint	Pipe	Joint	Pipe	Joint	Pipe	Joint
FLORS	86.25	88.43	87.17	89.47	77.72	82.40	56.67	61.61	36.61	43.06	68.88	72.99
Lapos	85.99	–	88.74	–	83.02	–	61.43	–	41.19	–	**72.07**	–
LemmaTag	**93.72**	**93.68**	**89.58**	90.03	76.99	73.10	**62.11**	**63.43**	37.88	38.09	72.06	71.67
MarMoT	86.99	89.69	88.60	**90.70**	**83.03**	**84.37**	60.84	63.18	40.69	**43.25**	72.03	**74.24**
Mate	83.56	84.90	87.72	89.95	82.04	83.33	59.49	61.66	41.05	42.33	70.77	72.43
OpenNLP	81.88	85.54	84.40	87.57	75.33	79.01	51.04	56.06	36.68	38.61	65.87	69.36
RDRPOSTag.	82.16	85.97	87.83	89.43	82.13	83.02	60.46	61.72	**41.30**	40.87	70.78	72.20
Stanford	81.70	–	85.61	–	74.75	–	52.36	–	31.12	–	65.11	–
TnT	76.88	85.34	86.23	89.52	80.76	83.89	58.71	62.31	39.03	42.38	68.32	72.69
TreeTagger	72.68	81.02	83.22	87.17	73.07	81.67	51.84	58.73	29.94	40.59	62.15	69.84

It is striking how LemmaTag outclasses other taggers for morphological tagging of the Tiger corpus, reaching 93.72% accuracy for pipeline tagging, i.e. 6.73% above MarMoT in second place. Results for Latin corpora (whether ID or OD) are far closer. LemmaTag still performs reasonably well on the Capitularies, both ID and OD. But when Proiel is processed in-domain or as training set to tag Capitularies, results for LemmaTag fall to the lower end of the spectrum. An outlier worth noticing is the RDRPOSTagger, which performs best on the Proiel→Capit OD setting. As expected, the joint approach outperforms pipeline tagging in most cases, and can thus be recommended when the tagger supports it. LemmaTag can be recommended for German and it also performs well on Latin corpora, at least on the Capitularies. When accuracy for the joint approach is averaged, MarMoT obtains the best results, mainly because of its better performance on Proiel. This may be due to the fact that Proiel lacks punctuation. Future work needs to address in more detail the robustness of taggers on morphological tagging for OD settings and small corpora.

6.2 *Lemmatization*

To evaluate lemmatizers, we trained systems on the (form,lemma) pairs available in the training data. For LAT, we considered lemmatization as a tagging task, as described in Gesmundo and Samardzic (2012). We used the MarMoT tagger for this, without any additional resources. We could have used any other tagger, but MarMoT has proven to work well for POS tagging *and* can deal with large label spaces. For example, using Tiger as the training corpus for lemmatization results in 2,405 labels. This would not be suitable for Lapos or Stanford. We found (Table 21) that LemmaTag performs best in most cases (performing joint POS tagging and lemmatization); LAT is consistently better than LemmaGen, which corroborates results from Gesmundo and Samardzic (2012). We also note that ID accuracy on Tiger is higher than on TGermaCorp, but Tiger is a larger corpus than TGermaCorp (∼50,000 sentences vs. ∼9,000) and more homogeneous, while TGermaCorp contains poems, various types of literature, etc. The OD results seem pretty low, compared to ID results, as was the case for tagging. We remark that specific lemma conventions differ between the two datasets. In addition, TGermaCorp contains spelling mistakes (*Widersehen*), and historical variants (*Capital*) that are con-

Table 21: Direct lemmatization accuracy in %

	LemmaGen	MarMoT-LAT	LemmaTag
TG	91.71	91.99	**92.11**
Tiger	98.07	98.19	**98.66**
TG→Tiger	87.00	**88.57**	88.37
Tiger→TG	87.26	88.05	**90.34**
Capit	95.64	95.81	**96.13**
Proiel	**90.63**	90.29	81.85
Capit→Proiel	81.39	81.24	**82.25**
Proiel→Capit	76.28	**76.37**	49.61

ventionally lemmatized by their modern lemma forms (*Kapital*). Such cases are difficult for trained systems to handle, since they are absent from Tiger.

As Tables 22 and 23 show for the German TGermaCorp and the Latin Capitularies, the most frequent lemmatization errors by LAT and LemmaGen correlate well. Confusing the two variants of the German conjunction “dass” and “daß” (old) is partly due to the training corpus, which contains documents from different decades and centuries, and partly to genuine inconsistencies in the gold standard, which should be corrected in future editions. The impact on overall accuracy is still quite minor – normalizing these variants improves accuracy for TGermaCorp ID by about 0.1%. Table 24 illustrates how the type of lemmatization error shifts when the model is either trained based on Proiel (ID) or on the Capitularies (OD), which decreases accuracy by about 9%.

6.3 *Tagging and lemmatization*

In Section 6.1.4, we addressed the pros and cons of joint vs pipeline learning of POS and morphology tags. Here, we complete the picture by examining accuracy for lemmatization *and* tagging combined. Table 25 shows accuracy for LemmaTag (joint fine-grained tagging and lemmatization), ranging from 92.95% for lemmatization and fine-grained tagging on Tiger ID to only 30.44% on Proiel→Capit. Likewise, the loss of accuracy resulting from the extension of lemmatization by POS and morphology ranges from 5.71% on Tiger and 7.6% Capitularies to 20–25% for Proiel and the two Latin OD settings.

Table 22: Most frequent lemmatization errors with LAT, LemmaGen, and LemmaTag on TGermaCorp

LemmaGen				LAT				LemmaTag			
Form	Gold	Pred.	#	Form	Gold	Pred.	#	Form	Gold	Pred.	#
was	wer	was	71	was	wer	was	34	was	was	wer	36
einer	einer	ein	42	–	–	–	32	–	–	–	32
am	am	an	38	was	was	wer	25	rtr	–	rtr	18
eine	einer	ein	36	rtr	–	rtr	24	dessen	dessen	der	17
–	–	–	32	ap	–	ap	20	dpa	–	dpa	16
alle	alle	aller	26	dpa	–	dpa	18	was	wer	was	14
rtr	–	rtr	24	dessen	dessen	der	17	ap	–	ap	10
viele	viele	vieler	22	viele	viele	vieler	16	weiter	weit	weiter	9
Deutschen	deutsche	deutsch	22	eine	einer	ein	15	allem	aller	alle	8
ap	–	ap	21	einer	einer	ein	14	gestattet	gestatten	statten	7
uns	sich	uns	21	alle	alle	aller	13	früher	früher	früh	7
einen	einer	ein	18	einen	einer	ein	13	.	.	–	7
dpa	–	dpa	18	am	am	an	12	uns	uns	sich	7
dessen	dessen	der	17	afp	–	afp	10	uns	sich	uns	7
mich	sich	mich	17	allem	aller	alle	8	mich	sich	mich	7
eines	einer	ein	17	allen	alle	aller	8	AFP	AFP	afp	6
Den	Den	der	16	weiter	weit	weiter	8	afp	–	afp	6
lange	lang	lange	14	.	.	–	7	geeignet	eignen	geeignet	5
Polen	Pole	Polen	10	Wissen	wissen	Wissen	7	unserer	mein	unser	5
einem	einer	ein	10	früher	früher	früh	7	,	,	–	5

Table 23: Most frequent lemmatization errors for LAT, LemmaGen, and LemmaTag on Capitularies

LemmaGen				LAT				LemmaTag			
Form	Gold	Predicted	#	Form	Gold	Predicted	#	Form	Gold	Predicted	#
quod	qui	quod	157	quod	qui	quod	135	quod	qui	quod	142
quam	qui	quam	101	quam	qui	quam	82	quam	qui	quam	51
nostri	nostri	noster	58	nostri	nostri	noster	43	qua	qui	quis	42
qua	qui	quis	51	qua	qui	quis	37	nostri	nostri	noster	38
quo	qui	quo	51	quo	qui	quo	19	quam	quam	qui	24
modo	modo	modus	32	qui	quis	qui	19	bannum	bannum	bannus	21
bannum	bannum	bannus	27	fiat	fio	fiat	17	modo	modo	modus	20
primo	primo	primus	27	quod	quod	qui	17	tantum	tantus	tantum	19
tantum	tantum	tantus	22	bannum	bannum	bannus	17	qui	quis	qui	19
una	una	unus	21	quo	quo	qui	16	fiat	fio	fiat	17
qui	quis	qui	19	postea	postis	postea	16	fiant	fio	fiant	17
postea	postis	postea	19	modo	modo	modus	15	eodem	eodem	idem	14
fiant	fio	fiant	17	eodem	idem	eodem	15	nostrum	nos	noster	14
fiat	fio	fiat	17	fiant	fio	fiant	14	tantum	tantum	tantus	13
una	unus	una	16	quam	quam	qui	12	quid	quid	quis	12
bona	bonum	bonus	16	postea	postea	postis	12	factum	factum	facio	12
eodem	eodem	idem	14	tantum	tantus	tantum	12	alium	alius	prefulgidus	11
nostrum	nos	noster	14	bannum	bannus	bannum	12	maxime	magnus	magis	11
factum	factum	facio	13	maxime	magnus	magis	11	excepto	excepto	exceptus	11
quid	quid	quis	12	etc	et cetera	et	11	forte	forte	fors	10

Table 24: Most frequent lemmatization errors for LAT Proiel ID and Capitularies → Proiel OD

LAT Proiel				LAT Capit→Proiel			
Form	Gold	Predicted	#	Form	Gold	Predicted	#
quod	qui	quod	17	se	is	sui	630
quam	qui	quam	8	a	ab	a	333
minus	parum	minus	7	ac	atque	ac	291
una	una	unus	5	sibi	is	sui	210
respondit	respondeo	respondo	5	castra	castra	castrum	177
oportere	oportet	oporteo	4	sese	is	sui	153
tergum	tergum	tergus	4	quod	qui	quod	140
coeperunt	incipio	cobeo	4	uti	ut	utor	120
actis	ago	actis	4	castris	castra	castrum	99
Hi	is	hic	4	quam	qui	quam	86
Si	si	is	4	se	se	sui	69
progressus	progredior	progredo	3	minus	parum	parve	57
magis	magis	magus	3	copias	copia	copio	51
primum	primus	primum	3	Germanos	Germani	Germanus	51
iuris	ius	iur	3	Germani	Germani	Germanus	51
vocibus	vox	vocis	3	equitatu	equitatus	equitas	48
reliquit	relinquo	reliquo	3	equites	eques	equites	45
plures	multus	plures	3	Germanorum	Germani	Germanus	45
pedem	pes	pedem	3	Haeduis	Aedui	Haeduis	43
influit	influo	infelio	3	equitatum	equitatus	equitas	40

	Lemma	Lemma POS	POS morph	Lemma POS morph
TG	92.11	87.44	–	–
Tiger	98.66	97.21	93.68	92.95
TG→Tiger	88.37	82.05	–	–
Tiger→TG	90.34	83.39	–	–
Capit	96.13	94.33	90.03	88.53
Proiel	81.85	78.54	73.10	62.48
Capit→Proiel	82.25	76.22	63.43	57.40
Proiel→Capit	49.61	44.06	38.09	30.44

Table 25: Accuracy in % for combined POS tagging, morphological joint-tagging, and lemmatization, using LemmaGen

Table 26: Run-times for a partition of Tiger (ID)

	POS		Joint	
FLORS	3h16m26s	3m42s	1d7h39m56s	7m14s
Lapos	6m30s	13s		
LemmaTag (GPU)	26m7s	2s	2h23m16s	10s
LemmaTag (CPU)	18h42m57s	1m58s		
MarMoT	6m1s	10s	44m47s	38s
Mate	11m57s	21s	3h0m0s	4m4s
OpenNLP	8m10s	4s	1h17m23s	27s
RDR	20m33s	10s	21m5s	1m
Stanford	35m12s	11s		
TnT	5s	697ms	13s	2s
TreeTagger	2s	438ms	1m12s	5s

6.4 *Computing time*

Until now, we have focused on evaluating tools with regard to the accuracy achieved for tagging and lemmatization. However, when a parameter space needs to be explored to determine the optimal configuration for training a model, or when large collections of text need to be tagged, runtime cannot be neglected. Table 26 shows the time needed to train and test a partition of the Tiger corpus. Here, 45,372 tokens in 799,406 sentences were used for training, and 5,100 tokens in 88,832 sentences for testing. Please note that these numbers may vary considerably, depending on the workstation or server the processes run on, as well as the machine's current load. However, they give a first impression of what to expect. Note also that the time for testing included the time taken to load the trained files.[17] While TreeTagger takes about 2 seconds to train and less than a second to test, FLORS takes more than 3 hours to train and 3 minutes 42 seconds to test. When taking accuracy into account, MarMoT appears to be the best trade-off, as it needs only 12 minutes for training and 10 seconds for testing. Alternatively, if a GPU workstation is available, LemmaTag is a good choice.[18] Without a GPU, training and testing for LemmaTag can also be done on CPUs. But for CPU-based POS tagging, we measured a training time of over 18 hours, with a test time

[17] LemmaTag performs training and testing in one process chain, so that the model files would have already been loaded.

[18] We conducted our experiments on a workstation equipped with one NVidia GTX1080 and one GTX1060.

of about 2 minutes. So even if a GPU is used for training, testing on a CPU is still not a good option with large corpora. In order to tag large amounts of text, we recommend splitting the corpus into chunks and processing them separately – either by separate processes or by multi-threading within one application. In this regard all taggers (to our knowledge) suffer from the problem of not being thread-safe, such that multiple instances cannot share the same model file in memory. This means that each instance needs to load its own representation of the trained model. So, depending on the hardware available for tagging, a bottle-neck may be caused by short memory rather than by CPU issues.

Finally, we examine the time needed to train and test lemmatization based on Tiger. LemmaGen takes 6 seconds for training and less than 1 second for testing, while LAT based on MarMoT takes 2h29m9s for training and 1m47s for testing. Since accuracy results for LemmaGen and MarMoT-LAT differ only marginally, LemmaGen has a much better performance-runtime trade-off than MarMoT-LAT. LemmaTag takes 26m7s for training and 2s for testing – the same time as it takes for POS tagging, since it performs both tasks in one sweep.

7 DISCUSSION

In terms of accuracy, MarMoT, FLORS, Lapos, Mate, and particularly LemmaTag are the methods of choice when (especially) fine-grained morphological tagging is the goal. LemmaTag performs exceptionally well on German corpora but its results for Latin, especially on Proiel and Latin OD, show that it is not the unique solution that fits all cases. For POS tagging, MarMoT proves to be competitive. When trained with embeddings as a supplementary resource, it reaches up to 98.32% on Tiger, while LemmaTag achieved 98.58%. Moreover, the lexical resources as well as the word embeddings derived from unlabeled data that can be fed into MarMoT (and FLORS) make the systems more robust to change of domain, which is an immensely important aspect of real-world POS tagging. With respect to word embeddings, we note that the approach (FastText, GloVe, Mikolov, Komninos or Levy) and its parameters play an important role. The choice of parameters to compute embeddings can either improve or degrade accuracy, as shown in Table 16. In our settings, Mikolov and FastText outperform

GloVe. When dependency parsing information is available, Komninos yields better results than Levy. The question as to whether Komninos is better than Mikolov cannot be answered unambiguously, as the results are somewhat similar, depending on language and corpus. Our experiments on German corpora indicate that Komninos should be preferred over Mikolov, as long as enough parsed corpora are available to compute the embeddings.

Accuracy aside, it is also interesting to take a closer look at the time needed for training and testing (see Table 26).[19] Various aspects may have an impact on run-time. First, time depends linearly on the hardware being used. Second, processing times depend on the configuration of the taggers – especially with regard to training a new model. Finally, the taggers evaluated are implemented in different programming languages, and the implementations are not necessarily optimized for everyday field-use but for high accuracy. This means that even though the computational complexity of a given approach cannot be changed, a great deal of time can be saved by using efficient programming techniques, with extensive use of multi-threading. Both FLORS and the Stanford tagger can take hours or even days to train a model, depending on the size of the corpora and the parameters. In contrast, TnT and TreeTagger, while typically performing less well – sometimes badly – in terms of accuracy, take only a fraction of the training and testing time of the more recent generation. For a practitioner, computing time can be a critical issue. For example, when it comes to tagging large corpora, such as the entire Wikipedia, even a few seconds more per processed text can make a huge difference. So, depending on the task at hand, even the older approaches may still be attractive.

In this respect, TnT is particularly interesting: it is about as fast as TreeTagger, and its fine-grained tagging accuracy is often only marginally below that of MarMoT. For example, with only one exception, TnT is within ∼1% point of the (joint) fine-grained POS tagging accuracy of MarMoT (used without additional resources), while the TreeTagger is between 3% and 8% below MarMoT with respect to

[19] Please note that the computation times can only give a general impression, and include the time to load/save model files.

fine-grained tagging. At the same time, training time on Tiger (ID) for joint fine-grained tagging is 13s for TnT vs. 44m for MarMoT and above 1m for TreeTagger. The good results of LemmaTag on German corpora confirm the success of neural network approaches – at the cost of extensive computation time when no GPUs are available. This suggests that, in near future, tagging large corpora such as Wikipedia will require a solid GPU workstation or even a cluster.

Regarding the drop of performance in OD experiments, we note that OD experiments actually constitute a *lower bound* on performance, while ID experiments constitute an (ideal case) *upper bound*. The reason why ID experiments are not entirely reliable, for a practitioner, is that it is assumed that the test data will have the same distributional properties as the data on which a machine-learning system has been trained. This is implausible in almost all practical scenarios. On the other hand we also note that OD experiments often indicate errors that are, from a linguistic perspective, not errors at all but correspond, for example, to different *conventions* according to which different datasets have been annotated.

Similar to POS tagging, we can observe a trade-off between accuracy and run-time for lemmatization. In most cases, LemmaTag performs best, followed by MarMoT-LAT, and LemmaGen. Thus LemmaTag is the best solution if GPUs are available and processing time is less of an issue. When having to decide between LemmaGen and MarMoT-LAT, the latter appears to be the best choice. But for in-domain settings the difference is only marginal – the largest gap of 1.57% can be observed when TGermaCorp is used to train a model for lemmatization of Tiger. On the other hand, MarMoT-LAT takes 107 seconds for lemmatization of Tiger (ID), whereas LemmaGen takes only 5 seconds. So the choice of lemmatizer depends on the use-case. If processing time is not an issue or if there is enough hardware to scale on, MarMoT-LAT would be a good choice. Since LemmaGen's accuracy is at a similar level, even outperforming LAT in some cases, it can be considered a good option to lemmatize large corpora.

In a nutshell, we conclude that the performance of taggers can be raised by including distributional information (about paradigmatic word associations), as computed by word embeddings or by means of (still mostly handcrafted) lexicons. It is no surprise that this relates especially to OD scenarios. However, it also means that, especially in the

case of low-resource languages, for which lexicons are rarely available, neural network models are the first choice when trying to improve tagging results. In scenarios of processing historical texts, either lexical or procedural information about alternative spellings would be an alternative informational resource that may also help computing word embeddings at a more abstract level (of super-lemmas instead of lemmas alone).

Our study has also demonstrated the computational effort necessary to train and to evaluate new taggers – especially in the light of the ever-increasing size of annotated corpora that can be used for training and testing. In order to capture this effort, one may think of a *meta-learner* that takes as input (1) taggers, (2) learning scenarios (e.g., fine-grained/coarse-grained), (3) annotated corpora, and (4) additional resources (embeddings or lexica), in order to update the desired evaluation. One may think of a toolbox for building large-scale evaluation studies allowing for laborious (hyper-)parameter studies in terms of big data-experiments. Ideally, this would function *out of the box* so that newly available annotation data could be rapidly used to retrain taggers. Currently, comparative studies are still very laborious rather than being easily manageable.

8 CONCLUSION

We conducted a study of tagging for German and (classical as well as medieval) Latin texts by examining a range of older taggers in comparison with more recent ones. We experimented with coarse-grained as well as fine-grained POS tagging, and with lemmatization. Our findings highlight the improvements achieved by the most recent tagger developments. LemmaTag, in particular, performed best in most of the tasks considered here. We also show that out-of-domain (OD) tagging leads to a considerable loss in tagging accuracy. The same is true when we consider pipeline learning of inflectional categories. These findings hint at the need to further develop taggers, possibly by extending feature space (e.g., by morphological, syntactic, or even semantic features). However, our experiments also show that such an extension may lead to a considerable increase in training and operating time, and thus may be problematic in the case of time-critical scenarios. Last but not least, we evaluated three lemmatizers. Here

also, LemmaTag performs best in most cases, followed by LAT. Our experiments show once more that accuracy drops significantly if the lemmatizer is applied OD. As before, this is a good argument for further developments in this area of NLP, in particular, to address domain adaptation.

OPEN SOURCE

Models for taggers evaluated here are available online.[20] This includes all annotated corpora, as far as license terms allow.

REFERENCES

Bernd BOHNET and Joakim NIVRE (2012), A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing, in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 1455–1465, Association for Computational Linguistics, Jeju Island, Korea, http://www.aclweb.org/anthology/D12-1133.

Piotr BOJANOWSKI, Edouard GRAVE, Armand JOULIN, and Tomas MIKOLOV (2017), Enriching word vectors with subword information, *Transactions of the Association for Computational Linguistics*, 5:135–146, ISSN 2307-387X.

Sabine BRANTS, Stefanie DIPPER, Peter EISENBERG, Silvia HANSEN-SCHIRRA, Esther KÖNIG, Wolfgang LEZIUS, Christian ROHRER, George SMITH, and Hans USZKOREIT (2004), TIGER: Linguistic interpretation of a German corpus, *Research on Language and Computation*, 2(4):597–620, doi:10.1007/s11168-004-7431-3.

Thorsten BRANTS (2000), TnT: A statistical part-of-speech tagger, in *Proceedings of the Sixth Conference on Applied Natural Language Processing*, ANLC '00, pp. 224–231, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/974147.974178, http://dx.doi.org/10.3115/974147.974178.

Jacob COHEN (1960), A coefficient of agreement for nominal scales, *Educational and Psychological Measurement*, 20:37–46.

Paul COMPTON and Bob JANSEN (1988), Knowledge in context: A strategy for expert system maintenance, *Proceedings of the 2nd Australian Joint Artificial Intelligence Conference*, pp. 292–306.

Gregory CRANE (1991), Generating and parsing classical Greek, *Literary and Linguistic Computing*, 6(4):243–245, doi:10.1093/llc/6.4.243, http://dx.doi.org/10.1093/llc/6.4.243.

[20] https://www.texttechnologylab.org/applications/corpora/

Hannes DOHRN and Dirk RIEHLE (2011), Design and implementation of the Sweble Wikitext Parser: Unlocking the structured data of Wikipedia, in *Proceedings of the 7th International Symposium on Wikis and Open Collaboration*, WikiSym '11, pp. 72–81, ACM, New York, NY, USA, ISBN 978-1-4503-0909-7, doi:10.1145/2038558.2038571, http://doi.acm.org/10.1145/2038558.2038571.

Markus DREYER, Jason SMITH, and Jason EISNER (2008), Latent-variable modeling of string transductions with finite-state methods, in *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pp. 1080–1089, Association for Computational Linguistics, Honolulu, Hawaii, http://www.aclweb.org/anthology/D08-1113.

Greg DURRETT and John DENERO (2013), Supervised learning of complete morphological paradigms, in *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9–14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pp. 1185–1195.

Steffen EGER (2015), Designing and comparing G2P-type lemmatizers for a morphology-rich language, in *Systems and Frameworks for Computational Morphology – Fourth International Workshop, SFCM 2015, Stuttgart, Germany, September 17–18, 2015, Proceedings*, pp. 27–40, doi:10.1007/978-3-319-23980-4_2.

Steffen EGER, Tim VOR DER BRÜCK, and Alexander MEHLER (2015), Lexicon-assisted tagging and lemmatization in Latin: A comparison of six taggers and two lemmatization methods, in *Proceedings of the 9th Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaTeCH 2015)*, pp. 105–113, Beijing, China.

Joseph L. FLEISS (1971), Measuring nominal scale agreement among many raters, *Psychological Bulletin*, 76:378–382.

Andrea GESMUNDO and Tanja SAMARDZIC (2012), Lemmatisation as a tagging task, in *The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, July 8–14, 2012, Jeju Island, Korea – Volume 2: Short Papers*, pp. 368–372.

Birgit HAMP and Helmut FELDWEG (1997), GermaNet – a lexical-semantic net for German, in *In Proceedings of ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, pp. 9–15.

Dag Trygve Truslew HAUG and Marius JØHNDAL (2008), Creating a parallel treebank of the old Indo-European Bible translations, in *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*.

Wahed HEMATI, Tolga USLU, and Alexander MEHLER (2016), TextImager: a Distributed UIMA-based system for NLP, in *Proceedings of the COLING 2016 System Demonstrations*, pp. 59–63, Federated Conference on Computer Science and Information Systems.

Verena Henrich and Erhard Hinrichs (2010), GernEdiT – The GermaNet editing tool, in Nicoletta Calzolari, Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, European Language Resources Association (ELRA), Valletta, Malta, ISBN 2-9517408-6-7.

Mans Hulden, Markus Forsberg, and Malin Ahlberg (2014), Semi-supervised learning of morphological paradigms and lexicons, in *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 569–578, Association for Computational Linguistics, Gothenburg, Sweden, doi:10.3115/v1/E14-1060, https://www.aclweb.org/anthology/E14-1060.

Matjaž Juršič, Igor Mozetič, Tomaž Erjavec, and Nada Lavrač (2010), LemmaGen: Multilingual lemmatisation with induced ripple-down rules., *J. UCS*, 16(9):1190–1214, http://dblp.uni-trier.de/db/journals/jucs/jucs16.html#JursicMEL10.

Bernhard Jussen, Alexander Mehler, and Alexandra Ernst (2007), A corpus management system for historical semantics, *Sprache und Datenverarbeitung. International Journal for Language Data Processing*, 31(1–2):81–89.

Alexandros Komninos and Suresh Manandhar (2016), Dependency based embeddings for sentence classification tasks, in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1490–1500, Association for Computational Linguistics, San Diego, California, doi:10.18653/v1/N16-1175, https://www.aclweb.org/anthology/N16-1175.

Daniel Kondratyuk, Tomás Gavenciak, Milan Straka, and Jan Hajic (2018), LemmaTag: Jointly tagging and lemmatizing for morphologically-rich languages with BRNNs, *CoRR*, abs/1808.03703:4921–4928.

Klaus Krippendorff (1980), *Content analysis*, volume 5 of *The SAGE KommTexT Series*, SAGE Publications, Beverly Hills and London.

Matthieu Labeau, Kevin Löser, and Alexandre Allauzen (2015), Non-lexical neural architecture for fine-grained pos tagging, in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 232–237, Association for Computational Linguistics, Lisbon, Portugal, http://aclweb.org/anthology/D15-1025.

Omer Levy and Yoav Goldberg (2014), Dependency-based word embeddings, in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 2, pp. 302–308.

Andy Lücking, Armin Hoenen, and Alexander Mehler (2016), TGermaCorp – A (digital) humanities resource for (computational) linguistics, in *Proceedings of the 10th International Conference on Language Resources and Evaluation*, LREC 2016, pp. 4271–4277.

Christopher D. MANNING, Mihai SURDEANU, John BAUER, Jenny FINKEL, Steven J. BETHARD, and David MCCLOSKY (2014), The Stanford CoreNLP natural language processing toolkit, in *Association for Computational Linguistics (ACL) System Demonstrations*, pp. 55–60, http://www.aclweb.org/anthology/P/P14/P14-5010.

Barbara MCGILLIVRAY, Marco PASSAROTTI, and Paolo RUFFOLO (2009), The Index Thomisticus treebank project: Annotation, parsing and valency lexicon, *TAL*, 50(2):103–127, http://atala.org/IMG/pdf/TAL-2009-50-2-04-McGillivray.pdf.

Alexander MEHLER, Tim VOR DER BRÜCK, Rüdiger GLEIM, and Tim GEELHAAR (2015), Towards a network model of the coreness of texts: An experiment in classifying Latin texts using the TTLab Latin tagger, in Chris BIEMANN and Alexander MEHLER, editors, *Text Mining: From Ontology Learning to Automated text Processing Applications*, Theory and Applications of Natural Language Processing, pp. 87–112, Springer, Berlin/New York.

Tomas MIKOLOV, Ilya SUTSKEVER, Kai CHEN, Greg CORRADO, and Jeff DEAN (2013), Distributed representations of words and phrases and their compositionality, in *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.

Stefano MINOZZI (2008), La costruzione di una base di conoscenza lessicale per la lingua latina: LatinWordnet, http://hdl.handle.net/11562/324939.

Thomas MÜLLER, Ryan COTTERELL, Alexander M. FRASER, and Hinrich SCHÜTZE (2015), Joint lemmatization and morphological tagging with Lemming, in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17–21, 2015*, pp. 2268–2274.

Thomas MÜLLER, Helmut SCHMID, and Hinrich SCHÜTZE (2013), Efficient higher-order CRFs for morphological tagging, in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 322–332, Association for Computational Linguistics, Seattle, Washington, USA, http://www.aclweb.org/anthology/D13-1032.

Dat Quoc NGUYEN, Dai Quoc NGUYEN, Dang Duc PHAM, and Son Bao PHAM (2014), RDRPOSTagger: A ripple down rules-based part-of-speech tagger, in *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 17–20, Association for Computational Linguistics, Gothenburg, Sweden, http://www.aclweb.org/anthology/E14-2005.

Dat Quoc NGUYEN, Dai Quoc NGUYEN, Dang Duc PHAM, and Son Bao PHAM (2016), A robust transformation-based learning approach using ripple down

rules for part-of-speech tagging, *AI Communications*, 29(3):409–422, http://dx.doi.org/10.3233/AIC-150698.

Garrett NICOLAI, Colin CHERRY, and Grzegorz KONDRAK (2015), Inflection generation as discriminative string transduction, in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 922–931, Association for Computational Linguistics, Denver, Colorado, http://www.aclweb.org/anthology/N15-1093.

Marco PASSAROTTI (2015), What you can do with linguistically annotated data. From the Index Thomisticus to the Index Thomisticus Treebank, in *Reading Sacred Scripture with Thomas Aquinas. Hermeneutical Tools, Theological Questions and New Perspectives*, pp. 3–44, Brepols.

Marco PASSAROTTI, Marco BUDASSI, Eleonora LITTA, and Paolo RUFFOLO (2017), The Lemlat 3.0 package for morphological analysis of Latin, in *Proceedings of the NoDaLiDa 2017 workshop on processing historical language*, 133, pp. 24–31, Linköping University Electronic Press, Linköpings universitet, ISSN 1650-3740.

Jeffrey PENNINGTON, Richard SOCHER, and Christopher D. MANNING (2014), GloVe: Global vectors for word representation, in *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, http://www.aclweb.org/anthology/D14-1162.

Adwait RATNAPARKHI (1996), A maximum entropy model for part-of-speech tagging, in *Proceedings of the 1st Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Philadelphia, Pennsylvania.

Toni RIETVELD and Roeland VAN HOUT (1993), *Statistical techniques for the study of language and language behaviour*, Mouton de Gruyter, Amsterdam.

Anne SCHILLER, Simone TEUFEL, Christine STÖCKERT, and Christine THIELEN (1999), Guidelines für das Tagging deutscher Textcorpora mit STTS (kleines und großes Tagset), Technical report, Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart.

Helmut SCHMID (1994), Probabilistic part-of-speech tagging using decision trees, in *International Conference on New Methods in Language Processing*, pp. 44–49, Manchester, UK.

Tobias SCHNABEL and Hinrich SCHÜTZE (2014), FLORS: Fast and simple domain adaptation for part-of-speech tagging, *TACL*, 2:15–26, https://tacl2013.cs.columbia.edu/ojs/index.php/tacl/article/view/183.

Carsten SCHNOBER, Steffen EGER, Erik-Lân DO DINH, and Iryna GUREVYCH (2016), Still not there? Comparing traditional sequence-to-sequence models to encoder-decoder neural networks on monotone string translation tasks, in *Proceedings of COLING 2016, the 26th International Conference on Computational*

Linguistics: Technical Papers, pp. 1703–1714, The COLING 2016 Organizing Committee, Osaka, Japan.

Uwe SPRINGMANN, Helmut SCHMID, and Dietmar NAJOCK (2016), LatMor: A Latin finite-state morphology encoding vowel quantity, *Open Linguistics*, 2(1):386–392, doi:10.1515/opli-2016-0019.

Kristina TOUTANOVA, Dan KLEIN, Christopher D. MANNING, and Yoram SINGER (2003), Feature-rich part-of-speech tagging with a cyclic dependency network, in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology – Volume 1*, NAACL '03, pp. 173–180, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1073445.1073478, http://dx.doi.org/10.3115/1073445.1073478.

Yoshimasa TSURUOKA, Yusuke MIYAO, and Jun'ichi KAZAMA (2011), Learning with lookahead: Can history-based models rival globally optimized models?, in *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, CoNLL '11, pp. 238–246, Association for Computational Linguistics, Stroudsburg, PA, USA, ISBN 978-1-932432-92-3, http://dl.acm.org/citation.cfm?id=2018936.2018964.

Tim VOR DER BRÜCK and Alexander MEHLER (2016), TLT-CRF: A lexicon-supported morphological tagger for Latin based on conditional random fields, in Nicoletta CALZOLARI, Khalid CHOUKRI, Thierry DECLERCK, Sara GOGGI, Marko GROBELNIK, Bente MAEGAARD, Joseph MARIANI, Helene MAZO, Asuncion MORENO, Jan ODIJK, and Stelios PIPERIDIS, editors, *Proceedings of the tenth international conference on language resources and evaluation (LREC 2016)*, European Language Resources Association (ELRA), Paris, France, ISBN 978-2-9517408-9-1.

Peilu WANG, Yao QIAN, Frank K. SOONG, Lei HE, and Hai ZHAO (2015), Part-of-speech tagging with bidirectional long short-term memory recurrent neural network, *CoRR*, abs/1510.06168, http://arxiv.org/abs/1510.06168.

Wenpeng YIN, Tobias SCHNABEL, and Hinrich SCHÜTZE (2015), Online updating of word representations for part-of-speech tagging, in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1329–1334, Association for Computational Linguistics, Lisbon, Portugal, http://aclweb.org/anthology/D15-1155.

Modeling morphological learning, typology, and change: What can the neural sequence-to-sequence framework contribute?

Micha Elsner[1], *Andrea D. Sims*[1], *Alexander Erdmann*[1], *Antonio Hernandez*[1], *Evan Jaffe*[1], *Lifeng Jin*[1], *Martha Booker Johnson*[1], *Shuan Karim*[1], *David L. King*[1], *Luana Lamberti Nunes*[2], *Byung-Doh Oh*[1], *Nathan Rasmussen*[1], *Cory Shain*[1], *Stephanie Antetomaso*[1], *Kendra V. Dickinson*[2], *Noah Diewald*[1], *Michelle McKenzie*[1], *and Symon Stevens-Guille*[1]

[1] Department of Linguistics, The Ohio State University

[2] Department of Spanish and Portuguese, The Ohio State University

ABSTRACT

We survey research using neural sequence-to-sequence models as computational models of morphological learning and learnability. We discuss their use in determining the predictability of inflectional exponents, in making predictions about language acquisition and in modeling language change. Finally, we make some proposals for future work in these areas.

Keywords: morphology, computational modeling, typology

1 INTRODUCTION

Theoretical morphologists have long appealed to notions of learning, or learnability, to explain language change and the varied typological patterns of the world's languages. The high-level argument is simple: all natural languages must be learned, and "unlearnable" linguistic systems cannot survive. Therefore, the learning mechanism provides constraints on what sorts of languages can exist in the world. In the realm of morphology, however, it has not proven simple to define

learnability (or, as it is often described, *morphological complexity*). Different theories offer different ideas of what must be learned in order to acquire a morphological system, and how to measure the difficulty of the learning problem for a particular language.

In doing so, they have sometimes used computational models of the learner to buttress their claims. In many cases, their tools for model-building draw on the rich tradition of morphological processing within computational linguistics. Computational linguists construct models of morphology not only as direct contributions to linguistic research, but as engineering solutions to the low token/type ratios of languages with large inflectional paradigms; such models have been applied to language generation, machine translation, and other tasks. In recent years, a particular model from the machine translation community, the *neural sequence-to-sequence* model, has grown in popularity for morphological tasks. Sequence-to-sequence models are now being applied, not only as engineering solutions, but also as theoretically interesting models of morphological complexity.

This paper provides an overview of both theoretical and computational work in this framework. Beginning with an overview of morphological complexity, and the different proposals for how it can be measured, we show that sequence-to-sequence models are a natural fit for the Word and Paradigm model and its notion of Integrative Complexity. We present some criticisms of previous implementations of Integrative Complexity, and explain how sequence-to-sequence modeling has already begun to address them. However, we spotlight several areas where the framework, as currently conceived, falls short. We go on to describe some important open questions to which it might be applied in the future.

2 THEORETICAL FOUNDATIONS

Work in computational morphology has often been concerned with engineering questions, rather than with modeling speakers' morphological knowledge. Whether recent computational models can be profitably applied to theoretical questions thus depends on the extent to which the structure of a model reflects principles of morphological theory. To see the issues at hand, we begin with an overview of two

theoretical positions, the Item-and-Arrangement (IA) family of theories, and the Word-and-Paradigm (WP) family (names proposed by Hockett (1954); see Blevins (2016) for a historical overview). These contrasting positions set forth different concepts of what morphological complexity is, and how it might shape morphological typology.

2.1 *Learnability and typology in morpheme-based models*

IA models take the *morpheme* as the fundamental unit of analysis,[1] and describe inflectional systems in terms of their syntagmatic structure – that is, the associations between stems, affixes and meanings. For models built upon this assumption, it is natural to define the language learner's task as acquisition of the morpheme inventory and the syntagmatic rules for composing morphemes into words. This, in turn, tends to lead to a focus on the size of morphological systems, what Ackerman and Malouf (2013) call a language's 'enumerative complexity' (E-complexity). Quantitative measurement of this kind of morphological complexity has a long history, going back to Greenberg (1960).

One typological generalization that has been approached from an IA perspective is that languages tend to have far fewer inflection classes than they could, given their number of allomorphs. Table 1 shows a simplified example from Icelandic. Two allomorphs are shown

[1] Traditional IA models define morphemes as lexical bundles of minimal form and minimal meaning. This is consistent with the principle of 'incrementalism' (Stump 2001), according to which concatenating a morpheme to a stem adds the morpheme's form to the word and simultaneously adds its meaning, with meaning broadly construed to include morphosyntactic and morphosemantic values. The meaning of a word should thus be fully determined by the meanings of its parts plus their order of combination. However, as Blevins (2013: 436) points out, "...ideas tend to outlive the traditions that initially hosted them and mutate during their own lifespans". Incrementalism has proven too restrictive, and starting in the early 1990's (Anderson 1992; Halle and Marantz 1993) it was largely replaced by 'realizationalism', which postulates that operations on form, such as concatenating an affix to a stem, are licensed by the morphosyntactic properties of a word. Formal operations thus *realize* the meaning of a word rather than adding to a word's meaning. Some modern theories, such as Distributed Morphology (Halle and Marantz 1993; Harley and Noyer 2003), adopt realizationalism while retaining other IA/morpheme-based assumptions, such as the primary importance of concatenative operations and syntagmatic (stem-affix) relations in morphological structure. These theories are 'lexical-realizational' in the terminology of Stump (2001).

Table 1: Select inflected forms of Icelandic verbs: three paradigm cells with two allomorphs each. 1 and 2 are logically possible but unattested classes

	GRÍPA 'grasp'	KALLA 'shout'	*1	*2
1SG.PST	greip	kall-aði	X	X-aði
2SG.PST	greip-st	kall-aðir	X-aðir	X-st
3SG.PST	greip	kall-aði	X	X-aði

for each of three paradigm cells. (The Icelandic verb system has both more paradigm cells and more classes, but these few forms are sufficient for illustration.) Based on the forms shown, there could mathematically be as many as $2\times2\times2 = 8$ inflection classes, if the allomorphs of different paradigm cells were independent of each other. And as the number of allomorphs and paradigm cells grows, the number of possible classes – and thus the potential E-complexity of the inflectional system – increases rapidly. Yet allomorphs tend not to be independent of each other: this is why it is useful to talk about inflection *classes*. Indeed, in Icelandic verbs the 1SG.PST zero allomorph (as in GRÍPA) is never found in the same paradigm as the 2SG.PST allomorph *-aðir*. Likewise, 1SG.PST *-aði* (as in KALLA) is never found with 2SG.PST *-st*, and so on. While eight classes are potentiated by these allomorphs, the shown allomorphs in fact group into two classes – the minimum possible number. Moreover, this is representative of a strong tendency cross-linguistically for the actual number of classes observed in a language to be far fewer than the mathematically possible number of classes (Carstairs 1987).[2] This raises the question: Why?

A number of morphological theories attempt to explain this and other constraints by appealing to learnability. In an IA framework, there is often an assumption that more inflection classes and larger paradigms make languages more difficult to learn, and that inductive learning biases must therefore serve to constrain the learner's hypothesis space. Perhaps most famously, the No Blur Principle (Carstairs-McCarthy 1994), later revised as Vocabular Clarity (Carstairs-McCarthy 2010), posits that for each paradigm cell, only one allomorph can "...fail to identify inflection class unambiguously" (1994:742). In other words, there can be only one 'default' (class-unspecified) form for each paradigm cell. This proposed constraint is

[2] Apparent exceptions include Burmeso and Nuer (Baerman 2012; Baerman *et al.* 2017).

rooted in the idea that learning biases must serve to constrain learners' hypotheses about allomorph distributions. Specifically, Carstairs-McCarthy views No Blur as a byproduct of the Principle of Contrast (Clark 1987), the idea that in lexical acquisition, children are biased towards hypothesizing that a difference in word form corresponds to a difference in word meaning. Extending the Principle of Contrast from words to inflectional allomorphs, Carstairs-McCarthy defines inflection class membership as meaning in the relevant sense. He proposes that the observed cross-linguistic restriction on the proliferation of inflection classes is indirectly caused by an inductive bias that pushes child language learners towards positing that each suppletive allomorph either belongs to a different inflection class, or does not bear inflection class meaning. From an IA perspective there is logic to this extension, since morphemes (including any suppletive allomorphs) are taken to be the units of storage in the lexicon – the level at which form and meaning are related.

Important here are ways in which Carstairs-McCarthy's assumptions about the nature of morphological knowledge shape his conceptualization of the relationship between learning and inflectional typology. The first thing to observe is that Carstairs-McCarthy posits a fundamental distinction between concatenative and non-concatenative morphological processes; No Blur applies only to concatenative allomorphs. While this distinction is motivated theory-internally in IA models, it has no clear independent motivation (Stump 2001). Minimally, this raises questions about why the Principle of Contrast should constrain learners' hypotheses about allomorph distributions *only* for concatenative morphology. We know of no empirical evidence supporting this assumption.

Second, Halle and Marantz (2008) point out that cross-linguistically, inflection classes often group hierarchically into macroclasses, yet such distributions virtually require individual allomorphs to belong to multiple classes. They focus on the empirical problems that such patterns create for No Blur, but even if we set these aside, we can observe that Carstairs-McCarthy's hypothesis about how learning might shape morphological typology reflects an IA emphasis on morphemes as isolable form-meaning units and the syntagmatic (stem-affix) dimension of structure. It seems to imply that paradigmatic relationships among words/classes (e.g. whether they fully separate,

hierarchically grouped, cross-classifying, etc.[3]) are irrelevant to questions of learnability, beyond what is dictated by No Blur. However, this is an open question.

Finally, although No Blur does not place an absolute limit on the number of inflection classes in a language,[4] it will generally predict that the actual number of classes in a language is substantially smaller than the potential number of its classes, capturing the cross-linguistic tendency observed above.[5] Since allomorphs realizing different paradigm cells must be class-specific for the most part, No Blur indirectly captures the grouping of allomorphs into classes as a byproduct of the learner's acquisition of the morpheme inventory.

Blevins (2004) and Ackerman and Malouf (2015) argue that No Blur is derivative of the paradigmatic structure of inflectional morphology, and paradigmatically-structured learning of morphology. We turn to this perspective in the following section.

2.2 *Learnability and typology in word-based models*

Word-and-Paradigm morphology offers an alternative link between learnability and inflectional typology. WP is in many respects the oldest framework for inflectional theory, reflected in the traditional pedagogical approach to describing classical languages' inflectional systems in terms of their principal parts. (A lexeme's principal parts are those inflected forms that together suffice to deduce all of the lexeme's inflected forms, i.e. its full paradigm of surface word-forms.) Such models take the word as the basic unit of morphological structure and analyze inflectional meaning as being instantiated via paradigmatic contrasts – that is, contrasts between the forms filling different inflectional cells.[6] The learner's task, therefore, is to understand the relationships between the forms of each lexeme. We focus here on

[3] For further discussion from the typological perspective, see Dressler *et al.* (2006) and Brown and Hippisley (2012).

[4] Unlike its predecessor, Paradigm Economy (Carstairs 1987), which directly defines constraints on the number of classes that a language can have.

[5] Except that there are languages that violate the various formulations of the constraint, whether stated as Paradigm Economy, No Blur, or Vocabular Clarity (Halle and Marantz 2008; Müller 2007; Þorgeirsson 2017; Stump and Finkel 2013).

[6] WP models differ in the extent to which the abstract concept of the paradigm is considered to be a metaphor, emergent structure, or a reified theoretical primi-

the *abstractive* WP framework, in which the key relationships are directly between surface forms.[7] Abstractive models take as their starting point the idea that the inflected forms of a lexeme are interpredictable to some (potentially substantial) degree. Or, as Wurzel (1989: 114) stated it, "...inflectional paradigms are, as it were, kept together by implications. There are no paradigms (except highly extreme cases of suppletion) that are not based on implications valid beyond the individual word, so that we are quite justified in saying that inflectional paradigms generally have an implicative structure."

Importantly, in WP models there is no requirement that paradigm cells be realized by some segmentable phonological form (a classical morpheme); cells can also be realized by non-concatenative morphological operations that alter the phonological form of the stem. These operations can encompass, for instance, root-and-pattern morphology in Semitic languages, tonal morphology in Bantu languages, and German ablaut. Moreover, WP models make no explicit or implicit assumptions that there should be a one-to-one correspondence between morphological form and meaning. They accommodate insertion of material (exponents) with no obvious meaning,[8] multiple exponence, in which a meaning is signaled by multiple morphological pieces, and zero exponence, in which there is no phonological change corresponding to a change in meaning. WP models are thus "inferential-

tive and direct object of study. We touch on this interesting question in Section 6, in the context of what we call the Paradigm Cell Discovery Problem.

[7] As with IA models, modern WP models differ in many respects from classical WP models, reflecting in part the adoption of goals and principles from modern generativism (Blevins 2016; Matthews 1972). Importantly here, modern WP models can be divided into *constructive* and *abstractive* types (Blevins 2006). Constructive models (Anderson 1992; Stump 2001) characterize the morphological structure of a word in terms of form operations applied to lexically-stored stems to produce surface inflected forms. In contrast, abstractive models (Blevins 2016; Bochner 1993; Albright 2002a) describe the morphological structure of a word in terms of form operations applied to one or more surface word-forms to produce another.

[8] For example, verbal inflection classes in many Indo-European languages are organized around so-called 'theme vowels' (e.g. [a] vs. [e] vs. [i] in Spanish *am-a-r* 'love-TV-INF', *ten-e-r*, 'have-TV-INF', and *part-i-r* 'depart-TV-INF'), which serve to mark the inflection class of the verb but do not bear any syntactically- or semantically-relevant meaning.

Table 2: Indicative present forms of two Icelandic verb classes. Some allomorphs are the same in both classes

	GRÍPA 'grasp'	KALLA 'shout'
1SG.PRS	gríp	kall-a
2SG.PRS	gríp-ur	kall-ar
3SG.PRS	gríp-ur	kall-ar
1PL.PRS	gríp-um	köll-um
2PL.PRS	gríp-ið	kall-ið
3PL.PRS	gríp-a	kall-a

realizational" (Stump 2001), also sometimes called "a-morphous" (Anderson 1992).

Given its postulation that surface forms serve as the bases for other surface forms, abstractive WP models must contend with the question of how hard it is for speakers to predict an unobserved surface word-form for a lexeme, given some other word-form(s) in the paradigm; this is the Paradigm Cell Filling Problem (PCFP) (Ackerman *et al.* 2009). For illustration, we return to the simplified Icelandic example introduced earlier. Table 2 shows that while the two verbs represent different classes, some allomorphs are the same for both. If a speaker encounters a new verb in the 3PL.PRS with allomorph *-a*, the distribution of allomorphs engenders some amount of uncertainty regarding what some of the present and past tense forms of the verb are. (For the latter, see Table 1 in the previous section.) When a word is subject to the PCFP, there may thus be some amount of ambiguity regarding its inflection class membership. Morphological complexity, then, is taken to be the difficulty of this problem for a speaker or learner of some particular language.

As noted above, in an IA framework the complexity of an inflectional system tends to be conceptualized in terms of the size of its paradigms and the number of its classes (i.e. its E-complexity). However, we know of no clear evidence that languages with high E-complexity are more challenging to learn. There are many clear examples of natural languages with large paradigms – Kibrik (1998) famously observed that in principle, verbs in the Nakh-Daghestanian language Archi can have more than 1.5 million forms each. Moreover, historical change may increase, rather than reduce, the E-complexity of an inflectional system, even though we might predict that this renders languages less learnable. To give just one example, in the Iranian language Zazaki, phonological and syntactic competition among ezafe

forms has resulted in the development of a complex system of nominal inflection (with upwards of 144 paradigm cells) that includes rampant fusionality and syncretism. This stands in stark contrast with closely related languages in which a more agglutinative system can still be observed (Karim 2019). It is admittedly harder to provide example languages with large numbers of inflection classes, since the number of classes identified for an inflectional system depends heavily on analytic assumptions (Parker 2016), rather than being directly empirically observable. But such examples have certainly been proposed: for example, as many as 115 classes of Russian nouns (Parker 2016). Ultimately, these arguments raise doubts about whether metrics based on E-complexity are directly related to the learnability of morphological systems.

However, there is no particular reason that an E-complex language should have a difficult PCFP; it is not the *number* of forms that matters but their *predictability*. Thus, the WP model offers a different formulation of complexity, which Ackerman and Malouf (2013) term "Integrative complexity" (I-complexity). As the number of allomorphs in an inflectional system grows, the potential I-complexity of the system grows, but to the extent that the inflected forms of lexemes are interpredictable, it is possible for the actual I-complexity to remain low (Ackerman and Malouf 2013). Studies have attempted to measure the complexity in this sense for various languages' inflectional systems using set-theoretic (Stump and Finkel 2013) and information-theoretic (Ackerman and Malouf 2013; Stump and Finkel 2013; Bonami and Beniamine 2016; Sims and Parker 2016; Cotterell *et al.* 2018a) measurements.[9] These studies have generally focused on the role of abstractive paradigmatic relations – conceptually, proportional analogy – in solving the PCFP.

Like IA models, WP models have attempted to explain typological patterns by appealing to learnability, and in WP frameworks the PCFP has been intimately connected to typological questions. In particular, abstractive WP models propose that the cross-linguistic restriction on

[9] A separate line of investigation has found that information-theoretic measurements of inflectional paradigmatic relations predict speakers' response times in lexical decision tasks (Milin *et al.* 2009; Moscoso del Prado Martín *et al.* 2004), suggesting the relevance of abstractive-type relations to morphological processing.

the proliferation of inflection classes that we noted in the preceding section is caused by the need for learners to be able to solve the PCFP.

Ackerman and Malouf (2013) estimate the average difficulty of the PCFP in a language by its average conditional entropy: abstracting away from stems, they calculate the average unpredictability of the exponent (affix) realizing one paradigm cell, given the exponent for another paradigm cell of the same lexeme. This implementation of proportional analogy does not capture any predictiveness that derives from similarity among whole words, an issue that we return to below, but it does offer a quantification of how difficult it is to predict inflectional exponents from other exponents (for example, Icelandic 2SG : *-ir* :: 3SG : *-i*). Finding low average conditional entropy in each of ten languages (generally, less than 1 bit, equivalent to or better than a coin toss), they conclude that the PCFP is not exceptionally difficult in these languages and propose a typological universal, the Low Entropy Conjecture: "...enumerative morphological complexity is effectively unrestricted, as long as the average conditional entropy, a measure of integrative complexity, is low..." (436). Their conclusion is consistent with that of Stump and Finkel (2013), who find that an inflectional system's average cell predictor number – a set-theoretic measure of the number of dynamic principal parts required to determine the inflected form corresponding to a given paradigm cell – tends to be low.[10]

Ackerman and Malouf connect the PCFP to the Low Entropy Conjecture via learnability:

> If low entropy is the correct measure for explaining the implicational organization of paradigms, rendering complex systems learnable, then this makes a prediction about types of systems that we do not find… [T]here are no known fully suppletive systems in the languages of the world. This absence is easily explicable given the low entropy conjecture and its facilitating function for learnability: a completely suppletive system is one in which no form bears an implicational

[10] In dynamic principal parts analysis, the paradigm cells identified as principal parts need not be the same from one inflection class to another. This contrasts with static principal parts analysis, in which the cells that function as principal parts are identified for an inflection class system as a whole, rather than on a class-by-class basis.

> relation with any other form, and thus there is no useful patterned organization reflective of low entropy. (Ackerman and Malouf 2013: 454)

The logic here seems to be that inflected forms that are not learnable on the basis of implicative relations are likely to be subject to analogical changes that make them more predictable.

Moreover, word tokens have Zipfian distribution, so most inflected forms of most lexemes are sparsely attested in both adult speech (Baayen 2001) and child-directed speech (Lignos and Yang 2018). This suggests that speakers of morphologically rich languages encounter the PCFP on a regular basis and throughout their lifetimes (Bonami and Beniamine 2016). From the perspective of a WP theory, we might posit that token frequencies of word-forms play an important role in defining which lexemes and paradigm cells are subject to the PCFP. Words that are of sufficiently high frequency as to be directly stored and retrieved from memory are not subject to the PCFP (although, as Bybee (1995) observes, memory is not a dictionary; speakers may use relationships within the lexicon to aid retrieval even of word forms for which they have extensive experience). If we assume a memory-rich model of word storage, abstractive WP models thus seem to predict that learnability will have the potential to enforce low I-complexity only when words cannot be retrieved directly from memory and are therefore subject to the PCFP.

Taken together, these distributional facts raise further questions about the connection between the learnability of individual forms and the I-complexity of inflectional systems. The model of morphological knowledge that is assumed by abstractive WP models, combined with the Zipfian distribution of word tokens, predicts that the PCFP should be more challenging in some languages than others, with implications for morphological typology. Recent work by Cotterell *et al.* (2018a) moves in the direction of exploring these implications. Based on calculations for thirty-one languages, they argue that inflectional systems may be high in either E-complexity or I-complexity, but not both. The presumption is that in inflectional systems with small paradigms, each paradigm cell will be attested more often on average, as compared to an inflectional system with large paradigms, all else being equal. Essentially, the same amount of semantic 'space' (and thus, presumably, usage) is being divided among more inflected forms in the lat-

ter. This leads to cross-linguistic differences in the extent to which the PCFP presents challenges in learning. We might expect fewer constraints on I-complexity in languages where the PCFP presents less of a challenge (i.e. languages with small paradigms, and thus low E-complexity). Cotterell *et al.* argue that this prediction is borne out in the data.

Abstractive WP models also predict language-internal differences: some subparts of an inflectional system may be more challenging for the PCFP than others. This suggests the need for fine-grained measures of learnability, as well as more nuanced hypotheses about how learnability might shape morphological typology.

3 THE NEED FOR FINE-GRAINED MEASURES OF LEARNABILITY

As discussed above, I-complexity (the difficulty of the PCFP in a particular language) has been proposed as a measure of morphological learnability. Average conditional entropy is often taken as a formal model of I-complexity. In turn, average conditional entropy is often computed by segmenting word forms into two substrings: a 'stem' and an 'exponent' or 'affix',[11] and applying four-part morphological analogy based on pairs of exponents (Ackerman and Malouf 2013). For example, Icelandic 2SG *-ir* implies 3SG *-i*. (In Section 3.2 we note differences between this notion of proportional analogy and how the concept is often employed in historical linguistics.) However, both the choice of average entropy, and calculating entropy based only on pairs of exponents, have been criticized. In this section, we argue that continued progress towards understanding learnability-based constraints will require a more fine-grained understanding of how inflectional distributions are learned, because the difficulty of predicting a lexeme's entire paradigm (i.e. learning how to use it as a speaker) is not a direct function of the difficulty of predicting an individual form (the PCFP). Moreover, looking only at exponents can miss regularities in

[11] The segmentation process is often implemented using computational string alignment; see Beniamine *et al.* (2018) for a discussion. Stump and Finkel (2013) prefer to call the lexically-specific part of the form the 'theme' and the inflectionally-specific part the 'distinguisher', since these may not correspond to linguistically justified morphological analyses.

some inflectional systems which we believe human learners are able to exploit. Section 4 discusses how some of these troublesome cases can be addressed with more sophisticated computational models.

3.1 *Criticisms of averaging*

Averaging can conceal large differences in the predictability of individual cells; the average does not indicate whether every pairwise PCFP in a given language is somewhat unpredictable, or whether some PCFPs are very easy while others are very difficult. Thus, average conditional entropy can group together languages for which the underlying network of predictability relationships between cells are in fact quite different. There is some evidence that indeed, the typology of cell-to-cell predictability values is quite diverse: Stump and Finkel (2013) find that there is substantial cross-linguistic variation in the number of dynamic principal parts required to predict *all* inflected forms of lexemes, i.e. full paradigms, in contrast to the relatively uniform number of dynamic principal parts needed to predict a single inflected form. While this does not invalidate the average as an overall measure of morphological complexity, it does call for more sophisticated tools which can distinguish between these different kinds of systems.

Systems characterized by recurring partials are one example in which the individual PCFPs have predictability values far from the average. Recurring partials are groups of cells that divide the paradigm into implicatively coherent subsets. Within each subset, cells predict one another especially well, but they predict cells outside their subset poorly. The more deeply these subsets divide the paradigm, the more principal parts will be required to reproduce it.[12] Averaging the pairwise values suggests that the PCFP in these systems has an intermedi-

[12] In the extreme case, where interprediction is perfect within subsets and impossible between them, predicting the full paradigm requires exactly as many principal parts as there are subsets. Large numbers of principal parts may also be required to describe languages with cross-classifying inflection class subsystems, such as Chiquihuitlán Mazatec (Jamieson 1982), Russian (Brown *et al.* 1996) and Greek (Sims 2006). In such languages, different dimensions of inflectional exponence (e.g. suffixes, inflectional stress, stem extensions) vary semi-independently of one another, so that forms which predict one inflectional dimension may not be sufficient to predict another.

ate difficulty, but this is deceptive; each individual pairwise decision is either very easy, or very hard. Cotterell *et al.* (2018a) level the same criticism from a mathematical point of view, observing that averaging the pairwise entropy values does not compute the joint entropy of the distribution but generally yields an overestimate.

Even in cases where averaging across paradigm cells yields a good description of the system's difficulty, averaging across words, or classes of words, may not. Words with high token frequency are more likely to be irregular (Bybee 2003; Corbett *et al.* 2001) – that is, they belong to inflection classes with relatively few members (low type frequency). Stump and Finkel (2013) propose that these irregulars contribute disproportionately to the difficulty of the PCFP, the so-called Marginal Detraction Hypothesis. This property holds for a variety of languages, although seemingly not universally (Sims and Parker 2016). These classes expose a trade-off between predictability and predictiveness; exponents that are unpredictable by virtue of being irregular (and thus associated with a specific class) tend to be highly predictive of the other inflected forms of the same lexeme by virtue of this same fact (Finkel and Stump 2009). Again, the average is deceptive, failing to distinguish systems with a few highly irregular classes from systems in which every word is slightly unpredictable.

3.2 *Criticisms of simplistic implementations of morphological analogy*

We now turn to criticism of four-part morphological analogy based only on pairs of exponents as a measure of predictability. We start by observing that this type of "analogy" is not quite the same concept as analogy in historical linguistics (Hock and Joseph 1996: 10) or exemplar models (Skousen 1989). The issue has to do with abstracting away from stems and modeling only the relationship among exponents. Analogical inflectional change is not always sensitive to similarities between whole word forms,[13] but sometimes it clearly is. It

[13] Hock (1991: 172) suggests that the spread of English plural *-s*, e.g. *kine* to *cows*, should be considered a result of proportional analogy on the model of, e.g., *stone-stones*. This extension of *-s* to new words was not dependent on the overall phonological similarity of the words which gained plural *-s* to existing words with *-s*. Regularizations of this sort have sometimes been treated as simplification of the rule system (Kiparsky 1968), as something distinct from analogical change, but a distinction between rule-based change and analogical change is not even

is well known that in English, some classes of irregular verbs have attracted the occasional new member based on whole word similarity (e.g. the historically weak verb *string* changed to the *string-strung* pattern on the model of *swing-swung, sting-stung, sling-slung*, etc.). In historical linguistics (also in exemplar models), analogy is thus generally conceptualized as based on relationships among whole words.

The analogical computations of Ackerman and Malouf (2013) rely only on similarity between pairs of exponents (distinguishers), without taking stems (themes) into account. While tractable methodologically, this leads to a number of issues. We present cases in which inflectional forms are predictable, or partly predictable, on the basis of whole-word information. Importantly, the issue is not just that morphological analogy can overestimate the difficulty of the PCFP, but that the overestimation problem is likely to be larger for some languages than for others, so that the overly simplistic implementation based on pairs of exponents give an unrealistic description of the typological space. Baerman (2014) suggests that there is a typologically interesting class of languages in which information beyond what is captured by this narrow notion of morphological analogy contributes heavily to determining exponence; such a conjecture is difficult to test at a large scale without a model which is capable of exploiting these regularities as it learns to predict inflectional forms.

Several previous studies have emphasized the importance of stem information to predicting exponence in particular inflectional systems. Verb conjugation class membership in Italian (Albright 2002b) and English past tense verb forms (Albright and Hayes 2002; Bybee and Moder 1983; Rumelhart and McClelland 1986) are included among many other cases. It is thus necessary to attend to the syntagmatic dimension when modeling the predictability of inflectional exponence. In fact, Baerman (2014) argues that in Võro, a variety of Estonian, inflectional exponents are predictable *predominantly* from stem shape (specifically, how stem alternants are distributed in the paradigm), and that the exponents of other inflected forms of the same lexeme are uninformative. Morphological stem shape can also matter: the inflection class that a lexeme belongs to may be predictable from its

possible in some theoretical frameworks and we see no good motivation for it in this case.

derivational morphology. For example, in Croatian, abstract nouns derived from adjectives with *-ost* always belong to the feminine Class III, even though nouns whose stems end in a consonant normally fall into the masculine Class I. Capturing this syntagmatic dimension of predictability requires a model to be sensitive to stem shape.

Another property not captured by analogy is the re-use of affixes in different parts of the paradigm: in Kashmiri, the same set of suffixes express the remote past in one inflection class and the recent past in another (Stump and Finkel 2015). This is one example of what Baerman *et al.* (2017) call 'distributional' systems, in which inflection class distinctions are instantiated not by different exponents themselves, but by the distributions of exponents among paradigm cells. Analogy-based entropy calculations treat the different paradigm cells as separate random variables; the PCFP becomes harder when two classes share the same affix in the same paradigm cell (since this makes it harder to predict the realizations of other cells). Occurrences of the same affix in *different* cells are not modeled, either as a source of potential confusion or a regularity which the learning mechanism can exploit.

Finally, some systems have predictable relationships between cells, even where the content of those cells is unpredictable. For instance, a small number of Croatian nouns, for instance JAJE 'egg', have weak stem suppletion in oblique singular cases but not direct singular cases (*jaj-e* 'egg-NOM.SG' but *jajet-u* 'egg-DAT.SG'). This arguably makes the PCFP easier for learners of Croatian – faced with a new noun, they may be unable to guess whether it is (weakly or strongly) suppletive or what its suppletive stem may be. However, there are no Croatian nouns where a suppletive stem applies to a miscellaneous collection of singular and plural cells. So learners can predict to some extent the set of cells in which suppletive forms are allowed to appear.

To sum up, predictability based on simplistic implementations of morphological analogy captures only the interpredictability of exponents, not whether the interpredictable forms are coherent in any sense. To us, it seems unlikely that learnability constraints operate at the level of inflectional systems as a whole, even though this is the level at which the Low Entropy Conjecture and similar proposals have been formulated. It seems more likely that any upper bound on how

complex inflectional systems can be is an emergent property that derives (at least partly) from the learnability of individual inflected forms in the context of their local relationships to other inflected forms. The distributional patterns highlighted in this section identify deficiencies with previous single-measure estimates of I-complexity, and show the need for a more fine-grained approach. Sophisticated tools are therefore needed to model acquisition at this fine-grained level, and to explore its implications for morphological typology.

4 COMPUTATIONAL TASKS AND METHODS

Studies of the PCFP, whether using simple or sophisticated tools, are inherently statistical in nature; their conclusions depend on the data and on how well the data can be modeled. Thus, the advent of larger datasets and better systems for computational morphology are well-placed to make theoretical contributions to this field of study. In the next section, we discuss "morphological reinflection" as a computational formalization of morphological predictability. This approach is theoretically underpinned by abstractive WP models, making it well suited to investigation of the PCFP and the typological questions that stem from it. We summarize arguments that this formalization captures forms of predictability which are accessible to human learners, but were not measured by previous formal models such as that of Ackerman and Malouf (2013). By changing the way we estimate predictability to more closely conform to the human learner, we have the potential to change our current understanding of the morphological typology of the world's languages and its relationship to learnability.

In addition to refining our estimates of morphological complexity, we discuss ways in which computational models can be used to more precisely locate potential sources of learning difficulty within inflectional morphology. In other words, the models can tell us not only whether a particular *system* is easy to learn, but which *forms*, *classes* or other elements of the system contribute to its difficulty, and what errors we might expect an imperfect learner to make in acquiring them. We discuss these issues in the following sections.

4.1 *Reinflection with sequence-to-sequence models*

(Re)inflection tasks[14] involve converting one surface inflected word-form into a target form of the same lexeme, as illustrated in Table 3 for the German noun *Aak* (a kind of boat) and verb *aalen* 'to hunt eels, to relax' (Cotterell *et al.* 2016). The morphosyntactic values of the target are known, so reinflection amounts to predicting an inflected form for a known paradigm cell, given another known form. The task is thus equivalent to the PCFP and fits well into the theoretical framework of WP morphology. Of course, this equivalence comes with a few methodological caveats: most reinflection models are trained on orthographic, rather than phonemically transcribed data. And the datasets used are traditionally word lists, which do not reflect the token or type frequencies of the natural language, an issue we discuss in detail in Section 5. The ACL Special Interest Group on Computational Morphology and Phonology (SIGMORPHON) has sponsored a series of such tasks for shared use (Cotterell *et al.* 2016, 2017, 2018c), motivating the recent development of highly effective reinflection systems.

Table 3: Example German reinflection problems from SIGMORPHON 2016 tasks 1 and 3. In task 1, above the line, the input form is a citation form; in task 3, below, the input form is arbitrary

Input	Target features	Target
Aak	pos = N, case = NOM, gen = NEUT, num = SG	Aak
aalen	pos = V, tense = PST	geaalt
Aakes	pos = N, case = NOM, gen = NEUT, num = SG	Aak
aaltet	pos = V, tense = PST	geaalt

Broadly speaking, these systems fall into the machine learning framework of sequence-to-sequence modeling. Such models map one sequence of tokens to another sequence, which is potentially longer or shorter. The source and target tokens need not line up one-to-one, a useful property (perhaps an indispensable one) for modeling morphology. Older reinflection models in the sequence-to-sequence framework

[14] Properly speaking, "inflection" suggests the prediction of variable target forms from a fixed base, as in SIGMORPHON task 1, while "reinflection" suggests the prediction of one arbitrary form from another, as in tasks 2 and 3. For convenience, we discuss both task settings as "reinflection".

operated by inducing string edit rules (Durrett and DeNero 2013; Albright 2002b) or transductions (Nicolai *et al.* 2015). Current models use a *neural network* sequence-to-sequence (encoder-decoder) framework, which was devised for machine translation (Sutskever *et al.* 2014; Bahdanau *et al.* 2014), but which later also proved capable of learning inflectional morphology (Faruqui *et al.* 2016; Kann and Schütze 2016; Aharoni and Goldberg 2017; Malouf 2017).

Sequence-to-sequence models can be thought of as relying on whole-word analogy, just as exemplar models (Skousen 1989) do. But unlike traditional exemplar models, they induce their own, implicit, similarity function between examples. The models project the input form and featural specification of the desired output into a latent space described by a set of numerical features. The space is "latent" in that its features have no pre-specified interpretations, but reflect whatever information about the input the system finds most useful for producing the correct output. This space defines the implicit similarity metric, by clustering related inputs near one another. From its latent representation, the system can extract the output character by character. This architecture does not enforce a clear separation between stems and affixes, nor between phonological and morphological conditioning; because the output is produced character-by-character, rather than assembled from concatenated pieces, sequence-to-sequence models can learn to capture dependencies between each output character, the preceding outputs, and the *entire* input string.

This representational flexibility makes neural sequence-to-sequence models appealing as formal models of the WP morphological framework. Because the latent representation does not make implicit assumptions about morphemes as one-to-one form-meaning mappings, they are capable of learning nonconcatenative morphological processes, as discussed in Section 2.2. Cotterell *et al.* (2018a) point this out as an advantage of the "a-morphous" sequence-to-sequence framework over approaches relying on morpheme segmentation. Sequence-to-sequence models are also capable of learning stem-affix relationships, both morphological and phonological, as discussed in Section 3. Faruqui *et al.* (2016) illustrates this for the case of Finnish vowel harmony (see also Corkery *et al.* 2019); many earlier models with explicit morpheme segmentation were forced to represent this process as suppletive allomorphy (for example, positing the two phonological

variants of the inessive suffix, *-ssa* and *-ssä*, as suppletive allomorphs), which could lead to overassessment of the system's complexity (Stump and Finkel 2015). But the sequence-to-sequence model learns a generalizable harmony rule. Similarly, sequence-to-sequence models can learn a fully general process of reduplication (Prickett *et al.* 2018) where earlier models would be forced to memorize a separate rule for each reduplicating substring.

Kirov and Cotterell (2018) argue that these models remedy many of the shortcomings of earlier neural networks for inflection (Rumelhart and McClelland 1986), which were criticized (Pinker and Prince 1988; Pirrelli *et al.* 2015; Lignos and Yang 2018) both for inaccurate predictions, and for using representations which obscured the sequential nature of the input and were thus incapable of learning common typological patterns such as position class systems.

Due to these advantages, sequence-to-sequence models have been applied to several theoretical questions. Cotterell *et al.* (2018a) use them to measure I-complexity without relying on simplistic variants of four-part morphological analogy. Cotterell *et al.* (2018b) use them to simulate acquisition-based change in predicting the regularization of English past tense verbs. In doing so, projects like these implicitly assume that the SIGMORPHON reinflection task can be treated as a model of morphological acquisition and the sequence-to-sequence neural net as a model of the learner.

Thinking of sequence-to-sequence models in this way opens up several questions which deserve further attention. First is the issue of qualitative evaluation: which parts of the morphological system can the model acquire, and which does it struggle with? Second is the interpretation of the reinflection task as a model of acquisition: what sorts of datasets are appropriate and what sorts of task settings are realistic as representations of the data that humans (adults or children) encounter and the tasks that they face? Third is the question of how the model can be used to predict language change and language typology. What kinds of languages are predicted to be learnable, how does the input affect morphological learning, and what are the consequences for typological distributions, especially of inflectional systems?

5 EVALUATING MODELS' MORPHOLOGICAL KNOWLEDGE

The performance of reinflection models is generally measured in terms of percentage accuracy of the predicted output strings.[15] This yields a single number per language – usually a fairly high one. For the purpose of comparing different model variants, this kind of evaluation is sufficient, but as a tool for understanding the morphological system, it falls short. In fact, this kind of evaluation is vulnerable to the criticisms of other measurements of morphological complexity which attempt to distill the question of "how learnable" a system is into a single number (Section 3). A few other studies (Gorman *et al.* 2019; King *et al.* 2020) make the same criticisms and propose techniques for error analysis similar to the one outlined below. Malouf (2017) and Corkery *et al.* (2019) perform a different, and complementary, analysis of what is learned, by plotting reduced-dimensional projections of the model's latent space.

We argue here for a fine-grained, linguistically sophisticated analysis of model errors. Counterintuitively, we argue that such an informative error analysis requires the very theoretical concepts which the model was designed to do without: "inflection class" and "exponent". As a simple case study, we present an experiment using Latin nouns, one of the best-studied examples of an inflection class system (Carstairs-McCarthy 1994; Stump and Finkel 2015; Beniamine *et al.* 2018). Latin nouns inflect for case (6 cases, not counting the rare locative) and number (singular and plural) for a total of 12 paradigm cells per noun. We use a Pytorch (Paszke *et al.* 2017) implementation of Kann and Schütze (2016)[16] on the nouns from Latin Unimorph (Kirov *et al.* 2018),[17] training for 5 epochs (passes over the training data) of stochastic gradient descent. The dataset contains 8342 nominal paradigms; we hold out 10% of the lexemes as a development set and another 10% for testing. This training/testing procedure is the

[15] The Levenshtein edit distance on character strings is also reported, but exact match is more useful; under an edit metric, a system can achieve fairly high scores by copying the input form, and the performance of this baseline varies across languages based on the relative lengths of stems versus affixes.

[16] `https://github.com/DavidLKing/MED-pytorch`

[17] Unimorph 1.0, accessed 5 December 2018.

one used in SIGMORPHON 2016 (Cotterell *et al.* 2016), and is the most standard from a machine learning point of view. It has been criticized on cognitive grounds, since it gives the learner access to complete paradigms for 90% of the words, a point we return to in our discussion of acquisition below.

As in SIGMORPHON task 1, we inflect a citation form to produce the other forms. The choice of input form is important for task performance; Stump and Finkel (2015) show that Latin nouns have 4 static principal parts; in other words, that if a single set of paradigm cells must predict the output cell, it would have to consist of 4 members per lexeme. We instead use the NOM.SG, which is the conventional dictionary head word, but not a particularly predictive paradigm cell – thus, the system will be forced to guess at the class memberships for some of the words.[18] Our measure of success is exact match accuracy.

The accuracy of the model on test is 86%; the accuracy on the validation data is 93%. This is comparable with previously reported results using this model and this type of task (though far from the current state of the art – see Cotterell *et al.* 2018c), and is high enough to render the model useful for some practical applications in language generation (King and White 2018). But what is the system getting wrong? A first attempt at an error analysis (on the development set) is to compute error counts by paradigm cell (Table 4). This shows that the VOC.SG has much lower error than the other cells, which is unsurprising since VOC.SG is identical to NOM.SG unless the noun ends in *-us* or *-ius*.[19] But the remaining errors are distributed relatively evenly across the cells.

[18] It is theoretically relevant whether morphological systems have defined bases, and if so, whether the base is one surface form (Albright 2002a), many (as assumed by some abstractive WP models; Kann *et al.* 2017), or an abstract stem (as assumed, more or less, by constructive WP models; Cotterell *et al.* 2015; Stump 2001). This question could be evaluated in this framework. If the stem is abstract, it is also theoretically relevant how to decide what is part of the stem and what is part of inflectional exponence (Spencer 2012; Beniamine *et al.* 2017). Our choice here is theoretically unmotivated and merely represents a common way of constructing a reinflection task.

[19] Why does the system ever make errors in the (easy) vocative singular? Some of these reflect inconsistency in the training data, as also noted by Gorman *et al.* (2019), but the system also applies two generalizations somewhat inconsistently: Nouns in *-us* take *-e*, but not all nouns ending in *-s* do so: *rinoceros* 'rhinoceros'

Case	SG errors	PL errors
NOM	–	59
GEN	56	71
DAT	58	57
ACC	55	62
ABL	61	57
VOC	28	57

Table 4:
Errors by paradigm cell

We can move beyond this by examining the inflectional microclasses. Taking for granted that Latin nominal inflections are entirely suffixing, and without any model of phonological alternations, we construct a function to assign each lexeme to a class: we compute a stem by taking the longest common initial string across all inflected forms of the lexeme, and a signature which is a list of 'suffixes' (that is, everything except the stem) ordered by paradigm cell. This procedure is a simplification of existing approaches designed for more complex problems (Gaussier 1999; Goldsmith 2001; Durrett and DeNero 2013) and for calculations of inflectional complexity (Stump and Finkel 2013).[20] The system finds 272 classes in the full dataset; the most common class in our data is exemplified by GUTTA 'drop'.[21] (Many of the smaller "classes" represent compound words like *dies Martis* 'Tuesday' for which the assumption of suffixation is erroneous.)

Of these classes, 79 occur in the development set, but only 11 of them are represented by 9 or more lexemes. Table 5 shows each one with its error rate. Interestingly, several of the most common classes have error rates of around 1%, while the 5 least frequent have rates between 10–20%. This is consistent with the observation that inflected forms within small classes are unpredictable as a result of irregularity. It can also be construed as consistent with the Marginal Detraction Hypothesis (Stump and Finkel 2013), inasmuch as small classes are disproportionately responsible for whatever total amount of unpredictability is found in the inflectional system. Again, it is possible to

rather than *rinocere*; nouns in *-ius* take *-ī* rather than *-e*: *sēcrētārī* 'secretary' rather than *sēcrētārie*.

[20] As noted earlier, Stump and Finkel (2013) call the initial string the 'theme' and the remainder the 'distinguisher' to make it clear that while these subparts of the word may correspond to a stem and affix in theoretical terms, they need not.

[21] With suffixes *-īs, -ā, -ās, -am, -īs, -ae, -ārum, -ae, -ae, -a, -ae, -a*.

look deeper by examining the actual erroneous outputs. In each case, we compute a suffix for the errorful form and ask which microclasses (if any) it *could* have belonged to.[22] Many of the erroneous forms cannot be assigned to any microclass and cannot therefore be easily interpreted. These include cases where the system produces sequences with no linguistic interpretation (*honōrificābilitūdinitās* to *honōrifiiilisitūditās*) – an occupational hazard of running recurrent neural networks on long strings – and other cases where the stem is incorrectly copied.

In the remaining instances, the word receives an ending that is legitimate elsewhere in the language but not for the lexeme in question. These include mispredictions of stem elements that are neutralized in the NOM.SG: the NUTRIX 'nurse' class contains nouns whose NOM.SG ends in *-ix* and whose stem ends in *-ic*. Members of this class are frequently misinflected like HARUSPEX 'diviner', which has *-ex* in the nominative and *-ic* elsewhere,[23] and GREX 'herd' (not shown in the chart), whose stem ends in *-eg*.[24] They also include cases where the nominative form does not identify the inflection class: the SENATUS 'Senate' class (part of the traditional 4th declension) has an *-us* suffix in NOM.SG and an *-u* stem vowel. The *-us* suffix is also consistent with the much more common class of *-o*-stem nouns like ASELLUS 'donkey.DIM', and the system is not always capable of telling the difference.[25]

While the overall performance number tells us little about what is difficult in Latin nominal inflections, and the featural analysis not much more, it is possible to learn something about the system by examining the sequence-to-sequence model errors. A natural next step is to ask whether these errors tell us something about how the system is acquired (did Roman infants learn the SENATUS class later than the

[22] In Table 5, the microclasses are labeled with the citation form of an arbitrary member. When calculating confusions between classes, we restrict ourselves to alternative classes including at least 10 lexical items. A single error may be consistent with multiple overlapping classes, so the counts of 'confused classes' can exceed the total errors.

[23] Ex. *faicēs* for *faecēs* 'dregs.ACC.PL'.

[24] Ex. *quincungēs* for *quincuncēs* 'five-twelfths.ACC.PL'.

[25] Ex. *quassīs* for *quassibus* 'shaking.ABL.PL' and *Scōtibus* for *Scōtīs* 'Scot.ABL.PL'.

ASELLUS class) and how stable it might be (should we predict that the SENATUS class eventually merged into ASELLUS). We address these questions below.

Table 5: Errors for common microclasses

Class	Lexemes	Forms	Errs	Err. rate	Frequently confused classes
GUTTA	171	1881	27	0.014	none (27)
GRAVITATIO	170	1870	28	0.015	GREMIUM (9), LEXICON (9)
GREMIUM	145	1595	5	0.003	none (3), MINUTAL (1)
ASELLUS	87	957	43	0.045	none (16), SENATUS (10)
IMPERATOR	44	484	4	0.008	LITTUS (3), none (1)
GRAVITAS	41	451	32	0.07	none (11), ASELLUS (10)
NUTRIX	18	198	39	0.197	none (9), HARUSPEX (9)
GUTTUR	10	110	33	0.3	IMPERATOR (10), none (8)
SENATUS	9	99	11	0.11	ASELLUS (10), MYTHOS (10)
HOSTIS	9	99	23	0.232	none (2)
GYMNAS	9	99	25	0.253	none (11), GRAVITATIO (8)

6 ACQUISITION

Cotterell *et al.* (2018b) suggests that sequence-to-sequence models can function as cognitive models of infant language learners (though see Corkery *et al.* (2019) for some differences in behavior for nonce words). But to use a sequence-to-sequence model as a credible stand-in for the human infant, we must determine what the input for acquisition of morphology looks like – the right representation and learning algorithm cannot tell us anything if it is supplied with the wrong data. From the computational point of view, this question divides more or less neatly into two parts: first, what is the distribution of lexemes and paradigm cells in the input? And second, what information (phonological, syntactic or otherwise) is available to the learner when they hear a form?

The answer to the first question is conceptually well-known: both lexical items and paradigm cells have a Zipfian distribution (Blevins *et al.* 2017; Lignos and Yang 2018). In informal terms, natural language consists of many repetitions of the most common words, interspersed with a large population of rare words which appear a few

times each. Roughly the same is true for inflections: a natural corpus contains many repetitions of the most-used cells, but rarely-used cells are sparse in the data, and, in general, found only with common words. These distributions interact on the semantic level, so that (for instance) paired body parts like hands and eyes are commonly attested in the dual, while the dual forms for nouns like "nose" and "tooth" are relatively rare (Tiersma 1982; Bybee 1995).

Simulations in the sequence-to-sequence framework are just beginning to engage with this issue, perhaps because appropriate training data can be difficult to acquire. The datasets released by the SIGMORPHON shared tasks do not reflect the Zipfian frequency distributions of natural language. The 2016 dataset was chosen at random from Wiktionary, while the 2017 provided fewer examples, with complete paradigms for only a few words in each language. Systems trained on these datasets tend to learn from and be evaluated mostly on rare words. They have little incentive to learn about rare inflection classes, even where these contain extremely common words that make up a large percentage of child input. As already mentioned, frequency appears to be critical to the acquisition and diachronic stability of these small classes (Bybee 1995). Cotterell *et al.* (2018b), in their study of English irregulars, instead provide the system with data balanced by token frequency. This forces the system to learn irregulars like *go* ∼ *went*.

But the straightforward choice to balance the system by token frequency is also problematic, since many theories propose that learners are more sensitive to type frequency (Bybee 2003; Yang 2017; Goldwater *et al.* 2006). Typically, such theories suggest that generalization of a pattern to new items depends on the number of types to which it applies, while retention of a pattern for observed items depends on the number of tokens experienced. Bayesian models like adaptor grammars (Johnson *et al.* 2006) are capable of interpolating between types and tokens by using a separate "memory" component to store high-token-frequency training items, while some tokens of each type (logarithmically many) are treated as evidence for a general base distribution. The same process has been proposed as a model of morphological processing (Bertram *et al.* 2000; Baayen 2007). In modeling terms, two alternatives suggest themselves. One possibility is to use a conventional model, but provide it with a dataset in which a word

type whose frequency is t has $\log(t)$ tokens. The other is to add a memory component which can use the neural model as a Bayesian prior (Kawakami *et al.* 2018).

Regardless of the particular theoretical choices a researcher wishes to make, any attempt to study a real language via simulation requires access to high-quality data. Here, it is important to note that none of the SIGMORPHON datasets, nor the newer and larger Unimorph dataset (Kirov *et al.* 2018), provide an adequate set of lexical items for preparing Zipfian datasets in a large set of languages. The German Unimorph 2.0 dataset, for instance,[26] lacks paradigms for the copula *sein*, the auxiliary verbs *können, möchten, sollen, wollen*, and some commonly used content words in the child-directed inventory: *hören* 'hear', *essen* 'eat', *Hund* 'dog', etc. Many of these words exist in derived or compounded form (for instance, *Dachshund, Kampfhund, Schweinhund* are all represented), but this is unhelpful when attempting to construct a dataset which matches the token frequency of natural language, since none of these derivatives is particularly frequent. Unfortunately, the spotty coverage of high frequency words for German appears to be typical of the Unimorph datasets.

Thus, although Unimorph is an important resource for understanding morphological systems across a wide variety of languages, it is of limited use for simulations of language acquisition that seek to account for the role of frequency distributions. The easiest current option for creating frequency-matched datasets is to scavenge morphologically tagged forms from the Universal Dependencies syntactic datasets (Nivre *et al.* 2016). These do not have complete paradigms and do not represent child-directed speech; nonetheless, their coverage of commonly used forms such as auxiliary verbs is reasonably complete.

The second question, the issue of what information is available to the learner when they hear a form, is more complex. The SIGMORPHON reinflection problem, in its hardest form (task 3) is intended to model something like a "wug"-test (Berko 1958), in which an already somewhat proficient speaker of a language hears a novel word and

[26] Downloaded from `https://unimorph.github.io/` on 29 October 2019, with 179339 forms and 15060 lemmas.

then tries to produce some form of the word themselves. An English speaker, for instance, might participate in the following conversation (Berko 1958):

> A: This is a man who knows how to *spow*. He did the same thing yesterday. What did he do yesterday?
> B: Yesterday he _ (SPOW).

Here, B's role in the conversation requires them to produce a form of the abstract lexical item SPOW. In order to do so, they must guess that A's production *spow* is the V.NFIN form, and then infer the corresponding V.PST. But this description of B's mental processes assumes a relatively mature grammar of English, in which B already knows that *how to* _ is a good context for the nonfinite English verb, that English marks the past tense differently from the nonfinite, but that it is not necessary to mark person or number in the past tense, etc. All that is missing is the exponence, that is, the actual surface form which occupies the cell V.PST, thus the conventional description of this task as a paradigm cell *filling* problem.

For the developing language learner, however, this problem setup assumes too much. The learner does not start off knowing which features of the context are relevant to determining the form of SPOW, or which abstract features are active for the desired output form, or even which surface forms in dialogues like this belong to the same lexeme! This more complex problem can be viewed as one of paradigm cell *discovery*.[27] But how can a Paradigm Cell Discovery Problem (PCDP) be modeled computationally?

One possibility is the cloze task described in the SIGMORPHON 2018 shared task (Cotterell *et al.* 2018c). In this task, the output slot is described in terms of a sentence frame rather than a set of abstract features. However, the sentence frame representation has a serious problem in that it does not always specify the semantic features of the output. In the dialogue above, it is clear from the auxiliary verb *have* that the output has to express PST tense. But in many sentences, the morphological marker is the only expression of the property – in the sentence "The sun SHINE on the TREE", both *shines* and *shone*, and

[27] Boyé and Schalchli (2019) independently raise essentially the same issue, which they call the Paradigm Cell Finding Problem.

both *tree* and *trees*, are acceptable. SIGMORPHON 2018 deals with this by allowing both answers to be accepted. But from a learning standpoint, this is not reasonable; one response is presumably more faithful to the world context, the conversational common ground, and the speaker's own mental representation of the event than the other. It is not clear how such problems ought to be addressed. Visual language grounding (Kamper *et al.* 2017: among others) is a good source of information about objects and their properties, but probably cannot be used to learn features of verbs such as IRREALIS or REMOTE PAST, since abstract verb semantics are mostly inaccessible from visual context alone (Gillette *et al.* 1999; Papafragou *et al.* 2007). An effective solution to this problem probably cannot depend solely on learning semantic/surface correspondences, but requires attention to the structure of the surface morphological system itself (the *morphome*; cf. Maiden 2005; Aronoff 1994). In a morphomic analysis, the learner tries to determine how many different exponences each lexeme seems to have and their distributions, without necessarily assuming that each one corresponds to a coherent set of semantic meanings. Dreyer and Eisner (2011) models this process by clustering surface forms into lemmas and paradigm cells, at the same time inducing a set of morphological processes which relate the cells. But Dreyer's "cells" are purely formal groupings with no syntactic or semantic interpretation.

At the same time, the learner must do a realizational analysis using grounding and linguistic context to determine what external factors seem to license inflectional variations. The two analyses may not match; in some cases, as with complex tense/aspect distinctions, the surface differences between two forms may be much more salient than the distributions. In cases of syncretism, on the other hand, the surface forms are identical across two paradigm cells which nonetheless express different abstract features, and this can only be noticed on the basis of distributional evidence. The goal of a PCDP model must be to reconcile the two analyses by determining the interface between them.

A PCDP model, therefore, cannot be evaluated solely on the basis of a cloze task, since this will fail to test the model's ability to distinguish between too many feature pairs in most contexts. It should also function as a morphological part of speech tagger and can be evaluated

in that respect.[28] Given a form in context, it should be able to categorize it consistently by labeling it as an instance of an abstract paradigm cell, and perhaps even assigning it some latent semantic dimensions. These can be compared with the results of existing unsupervised POS taggers (Christodoulopoulos *et al.* 2010).

One way forward might be to augment existing grammar induction models for untagged word strings (Seginer 2007; Jin *et al.* 2018; He *et al.* 2018) to assemble words into morphological paradigms. These models are already constructed to predict correlations between words at the sentence level by positing syntactic relationships between them; equipping them with a model for morphological variation (Dreyer and Eisner 2011; Silfverberg *et al.* 2018) would allow them to model the morphosyntactic interface. Taking the nominative/accusative cases for example, if the case markings are relatively consistent and regular, and the statistical properties of the relationship between the subject/object and the verb are also relatively consistent, the grammar induction system should find distinguishing these cases beneficial to predicting sentence structure. It is unclear how well such an approach would work. Current grammar induction systems do not always induce linguistically plausible grammars. For systems that induce phrase structure grammars, morphological agreement features must be conveyed by aggressively subcategorizing syntactic categories (Petrov *et al.* 2006), which greatly increases the size of the model to be induced. Nevertheless, a combined model of this type might serve as a useful baseline for the PCDP.

7 CHANGE

Models of acquisition test how well a single learner can discover the rules of the system, given data produced by actual speakers of the language. But the language learners of today are the language users of tomorrow; a natural extension of the learning simulation is to make the output from one generation of computational learners serve as

[28] Taggers which use fine-grained, multidimensional tags to indicate all the morphosyntactic properties of a particular word token are generally trained in the supervised setting (Chrupała *et al.* 2008; Müller *et al.* 2013); for this task, it would be necessary to apply this fine-grained standard of evaluation to unsupervised models.

training data for another, observing how the system changes over time (Kirby and Hurford 1997, 2002). Such iterated learning experiments have been used to study the emergence and disappearance of irregular forms in both simulated (Ackerman and Malouf 2015; Parker *et al.* 2019) and real (Hare and Elman 1995; Cotterell *et al.* 2018b) datasets, and to study the spread and loss of different languages or linguistic features in a social network (Abrams and Strogatz 2003; Castelló *et al.* 2013).

Iterated language change simulations tend to take one of two perspectives on language change, modeling change as arising either from acquisition or from usage. Models of acquisition-based change treat most differences between generations as cases of imperfect acquisition: due either to data sparsity or to biased hypothesis selection, the "children" do not acquire the same language as the "parents". Sparsity and a preference for regularity lead the system to regularize, eliminating irregular forms and merging inflection classes (Kalish *et al.* 2007; Reali and Griffiths 2009); this is also a typical outcome in Ackerman and Malouf (2015), though some simulations do lead to large numbers of inflection classes. A preference for distinctiveness, on the other hand, can lead to the maintenance or even the creation of irregularity, since irregular forms are compact and easy to recognize (*go* ∼ *went* rather than *goed*; Dale and Lupyan 2012). In any case, these models see change as primarily arising from learning.

Many such models make the same incorrect prediction: morphological change should be rapid and common, and it should work to eliminate "non-functional" parts of the system, such as inflection class, which do not correspond to any abstract meaning. In fact, the typological pattern is the opposite; many real morphological systems have these non-functional elements, and while individual words may move from class to class, the classes can be remarkably stable across long periods of historical time. As Harris (2008: 66) says, "there is apparently no need of repair; the system works and can be acquired... there is nothing about our innate endowment that demands that a language simplify". Although some elements of a morphological system may take years to reach adult-like competence (Xanthos *et al.* 2011; Forshaw *et al.* 2017), given enough exposure, learners will eventually produce it with high fidelity. The preference for over-regularization observed in child learners may be a relatively temporary phase of

development (Maratsos 2000; Ambridge *et al.* 2013; Joseph 2011) which does not normally cause sweeping changes in the adult system (for an opposing viewpoint, see Huang and Pinker 2010). Additionally, the Natural Morphology framework (Dressler 2003; Wurzel 1989, 2000) emphasizes the role of languages' system-defining structural properties in making their morphological systems conservative when it comes to language change (Wurzel 1989: 104). From this perspective, the pressure in language change is towards greater system congruity, not necessarily towards elimination of inflection classes or other "non-functional" parts of the system.

Sociolinguistic models, on the other hand, see change as primarily a result of biased language usage, with significant language changes occurring throughout the lifetime (Labov 2007). In this kind of model, users make both conscious and unconscious decisions about what language features to use (Milroy 2007). For instance, in a model of language change in Spain (Castelló *et al.* 2013), agents in a social network speak either prestigious Castilian or stigmatized Galician. Speakers may switch languages in either direction, based on how many of their neighbors in the network they will be able to communicate with and how socially prestigious they will become. For most network topologies, Galician will eventually be lost entirely. This is not an effect of learning biases: in a model of this type, there is no difference in learnability between the systems; rather, it is taken for granted that agents could, in principle, acquire either system perfectly, if it proved to be worth the social investment.

In the case of morphological systems, change is likely to be a combination of both learnability and prestige and other social factors. While any linguistic variability is likely to gain some amount of social evaluation, some morphological variables within a population seem relatively unmarked (perhaps because they apply primarily to unfamiliar words; Dąbrowska 2008), while others attract widespread attention and stigmatization. This is the idea behind Labov's division of socially-relevant linguistic variables into indicators, markers, and stereotypes (Labov 1971, 2001). At the same time, however, the system may provide the learner with varying degrees of evidence for the different forms. These conflicting pressures are probably responsible for selecting among the possible outcomes. For example, Jutronic (2001) describes competition between two dialects of Croatian in the

city of Split; for instance, the local dialect form *profešuri* competes with standard *profesora* to realize 'professor.PL.GEN'. Dialect contact of this kind can eventually converge on one of the two original systems, but can also give rise to a more complex system in which both variants are analyzed as morphological markers (Trudgill 2011: 27).

In many cases, however, the real impact of social factors is to cause changes to the morphology indirectly, through their impact on other linguistic subsystems. For instance, socially conditioned phonological change can cause a reorganization of the inflectional system. On the one hand, sound change can destroy morphological distinctions, by merging or eliminating affixes. Where distinctions are not leveled outright, it can change which elements of the surface string act as markers for a morphological feature, raising a phonological alternation to the level of an exponent. On the other hand, processes of phonological reduction and grammaticalization can create new morphemes, as in the evolution of the French adverbial suffix *-ment* from the Latin noun *mente* 'mind.ABL' (Joseph 2003).

The real impact of learning biases in morphological change may be felt primarily in determining how a language reacts to this kind of disruption. Harris (2008) argues that the Georgian pattern in which the same case endings indicate different semantic roles for different classes of verbs (Series I vs II) results from the historical development of the Kartvelian languages from true ergative languages to split-ergative alignment. The Series II verbs began as a productive antipassive construction which was lost along with ergativity, but became "frozen" in the language as a morphologically complex relationship between classes of verbs and surface case markers. Harris argues that while some languages undergoing this kind of change converge on a single consistent set of case markers, the salience of the Georgian markers prevented this kind of mis-learning. In other words, as the language changed, the older meaning of the construction became too opaque for learners to acquire it, creating the potential for two eventual outcomes: one in which the new surface pattern persisted and one in which it was regularized. It was in this situation that the phonological distinctiveness of the markers themselves (and perhaps other acquisition biases) became important in determining which system would be learned.

In other examples, the external pressures on the system come from bilingualism or adult language learning. A wide variety of studies involving bilinguals can be interpreted as demonstrating these kinds of change, which can lead the system either to lose or to gain morphological features. Dorian (1978) shows the loss of features in a dying variety of Scots Gaelic in the process of being replaced by English. The Gaelic system retains a variety of exponents of the plural and gerund, even in the last generation of speakers, but morphological processes involving *features* which English does not use (for instance lengthening) were lost. On the other hand, Lefebvre (1996) shows the introduction of new features by bilingual speakers who expect a system to express certain abstract features, and recruit L2 features as surrogate markers. This is the case in Haitian Creole, where the tense/aspect/modality, pronominal and nominal systems have been interpreted as relexification of its substrates (mainly Ewe and Fongbe) using a French superstratum. For example, Ewe has the morpheme *wò* indicating the third person plural pronoun and the plural in noun phrases. Lefebvre argues that the Haitian Creole morpheme *yo* encodes both notions and that it reflects substratum influence. While the presence of L2 speakers has been suggested as a pressure towards less enumerative morphological complexity (Trudgill 2011; Dale and Lupyan 2012; Frank and Smith 2018), understanding the actual impact of a learner population might require more insight into their L1 system and how their learned representations can be adapted to fit the state of the L2, as well as into the social circumstances under which they learn.

Whether simulating L1 or L2 learning, sequence-to-sequence models provide an interesting platform for detailed and realistic learning simulations. But these simulations need to move beyond training the system on corpora reflecting synchronic steady states, then analyzing the errors to predict incipient large-scale restructuring towards some imagined typological ideal. Typological variety is the product of the historical paths down which languages travel (Harris 2008; Anderson 2004): typologically rare morphological systems occur when a particular change (phonological, social or otherwise) interacts with a particular morphological system. By incorporating data from outside the realm of morphology, we can hope to create better models of diachronic change and better understand the circumstances under which these rare systems arise.

8 CONCLUSIONS

The main goal of this paper has been to argue that sequence-to-sequence models hold out the possibility not only of improvements in practical tasks, but also of real advances in morphological theory and typology. Alongside this promise comes the necessity to think harder about our experimental setups. We have put forward possible improvements in how the models are evaluated, in what tasks they are trained to perform and in how we extrapolate from a single learner to a community of socially motivated language users. In particular, we have argued for error analyses in terms of paradigm cells and inflectional classes, rather than dataset-wide accuracy. We have proposed using Zipfian datasets and replacing, or at least supplementing, Paradigm Cell Filling with Paradigm Cell Discovery. And we have suggested that models of morphological change reach beyond the morphological system to incorporate factors such as prestige, bilingualism and sound change.

ACKNOWLEDGMENTS

This paper grew out of a joint seminar, co-taught by Micha Elsner and Andrea Sims, in the Ohio State University Department of Linguistics in fall 2018. We thank our chair, Shari Speer, for making it possible for us to work together on a course. We also thank three anonymous reviewers for their comments.

REFERENCES

Daniel M. Abrams and Steven H. Strogatz (2003), Linguistics: Modelling the dynamics of language death, *Nature*, 424(6951):900.

Farrell Ackerman, James P. Blevins, and Robert Malouf (2009), Parts and wholes: Patterns of relatedness in complex morphological systems and why they matter, *Analogy in grammar: Form and acquisition*, pp. 54–82.

Farrell Ackerman and Robert Malouf (2013), Morphological organization: The low conditional entropy conjecture, *Language*, 89(3):429–464.

Farrell Ackerman and Robert Malouf (2015), The No Blur Principle effects as an emergent property of language systems, in Anna E. Jurgensen, Hannah Sande, Spencer Lamoureux, Kenny Baclawski, and Alison Zerbe, editors, *Proceedings of the Forty-First Annual Meeting of the Berkeley Linguistics Society*, pp. 1–14, Berkeley Linguistics Society.

Roee AHARONI and Yoav GOLDBERG (2017), Morphological inflection generation with hard monotonic attention, in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2004–2015.

Adam ALBRIGHT (2002a), *The identification of bases in morphological paradigms*, Ph.D. thesis, Department of Linguistics, University of California, Los Angeles.

Adam ALBRIGHT (2002b), Islands of reliability for regular morphology: Evidence from Italian, *Language*, 78(4):684–709.

Adam ALBRIGHT and Bruce HAYES (2002), Modeling English past tense intuitions with minimal generalization, in *Proceedings of the Sixth Meeting of the Association for Computational Linguistics Special Interest Group in Computational Phonology in Philadelphia, July 2002*, pp. 58–69.

Ben AMBRIDGE, Julian M. PINE, Caroline F. ROWLAND, Franklin CHANG, and Amy BIDGOOD (2013), The retreat from overgeneralization in child language acquisition: Word learning, morphology, and verb argument structure, *Wiley Interdisciplinary Reviews: Cognitive Science*, 4(1):47–62.

Stephen R. ANDERSON (1992), *A-morphous morphology*, Cambridge University Press.

Stephen R. ANDERSON (2004), Morphological universals and diachrony, in Geert BOOIJ and Jaap VAN MARLE, editors, *Yearbook of morphology 2004*, pp. 1–17, Springer.

Mark ARONOFF (1994), *Morphology by itself: Stems and inflectional classes*, MIT Press.

R. Harald BAAYEN (2001), *Word frequency distributions*, Kluwer.

R. Harald BAAYEN (2007), Storage and computation in the mental lexicon, in Gonia JAREMA and Gary LIBBEN, editors, *The mental lexicon: Core perspectives*, pp. 81–104, Elsevier.

Matthew BAERMAN (2012), Paradigmatic chaos in Nuer, *Language*, 88(3):467–494.

Matthew BAERMAN (2014), Covert systematicity in a distributionally complex system, *Journal of Linguistics*, 50(1):1–47.

Matthew BAERMAN, Dunstan BROWN, and Greville G. CORBETT (2017), *Morphological complexity*, Cambridge University Press.

Dzmitry BAHDANAU, Kyunghyun CHO, and Yoshua BENGIO (2014), Neural machine translation by jointly learning to align and translate, *arXiv preprint arXiv:1409.0473*.

Sacha BENIAMINE, Olivier BONAMI, and Joyce MCDONOUGH (2017), When segmentation helps: Implicative structure and morph boundaries in the Navajo verb, in *Proceedings of the First International Symposium on Morphology*, pp. 11–15.

Sacha BENIAMINE, Olivier BONAMI, and Benoît SAGOT (2018), Inferring inflection classes with description length, *Journal of Language Modelling*, 5(3):465–525.

Jean BERKO (1958), The child's learning of English morphology, *Word*, 14(2-3):150–177.

Raymond BERTRAM, Robert SCHREUDER, and R. Harald BAAYEN (2000), The balance of storage and computation in morphological processing: The role of word formation type, affixal homonymy, and productivity, *Journal of Experimental Psychology: Learning, Memory and Cognition*, 26(2):489–511.

James P. BLEVINS (2004), Inflection classes and economy, in Gereon MÜLLER, Lutz GUNKEL, and Gisela ZIFONUN, editors, *Explorations in nominal inflection*, pp. 51–95, Mouton de Gruyter.

James P. BLEVINS (2006), Word-based morphology, *Journal of Linguistics*, 42(3):511–573.

James P. BLEVINS (2013), American descriptivism ('structuralism'), in Keith ALLAN, editor, *The Oxford handbook of the history of linguistics*, pp. 419–437, Oxford University Press.

James P. BLEVINS (2016), *Word and paradigm morphology*, Oxford University Press.

James P. BLEVINS, Petar MILIN, and Michael RAMSCAR (2017), The Zipfian paradigm cell filling problem, in Ferenc KIEFER, James P. BLEVINS, and Huba BARTOS, editors, *Perspectives on morphological organization: Data and analyses*, pp. 141–158, Brill.

Harry BOCHNER (1993), *Simplicity in generative morphology*, Mouton de Gruyter.

Olivier BONAMI and S. BENIAMINE (2016), Joint predictiveness in inflectional paradigms, *Word Structure*, 9(2):156–182.

Gilles BOYÉ and Gauvain SCHALCHLI (2019), Realistic data and paradigms: The paradigm cell finding problem, *Morphology*, 29(2):199–248.

Dunstan BROWN, Greville G. CORBETT, Norman FRASER, Andrew HIPPISLEY, and Alan TIMBERLAKE (1996), Russian noun stress and Network Morphology, *Journal of Linguistics*, 34:53–107.

Dunstan BROWN and Andrew HIPPISLEY (2012), *Network morphology: A defaults-based theory of word structure*, Cambridge University Press.

Joan BYBEE (1995), Diachronic and typological properties of morphology and their implications for representation, in *Morphological aspects of language processing*, pp. 225–246, Erlbaum Hillsdale.

Joan BYBEE (2003), Mechanisms of change in grammaticization: The role of frequency, in Brian D. JOSEPH and Richard D. JANDA, editors, *The handbook of historical linguistics*, pp. 602–623, Blackwell.

Joan Bybee and Carol Moder (1983), Morphological classes as natural categories, *Language*, 59(2):251–270.

Andrew Carstairs (1987), *Allomorphy in inflexion*, Croom Helm.

Andrew Carstairs-McCarthy (1994), Inflection classes, gender, and the principle of contrast, *Language*, 70(4):737–788.

Andrew Carstairs-McCarthy (2010), *The evolution of morphology*, Oxford University Press.

Xavier Castelló, Lucía Loureiro-Porto, and Maxi San Miguel (2013), Agent-based models of language competition, *International Journal of the Sociology of Language*, 221:21–51.

Christos Christodoulopoulos, Sharon Goldwater, and Mark Steedman (2010), Two decades of unsupervised POS induction: How far have we come?, in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 575–584, Association for Computational Linguistics.

Grzegorz Chrupała, Georgiana Dinu, and Josef Van Genabith (2008), Learning morphology with Morfette, in *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*.

Eve V. Clark (1987), The principle of contrast: A constraint on language acquisition, in Brian MacWhinney, editor, *Mechanisms of language acquisition*, pp. 1–33, Erlbaum.

Greville G. Corbett, Andrew Hippisley, Dunstan Brown, and Paul Marriott (2001), Frequency, regularity and the paradigm: A perspective from Russian on a complex relation, in Joan Bybee and Paul J. Hopper, editors, *Frequency and the emergence of linguistic structure*, pp. 201–226, John Benjamins.

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 *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3868–3877, Association for Computational Linguistics, Florence, Italy, doi:10.18653/v1/P19-1376, https://www.aclweb.org/anthology/P19-1376.

Ryan Cotterell, Christo Kirov, Mans Hulden, and Jason Eisner (2018a), On the complexity and typology of inflectional morphological systems, *arXiv preprint arXiv:1807.02747*.

Ryan Cotterell, Christo Kirov, Mans Hulden, and Jason Eisner (2018b), On the diachronic stability of irregularity in inflectional morphology, *arXiv preprint arXiv:1804.08262*.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Arya D McCarthy, Katharina Kann, Sebastian Mielke, Garrett Nicolai, Miikka Silfverberg, *et al.* (2018c), The CoNLL–SIGMORPHON 2018 shared task: Universal morphological reinflection, *arXiv preprint arXiv:1810.07125*.

Ryan COTTERELL, Christo KIROV, John SYLAK-GLASSMAN, Géraldine WALTHER, Ekaterina VYLOMOVA, Patrick XIA, Manaal FARUQUI, Sandra KÜBLER, David YAROWSKY, Jason EISNER, *et al.* (2017), CoNLL-SIGMORPHON 2017 shared task: Universal morphological reinflection in 52 languages, *arXiv preprint arXiv:1706.09031*.

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

Ryan COTTERELL, Nanyun PENG, and Jason EISNER (2015), Modeling word forms using latent underlying morphs and phonology, *Transactions of the Association of Computational Linguistics*, 3(1).

Ewa DĄBROWSKA (2008), The effects of frequency and neighbourhood density on adult speakers' productivity with Polish case inflections: An empirical test of usage-based approaches to morphology, *Journal of Memory and Language*, 58(4):931–951.

Rick DALE and Gary LUPYAN (2012), Understanding the origins of morphological diversity: The linguistic niche hypothesis, *Advances in Complex Systems*, 15(03n04):1150017.

Nancy C. DORIAN (1978), The fate of morphological complexity in language death: Evidence from East Sutherland Gaelic, *Language*, 54(3):590–609.

Wolfgang U. DRESSLER (2003), Naturalness and morphological change, in Brian D. JOSEPH and Richard D. JANDA, editors, *Handbook of historical linguistics*, pp. 461–471, Blackwell.

Wolfgang U. DRESSLER, Marianne KILANI-SCHOCH, Natalia GAGARINA, Lina PESTAL, and Markus PÖCHTRAGER (2006), On the typology of inflection class systems, *Folia Linguistica*, 40(1-2):51–74.

Markus DREYER and Jason EISNER (2011), Discovering morphological paradigms from plain text using a Dirichlet process mixture model, in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 616–627, Association for Computational Linguistics.

Greg DURRETT and John DENERO (2013), Supervised learning of complete morphological paradigms, in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1185–1195.

Manaal FARUQUI, Yulia TSVETKOV, Graham NEUBIG, and Chris DYER (2016), Morphological inflection generation using character sequence to sequence learning, in *Proceedings of NAACL-HLT*, pp. 634–643.

Raphael A. FINKEL and Gregory T. STUMP (2009), Principal parts and degrees of paradigmatic transparency, in James P. BLEVINS and Juliette BLEVINS, editors, *Analogy in grammar: Form and acquisition*, pp. 13–53, Oxford University Press.

Bill FORSHAW, Lucinda DAVIDSON, Barbara KELLY, Rachel NORDLINGER, Gillian WIGGLESWORTH, and Joe BLYTHE (2017), The acquisition of Murrinhpatha (Northern Australia), in Michael FORTESCUE, Marianne MITHUN, and Nicholas EVANS, editors, *The Oxford Handbook of Polysynthesis*, pp. 473–494, Oxford University Press.

Stella FRANK and Kenny SMITH (2018), A model of linguistic accommodation leading to language simplification, *PsyArXiv preprint: https://doi.org/10.31234/osf.io/4ynwu*.

Éric GAUSSIER (1999), Unsupervised learning of derivational morphology from inflectional lexicons, in Andrew KEHLER and Andreas STOLCKE, editors, *Unsupervised learning in natural language processing: Proceedings of the workshop*, pp. 24–30, Association for Computational Linguistics.

Jane GILLETTE, Henry GLEITMAN, Lila GLEITMAN, and Anne LEDERER (1999), Human simulations of vocabulary learning, *Cognition*, 73(2):135–176.

John GOLDSMITH (2001), Unsupervised learning of the morphology of a natural language, *Computational Linguistics*, 27(2):153–198.

Sharon GOLDWATER, Mark JOHNSON, and Thomas L GRIFFITHS (2006), Interpolating between types and tokens by estimating power-law generators, in Bernhard SCHÖLKOPF, John C. PLATT, and Thomas HOFFMAN, editors, *Advances in neural information processing systems 19*, pp. 459–466, Neural Information Processing Systems Foundation.

Kyle GORMAN, Arya D. MCCARTHY, Ryan COTTERELL, Ekaterina VYLOMOVA, Miikka SILFVERBERG, and Magdalena MARKOWSKA (2019), Weird inflects but OK: Making sense of morphological generation errors, in *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pp. 140–151, Association for Computational Linguistics, Hong Kong, China, `https://www.aclweb.org/anthology/D19-6714`.

Joseph H. GREENBERG (1960), A quantitative approach to the morphological typology of language, *International Journal of American Linguistics*, 26(3):178–194.

Morris HALLE and Alex MARANTZ (1993), Distributed Morphology and the pieces of inflection, in Kenneth HALE and Samuel Jay KEYSER, editors, *The view from building 20*, pp. 111–176, MIT Press.

Morris HALLE and Alex MARANTZ (2008), Clarifying "blur": Paradigms, defaults, and inflectional classes, in Asaf BACHRACH and Andrew NEVINS, editors, *Inflectional identity*, pp. 55–72, Mouton de Gruyter.

Mary HARE and Jeffrey L. ELMAN (1995), Learning and morphological change, *Cognition*, 56(1):61–98.

Heidi HARLEY and Rolf NOYER (2003), Distributed Morphology, in Lisa CHENG and Rint SYBESMA, editors, *The Second GLOT International state-of-the-article book*, pp. 463–496, de Gruyter Mouton.

Alice C. HARRIS (2008), On the explanation of typologically unusual structures, in Jeff GOOD, editor, *Linguistic universals and language change*, pp. 54–76, Oxford University Press.

Junxian HE, Graham NEUBIG, and Taylor BERG-KIRKPATRICK (2018), Unsupervised Learning of Syntactic Structure with Invertible Neural Projections, in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1292–1302, Association for Computational Linguistics, Brussels, Belgium, `https://www.aclweb.org/anthology/D18-1160`.

Hans Henrich HOCK (1991), *Principles of historical linguistics*, Mouton de Gruyter.

Hans Henrich HOCK and Brian D. JOSEPH (1996), *Language history, language change and language relationship: An introduction to historical and comparative linguistics*, Mouton de Gruyter.

Charles F. HOCKETT (1954), Two models of grammatical description, *Word*, 10:210–234.

Yi Ting HUANG and Steven PINKER (2010), Lexical semantics and irregular inflection, *Language and Cognitive Processes*, 25(10):1411–1461.

Carole Ann JAMIESON (1982), Conflated subsystems marking person and aspect in Chiquihuitlán Mazatec verbs, *International Journal of American Linguistics*, 48(2):139–167.

Lifeng JIN, William SCHULER, Finale DOSHI-VELEZ, Timothy A. MILLER, and Lane SCHWARTZ (2018), Unsupervised grammar induction with depth-bounded PCFG, *Transactions of the Association for Computational Linguistics*, 6:211–224, `https://github.com/lifengjin/db-pcfg`.

Mark JOHNSON, Thomas L. GRIFFITHS, and Sharon GOLDWATER (2006), Adaptor grammars: A framework for specifying compositional nonparametric Bayesian models, in Bernhard SCHÖLKOPF, John C. PLATT, and Thomas HOFFMAN, editors, *Advances in neural information processing systems 19*, pp. 641–648, Neural Information Processing Systems Foundation.

Brian D. JOSEPH (2003), Morphologization from syntax, in Brian D. JOSEPH and Richard D. JANDA, editors, *The handbook of historical linguistics*, pp. 472–492, Blackwell.

Brian D. JOSEPH (2011), Children rule, or do they (as far as innovations are concerned)?, *Bilingualism: Language and Cognition*, 14(2):156–158.

Dunja JUTRONIC (2001), Morphological changes in the urban vernacular of the city of Split, *International Journal of the Sociology of Language*, 147:65–78.

Michael L. KALISH, Thomas L. GRIFFITHS, and Stephan LEWANDOWSKY (2007), Iterated learning: Intergenerational knowledge transmission reveals inductive biases, *Psychonomic Bulletin & Review*, 14(2):288–294.

Herman KAMPER, Shane SETTLE, Gregory SHAKHNAROVICH, and Karen LIVESCU (2017), Visually grounded learning of keyword prediction from untranscribed speech, *arXiv preprint arXiv:1703.08136*.

Katharina KANN, Ryan COTTERELL, and Hinrich SCHÜTZE (2017), Neural multi-source morphological reinflection, in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pp. 514–524.

Katharina KANN and Hinrich SCHÜTZE (2016), MED: The LMU system for the SIGMORPHON 2016 shared task on morphological reinflection, in *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pp. 62–70.

Shuan KARIM (2019), Competition between formatives and the diversity of ezafat, Presented at the 24th International Conference on Historical Linguistics (ICHL24).

Kazuya KAWAKAMI, Chris DYER, and Phil BLUNSOM (2018), Unsupervised word discovery with segmental neural language models, *ArXiv e-prints*, arXiv:1811.09353.

Aleksandr E. KIBRIK (1998), Archi, in Andrew SPENCER and Arnold M. ZWICKY, editors, *The handbook of morphology*, pp. 455–476, Blackwell.

David KING, Andrea D. SIMS, and Micha ELSNER (2020), Interpreting sequence-to-sequence models for Russian inflectional morphology, in *Proceedings of the Society for Computation in Linguistics (SCiL)*, Society for Computation in Linguistics, New Orleans, USA.

David KING and Michael WHITE (2018), The OSU realizer for SRST'18: Neural sequence-to-sequence inflection and incremental locality-based linearization, in *Proceedings of the First Workshop on Multilingual Surface Realisation*, pp. 39–48.

Paul KIPARSKY (1968), Linguistic universals and linguistic change, in Emmon BACH and Robert T. HARMS, editors, *Universals in linguistic theory*, pp. 170–202, Holt, Rinehart and Winston.

Simon KIRBY and James HURFORD (1997), Learning, culture and evolution in the origin of linguistic constraints, in *Fourth European conference on artificial life*, pp. 493–502, Citeseer.

Simon KIRBY and James R. HURFORD (2002), The emergence of linguistic structure: An overview of the iterated learning model, in Angelo CANGELOSI and Domenico PARISI, editors, *Simulating the evolution of language*, pp. 121–147, Springer.

Christo KIROV and Ryan COTTERELL (2018), Recurrent neural networks in linguistic theory: Revisiting Pinker and Prince (1988) and the Past Tense Debate, *arXiv preprint arXiv:1807.04783*.

Christo KIROV, Ryan COTTERELL, John SYLAK-GLASSMAN, Géraldine WALTHER, Ekaterina VYLOMOVA, Patrick XIA, Manaal FARUQUI, Sebastian MIELKE, Arya D. MCCARTHY, Sandra KÜBLER, *et al.* (2018), UniMorph 2.0: Universal morphology, *arXiv preprint arXiv:1810.11101*.

William LABOV (1971), The study of language in its social context, in Joshua A. FISHMAN, editor, *Advances in the sociology of language, vol. 1*, pp. 152–216, Mouton.

William LABOV (2001), *Principles of linguistic change, vol. 2: Social factors*, Blackwell.

William LABOV (2007), Transmission and diffusion, *Language*, 83(2):344–387.

Claire LEFEBVRE (1996), The tense, mood, and aspect system of Haitian Creole and the problem of transmission of grammar in creole genesis, *Journal of Pidgin and Creole Languages*, 11(2):231–311.

Constantine LIGNOS and Charles YANG (2018), Morphology and language acquisition, in Andrew HIPPISLEY and Gregory T. STUMP, editors, *Cambridge handbook of morphology*, pp. 765–791, Cambridge University Press.

Martin MAIDEN (2005), Morphological autonomy and diachrony, in *Yearbook of morphology 2004*, pp. 137–175, Springer.

Robert MALOUF (2017), Abstractive morphological learning with a recurrent neural network, *Morphology*, 27(4):431–458.

Michael MARATSOS (2000), More overregularizations after all: New data and discussion on Marcus, Pinker, Ullman, Hollander, Rosen & Xu, *Journal of Child Language*, 27(1):183–212.

P.H. MATTHEWS (1972), *Inflectional morphology: A theoretical study based on aspects of Latin verb conjugation*, Cambridge University Press.

Petar MILIN, Dusica FILIPOVIĆ DJURDJEVIĆ, and Fermín MOSCOSO DEL PRADO MARTÍN (2009), The simultaneous effects of inflectional paradigms and classes on lexical recognition: Evidence from Serbian, *Journal of Memory and Language*, 60:50–64.

Lesley MILROY (2007), Off the shelf or under the counter? On the social dynamics of sound changes, *Topics in English Linguistics*, 53:149.

Fermín MOSCOSO DEL PRADO MARTÍN, Aleksandar KOSTIĆ, and R. Harald BAAYEN (2004), Putting the bits together: An information theoretical perspective on morphological processing, *Cognition*, 94:1–18.

Gereon MÜLLER (2007), Notes on paradigm economy, *Morphology*, 17(1):1–38.

Thomas MÜLLER, Helmut SCHMID, and Hinrich SCHÜTZE (2013), Efficient higher-order CRFs for morphological tagging, in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 322–332.

Garrett NICOLAI, Colin CHERRY, and Grzegorz KONDRAK (2015), Inflection generation as discriminative string transduction, in *Proceedings of the 2015 conference of the North American chapter of the association for computational linguistics: human language technologies*, pp. 922–931.

Joakim NIVRE, Marie-Catherine DE MARNEFFE, Filip GINTER, Yoav GOLDBERG, Jan HAJIC, Christopher D MANNING, Ryan T MCDONALD, Slav PETROV, Sampo PYYSALO, Natalia SILVEIRA, *et al.* (2016), Universal dependencies v1: A multilingual treebank collection., in *Proceedings of LREC*.

Anna PAPAFRAGOU, Kimberly CASSIDY, and Lila GLEITMAN (2007), When we think about thinking: The acquisition of belief verbs, *Cognition*, 105(1):125–165.

Jeff PARKER (2016), *Inflectional complexity and cognitive processing: An experimental and corpus-based investigation of Russian nouns*, Ph.D. thesis, Department of Slavic and East European Languages and Cultures, The Ohio State University.

Jeff PARKER, Robert REYNOLDS, and Andrea D. SIMS (2019), The role of language-specific network properties in the emergence of inflectional irregularity, in Andrea D. SIMS, Adam USSISHKIN, Jeff PARKER, and Samantha WRAY, editors, *Morphological typology and linguistic cognition*, Cambridge University Press, to appear.

Adam PASZKE, Sam GROSS, Soumith CHINTALA, Gregory CHANAN, Edward YANG, Zachary DEVITO, Zeming LIN, Alban DESMAISON, Luca ANTIGA, and Adam LERER (2017), Automatic differentiation in PyTorch, in *NIPS 2017 Autodiff Workshop*.

Slav PETROV, Leon BARRETT, Romain THIBAUX, and Dan KLEIN (2006), Learning accurate, compact, and interpretable tree annotation, in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pp. 433–440, Association for Computational Linguistics.

Steven PINKER and Alan PRINCE (1988), On language and connectionism: Analysis of a parallel distributed processing model of language acquisition, *Cognition*, 28(1-2):73–193.

Vito PIRRELLI, Marcello FERRO, and Claudia MARZI (2015), Computational complexity of abstractive morphology, in Matthew BAERMAN, Dunstan BROWN, and Greville G. CORBETT, editors, *Understanding and measuring morphological complexity*, pp. 141–166, Oxford University Press.

Brandon PRICKETT, Aaron TRAYLOR, and Joe PATER (2018), Seq2Seq models with dropout can learn generalizable reduplication, in *Proceedings of the*

Fifteenth Workshop on Computational Research in Phonetics, Phonology, and Morphology, pp. 93–100, Association for Computational Linguistics, Brussels, Belgium, doi:10.18653/v1/W18-5810, https://www.aclweb.org/anthology/W18-5810.

Florencia Reali and Thomas L Griffiths (2009), The evolution of frequency distributions: Relating regularization to inductive biases through iterated learning, *Cognition*, 111(3):317–328.

David E. Rumelhart and James L. McClelland (1986), On learning the past tenses of English verbs, in James L. McClelland, David E. Rumelhart, and PDP Research Group, editors, *Parallel distributed processing: Explorations in the microstructure of cognition, vol. 2: Psychological and biological models*, pp. 216–271, MIT Press.

Yoav Seginer (2007), Fast unsupervised incremental parsing, in *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pp. 384–391.

Miikka Silfverberg, Ling Liu, and Mans Hulden (2018), A computational model for the linguistic notion of morphological paradigm, in *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 1615–1626.

Andrea D. Sims (2006), *Minding the gaps: Inflectional defectiveness in paradigmatic morphology*, Ph.D. thesis, Department of Linguistics, The Ohio State University.

Andrea D. Sims and Jeff Parker (2016), How inflection class systems work: On the informativity of implicative structure, *Word Structure*, 9(2):215–239.

Royal Skousen (1989), *Analogical modeling of language*, Springer Science & Business Media.

Andrew Spencer (2012), Identifying stems, *Word Structure*, 5(1):88–108.

Gregory T. Stump (2001), *Inflectional morphology: A theory of paradigm structure*, Cambridge University Press.

Gregory T. Stump and Raphael A. Finkel (2013), *Morphological typology: From word to paradigm*, Cambridge University Press.

Gregory T. Stump and Raphael A. Finkel (2015), The complexity of inflectional systems, *Linguistics Vanguard*, 1(1):101–117.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le (2014), Sequence to sequence learning with neural networks, in Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in neural information processing systems 27*, pp. 3104–3112, Neural Information Processing Systems Foundation.

Haukur Þorgeirsson (2017), Testing Vocabular Clarity in insular Scandinavian, *Folia Linguistica*, 51(3):505–526.

Peter Tiersma (1982), Local and general markedness, *Language*, 58(4):832–849.

Peter TRUDGILL (2011), *Sociolinguistic typology: Social determinants of linguistic complexity*, Oxford University Press.

Wolfgang U. WURZEL (1989), *Inflectional morphology and naturalness*, Kluwer.

Wolfgang U. WURZEL (2000), Inflectional system and markedness, in Aditi LAHIRI, editor, *Analogy, levelling, markedness: Principles of change in phonology and morphology*, pp. 193–214, Mouton de Gruyter.

Aris XANTHOS, Sabine LAAHA, Steven GILLIS, Ursula STEPHANY, Ayhan AKSU-KOÇ, Anastasia CHRISTOFIDOU, Natalia GAGARINA, Gordana HRZICA, F Nihan KETREZ, Marianne KILANI-SCHOCH, *et al.* (2011), On the role of morphological richness in the early development of noun and verb inflection, *First Language*, 31(4):461–479.

Charles YANG (2017), Rage against the machine: Evaluation metrics in the 21st century, *Language Acquisition*, 24(2):100–125.

A dependency-based approach to word contextualization using compositional distributional semantics

Pablo Gamallo
Centro de Investigación en Tecnoloxías Intelixentes (CiTIUS)
University of Santiago de Compostela, Galiza
pablo.gamallo@usc.es

ABSTRACT

We propose a strategy to build the distributional meaning of sentences mainly based on two types of semantic objects: context vectors associated with content words and compositional operations driven by syntactic dependencies. The compositional operations of a syntactic dependency make use of two input vectors to build two new vectors representing the contextualized sense of the two related words. Given a sentence, the iterative application of dependencies results in as many contextualized vectors as content words the sentence contains. At the end of the contextualization process, we do not obtain a single compositional vector representing the semantic denotation of the whole sentence (or of the root word), but one contextualized vector for each constituent word of the sentence. Our method avoids the troublesome high-order tensor representations of approaches relying on category theory, by defining all words as first-order tensors (i.e. standard vectors). Some corpus-based experiments are performed to both evaluate the quality of the contextualized vectors built with our strategy, and to compare them to other approaches on distributional compositional semantics. The experiments show that our dependency-based method performs as (or even better than) the state-of-the-art.

Keywords: distributional semantics, compositionality, dependency-based parsing

1 INTRODUCTION

Semantic compositionality is the crucial property of natural language according to which the meaning of a complex expression is a function of the meaning of its constituent parts and of the mode of their combination (Montague 1970). In the last decade, different distributional semantic models endowed with a compositional component have been proposed. The basic approach to composition (Mitchell and Lapata 2008, 2009, 2010) is to combine vectors of two syntactically related words with arithmetic operations: addition or component-wise multiplication. However, this approach is not fully compositional because the mode of combining the constituent parts is not considered. This way, two sentences with the same constituents but with different functions, e.g. *cats chase mice* and *mice chase cats*, are wrongly interepreted with the same flat vector combination.

To take into account the mode of combination, more recent distributional approaches (Coecke *et al.* 2010) follow a strategy aligned with the formal semantics perspective. Using the abstract mathematical framework of category theory, they provide the distributional models of meaning with the elegant mechanism expressed by the principle of compositionality, where words interact with each other according to their type-logical identities (Kartsaklis 2014; Baroni *et al.* 2014). The categorial-based approaches define arguments as vectors while functions taking arguments (e.g., verbs or adjectives that combine with nouns) are n-order tensors, with the number of arguments determining their order. Function application is the general composition operation. This is formalized as the tensor contraction which is nothing more than a generalization of matrix multiplication in higher dimensions.

Even if the type-logical compostional approach based on category theory is a very elegant proposal, it has, at least, four important drawbacks:

1. It results in an information scalability problem, since tensor representations grow exponentially (Kartsaklis *et al.* 2014). For instance, if noun meanings are encoded in vectors of 500 dimensions, adjectives, which are 2-order tensors, become matrices of 500^2 cells, while transitive verbs are described as tensors

with 500^3 dimensions. This situation leads to data sparseness problems, particularly for less common adjectives and verbs.

2. The use of tensor product for function application does not always perform as well as basic composition operations on vectors, such as component-wise multiplication (Mitchell and Lapata 2010).
3. The same word that occurs in different syntactic contexts is assigned different semantic types with incomparable representations (Paperno *et al.* 2014). For example, verbs like *eat* can be used in transitive or intransitive constructions (*children eat meat*/*children eat*), or in passive (*meat is eaten*). The different uses of the verb differ in the predicate arity and, then, are encoded in tensors of different orders. Since each of these tensors must be learned from examples individually, their obvious relation is missed. For each word, the creation of as many lexical entries as the number of its different syntactic uses is a drawback shared by all grammars based on the category theory.
4. The meaning of a sentence is a single representation and there is no access to the meaning of the constituents within the context of the whole sentence. For instance, let us observe the sense of the pronoun *They* in the sequence of sentences: *children eat meat. They are fat*. By co-reference, this pronoun is linked to *children* whose sense is contextualized by the fact that they are eaters of meat. However, there is no trivial mechanism to infer this specific sense of *children* from the meaning of the whole sentence.

Some approaches have tried to solve the issues described in the aforementioned four points. However, no strategy has been designed to deal with all of them together. For instance, the first issue has been addressed by the work reported in Paperno *et al.* (2014), where the representation size grows linearly, not exponentially, for higher semantic types, allowing for simpler and more efficient parameter estimation, storage, and computation. The third issue is at the center of the work described in Weir *et al.* (2016), where the meaning of a sentence is represented by the contextualized sense of its constituent words. The final point is addressed by Kruszewski and Baroni (2014), where the authors have observed that simpler and more economical models based on multiplication or addition yield better results than more complex ones.

These drawbacks have already been addressed by Socher *et al.* (2012) who proposed a strategy based on recursive neural networks, and by Paperno *et al.* (2014) whose proposal, *practical lexical function model*, represents each function word by a vector plus an ordered set of matrices enconding its arguments. We also address the four drawbacks by proposing a dependency-based framework with transparent vectors (and not embeddings as in Socher *et al.* (2012)). Moreover, the compositional model is different from that reported in Paperno *et al.* (2014), since we define all content words as unary-tensors (standard vectors), while syntactic dependencies are binary functions combining vectors in an iterative and incremental way. Take again the sentence *"children eat meat"*. The *subject* dependency builds two contextualized senses: the sense of *children* as nominal subject of *eat* and the sense of *eat* given *children* as subject. The two contextualized senses are vectors that can be involved in further dependencies. Then, the *direct object* dependency combines the previously contextualized sense of *eat* with the noun *meat* to build two new contextualized senses: a new contextualization of the sense of verb, on the one hand, and the sense of *meat* in the context of *"children eat"*, on the other. The intrepretation of the sentence is formalized as an incremental iteration giving rise to three contextualized senses. So, in this model, the meaning of a sentence is no more a single meaning, but one (contextualized) sense per content word, and each sense is represented by means of a word vector. In the previous example, dependencies have been applied iteratively from left-to-right: first the subject, and then the direct object. But they may also be applied from right-to-left: first the direct object and then the subject. The right-to-left iteration would result in slightly different contextualized senses. This way, the sense of *children* would be more specific since it would be built in the context of *"eat meat"*.

In our approach, syntactic dependencies are compositional functions that combine vectors to build the contextualized senses of words (still vectors) in an incremental way. While words are semantically represented as vectors, dependencies are compositional operations on them. It means that we operate with only two types of semantic objects: first-order tensors (or standard vectors) for content words, and binary functions for syntactic dependencies. This solves the scalability problem of high-order tensors (first drawback). In addition, it also prevents us from giving different categorical representations to verbs

in different syntactic contexts. A verb is represented as a single vector which is contextualized as it is combined with its arguments (second drawback).

Concerning the compositional function, dependencies are operations that combine first-order vectors using simple arithmetic operations such as addition and multiplication, instead of more complex tensor products (third drawback). However, given that our vector space is enriched with syntactic information, the vectors built by composition cannot be a simple mixture of the input vectors as in the bag-of-words approaches (Mitchell and Lapata 2008). Our syntax-based vector representation of two related words encodes incompatible information and there is no direct way of combining the information encoded in their respective vectors. Vectors of content words (nouns, verbs, adjectives, and adverbs) are in different and incompatible spaces because they are constituted by different types of syntactic contexts. So, they cannot be merged. To combine them, on the basis of previous work (Thater *et al.* 2010; Erk and Padó 2008), we distinguish between direct denotation and selectional preferences (or indirect denotation) within a dependency relation.

The iterative application of the syntactic dependencies found in a sentence is actually the process of building the contextualized sense of all the content words constituting that sentence. So, the whole sentence is not assigned only one meaning – which could be the contextualized sense of the *root* word – but one sense per word, with the meaning of the root being only one such contentualized sense among many. This allows us to retrieve the contextualized sense of all constituent words within a sentence. The contextualized sense of any word might be required in further semantic processes, namely for dealing with coreference resolution involving anaphoric pronouns (fourth drawback).

The main contribution of our work is to propose a semantic space for Dependency Grammar, whose syntactic framework only consists of lexical units and dependencies (Kahane 2003; Hudson 2003). Our semantic model is wholly composed of binary operations (dependencies) and first-order vectors (words and selectional preferences). There is no room for semantic objects associated with composite expressions such as phrases or sentences. A sentence is interpreted as an iterative combination of word vectors with selectional preferences by using component-wise multiplication. This iterative and incremental com-

positional process may have two directions: from left-to-right and from right-to-left. These two directions result in slightly different contextualized words as we will show later in the experiments. Another important contribution of our work is that it should be seen as a continuation of Erk and Padó (2009) by allowing contextualized selectional preferences. Our approach was previously applied to other tasks: compositional translation (Gamallo and Pereira-Fariña 2017) and relational-based semantics (Gamallo 2017b). The current article is an extension of a previous conference work (Gamallo 2017c).

This article is organized as follows. In Section 2, our dependency-based compositional model is described. In Section 3, corpus-based experiments are performed to build and evaluate the quality of compositional/contextualized vectors. Then, in Section 4, several distributional compositional approaches are introduced and discussed. Finally, relevant conclusions are addressed in Section 5.

2 THE COMPOSITIONAL MODEL

We first give a quick overview of our vector space (Section 2.1), which is followed by a technical description of the compositional operations driven by syntactic dependencies (Section 2.2). We conclude by applying an incremental interpretation approach to our model (Section 2.3).

2.1 *Dependency-based vector representation*

Distributional Semantics associates the meaning of a word with the set of contexts in which it occurs (Firth 1957). Typically, in computational approaches, the distributional representation for a word is computed from the occurrences of that word in a given corpus (Grefenstette 1995). In distributional semantics models, each word is defined as a context vector, and each position in the vector represents a specific context of the word whose value is the frequency (or some statistical weight) of the word in that context. According to recent research, a vector space can be considered as a semantic model, since vector-based representations (i.e. distributional features) may be defined as extensions of logical expressions if they are seen as *ideal distributions* (Copestake and Herbelot 2012; Erk 2013).

Our model employs vector representations for words (or lemmas) based on syntactic contexts. Syntactic contexts are derived from bi-

nary dependencies, which can be found in a corpus analyzed with a dependency-based parser. Let's suppose the composite expression *a horse is running* was found in a corpus and is analyzed as the following syntactic dependency:

$$(nsubj, run, horse)$$

It states that the noun *horse* (dependent word) is related to the head verb *run* by means of the relation *nsubj* (nominal subject). A dependency is then a triple consisting of a relation, a head, and a dependent word. From this dependency, we can identify two complementary word contexts:

$$<nsubj_{\uparrow}, run>, <nsubj_{\downarrow}, horse>$$

Then, we count co-occurrences between words and contexts. In this case, the context $<nsubj_{\uparrow}, run>$ is assigned frequency 1 within the vector of *horse*, while we add a new occurrence to $<nsubj_{\downarrow}, horse>$ within the vector of *run*. The up arrow in $nsubj_{\uparrow}$ means that the head word *run* in the subject relation is expecting a dependent word, while the down arrow in $nsubj_{\downarrow}$ means that the dependent noun *horse* is searching for the head verb. This representation is inspired by Gamallo *et al.* (2005) and is similar to that used for distinguishing traditional selectional preferences from *inverse* selectional preferences (Erk and Padó 2008). To reduce the number of contexts, we apply a technique to filter out contexts by relevance. The filtering strategy to select the most relevant contexts consists in selecting, for each word, the R (relevant) contexts with highest log-likelihood measure. The top R contexts are considered to be the most *relevant* and informative for each word. R is a global, arbitrarily defined constant whose usual values range from 10 to 1000 (Biemann and Riedl 2013; Padró *et al.* 2014). In short, we keep at most the R most relevant contexts for each target word (where $R = 500$ in our experiments). This is an explicit and transparent representation giving rise to a non-zero matrix.

2.2 *Vector composition*

In our approach, composition is modeled by two semantic functions, *head* and *dependent*, that take three arguments each:

(1) $$head_{\uparrow}(r, \vec{x}, \vec{y}^{\circ})$$

(2) $$dep_{\downarrow}(r, \vec{x}^{\circ}, \vec{y})$$

where $head_{\uparrow}$ and $dep_{\downarrow}$ represent the head and dependent functions, respectively, r is the name of the relation (*nsubj*, *dobj*, *nmod*, etc.), and $\vec{x}$, $\vec{x}^{\circ}$, $\vec{y}$, and $\vec{y}^{\circ}$ stand for vector variables. On the one hand, $\vec{x}$ and $\vec{y}$ represent the denotation of the head and dependent words, respectively. They represent standard context distributions which we call *direct vectors*. On the other hand, $\vec{x}^{\circ}$ represents the selectional preferences imposed by the head, while $\vec{y}^{\circ}$ stands for the selectional preferences imposed by the dependent word. Selectional preferences are also called *indirect vectors* and the way we build them is described below.

Consider now a specific dependency relation, nominal subject (*nsubj*), and two specific words: *horse* and *run*. The application of the two functions consists of multiplying the direct and indirect vectors by taking into account the *nsubj* relation:

(3) $head_{\uparrow}(nsubj, \vec{run}, \vec{horse}^{\circ}) = \vec{run} \odot \vec{horse}^{\circ} = \vec{run}_{nsubj\uparrow}$

(4) $dep_{\downarrow}(nsubj, \vec{run}^{\circ}, \vec{horse}) = \vec{horse} \odot \vec{run}^{\circ} = \vec{horse}_{nsubj\downarrow}$

Each multiplicative operation results in a compositional vector which represents the contextualized sense of one of the two words (either the head or the dependent). Component-wise multiplication has an intersective effect: the selectional preferences restricts the direct vector by assigning frequency 0 to those contexts that are not shared by both vectors. Here, $\vec{horse}^{\circ}$ and $\vec{run}^{\circ}$ are indirect vectors resulting from the following vector additions:

(5) $$\vec{horse}^{\circ} = \sum_{\vec{w} \in \boldsymbol{H}} \vec{w}$$

(6) $$\vec{run}^{\circ} = \sum_{\vec{w} \in \boldsymbol{R}} \vec{w}$$

where $\boldsymbol{H}$ is the vector set of those verbs having *horse* as subject (except the verb *run*). More precisely, given the linguistic context $<nsubj_{\downarrow}, horse>$, the indirect vector $\vec{horse}^{\circ}$ is obtained by adding the vectors $\{\vec{w} | \vec{w} \in \boldsymbol{H}\}$ of those verbs (*eat*, *jump*, etc.) that are combined with the noun *horse* in that syntactic context. Component-wise addition of vectors has an union effect. In more intuitive terms, $\vec{horse}^{\circ}$ stands for the inverse selectional preferences imposed by *horse* on any verb at the subject position. As this new vector consists of verbal contexts, it lives in the same vector space as verbs and, therefore, it can be combined with the direct vector of *run*.

	$\vec{red}$	$\vec{white}$	$\vec{vague}$	$\vec{car}^{\circ}$	$\vec{red} \odot \vec{car}^{\circ}$
$<amod_{\uparrow}, car>$	5	2	0	2	10
$<amod_{\uparrow}, pencil>$	2	0	0	0	0
$<amod_{\uparrow}, idea>$	1	0	7	0	0
$<amod_{\uparrow}, book>$	2	1	2	1	2

Table 1: Deriving the vector of *red* in *red car* by dependency-based compositionality (dependent function)

	$\vec{run}$	$\vec{eat}$	$\vec{sleep}$	$\vec{horse}^{\circ}$	$\vec{horse}^{\circ} \odot \vec{run}$
$<nsubj_{\downarrow}, horse>$	3	5	1	6	18
$<dobj_{\downarrow}, program>$	5	0	0	0	0
$<prep\_in_{\downarrow}, prairie>$	2	1	1	2	4
$<prep\_with_{\downarrow}, gas>$	3	0	0	0	0

Table 2: Deriving the vector of *run* in *horses run* by dependency-based compositionality (head function)

On the other hand, $\boldsymbol{R}$ in Equation 6 represents the vector set of nouns occurring as subjects of *run* (except the noun *horse*). Given the lexico-syntactic context $<nsubj_{\uparrow}, run>$, the vector $\vec{run}^{\circ}$ is obtained by adding the vectors $\{\vec{w} | \vec{w} \in \boldsymbol{R}\}$ of those nouns (e.g. *dog*, *car*, *computer*, etc.) that might be at the subject position of the verb *run*. Indirect vector $\vec{run}^{\circ}$ stands for the selectional preferences imposed by the verb on any noun at the subject position. It is constituted by nominal contexts and, therefore, is compatible with the direct vector of *horse*.

Tables 1 and 2 are toy examples showing how to construct the compositional vectors of two contextualized words: *red* in *red car* (Table 1) and *run* in *horses run* (Table 2). Vectors are in columns and rows are dependency-based contexts. Each vector position is filled with the frequency of the word in the corresponding context. In the two tables, we represent three direct vectors, one indirect vector (derived from the direct vectors) and the compositional vector (last column). In this toy example, words are hypothetical four-dimensional vectors; whereas in real scenarios extracted from large corpora, vectors may have hundreds of thousands of dimensions.

In Table 1, the indirect vector $\vec{car}^{\circ}$, associated to the noun *car* given *red* as modifier, is obtained by adding the vectors of those adjectives that are also modifiers of *car* (except *red*). In this toy example, only the direct vector of *white* fulfills such conditions. In Table 2, the indirect vector $\vec{horse}^{\circ}$ is the result of adding the direct vectors of *eat* and *sleep*, since *horse* also occurs as subject of these verbs.

It is worth noticing that the contextualized vector of *red* within *red car* (last column in Table 1) has fewer contexts with positive values than the direct vector of the polysemous adjective *red* (out of context). The (inverse) selectional preferences imposed by *car* are able to select a more compact and less ambiguous vector of the adjective. This way, the context activating the ideological sense of *red* ($<amod_{\uparrow}, idea>$) is filtered out as it is multiplied by 0. Similarly, the resulting vector of *run* within *horses run* has fewer positive contexts and then tends to be less ambiguous than the direct vector of the polysemous verb *run* out of context. In Table 2, the contexts ($<prep\_with_{\uparrow}, gas>$, $<dobj_{\uparrow}, program>$), which hardly appears with words denoting animals, are removed (frequency 0) from the new contextualized vector of *run*. So, the inverse selectional preferences imposed by *horse* activate one specific sense of the verb: physical movement. Notice that we do not consider prepositions as content words, but as syntactic dependencies.

In approaches to computational semantics inspired by Combinatory Categorial Grammar (Steedman 1996) and Montagovian semantics (Montague 1970), the interpretation process activated by composite expressions such as *dogs chase cats*, *horses run* or *red car* relies on rigid function-argument structures. Relational expressions like verbs and adjectives are used as predicates while nouns and nominals are their arguments. In the composition process, each word is supposed to play a rigid and fixed role: the relational word is semantically represented as a selective function imposing constraints on the denotations of the words it combines with, while non-relational words are in turn seen as arguments filling the constraints imposed by the function. For instance, *run* and *red* denote functions while *horses* and *car* are their respective arguments.

By contrast, we do not define verbs and adjectives as functional artifacts driving the compositional process. In our compositional approach, dependencies are the active functions that control and rule the selectional requirements imposed by the two related words. Dependencies, instead of relational words, are then conceived of as the main functional operations taking part in composition. This way, two syntactically dependent expressions are no longer interpreted as a rigid "predicate-argument" structure, where the predicate is the active function imposing the semantic preferences on a passive argument, which

matches such preferences. On the contrary, each constituent word imposes its selectional preferences on the other. This is in accordance with non-standard linguistic research which assumes that the words involved in a composite expression impose semantic restrictions on each other (Pustejovsky 1995; Gamallo 2008; Gamallo *et al.* 2005). Not only verbs or adjectives are taken as predicates selecting different types of nouns, but so too do nouns select for different types of verbs and adjectives. Following this idea, we propose a co-compositional approach: in the head function, the dependent element imposes its restrictions on the head denotation, and the output is a more specific and less ambiguous denotation of the head. By contrast, in the dependent function, it is the head that imposes its selectional restrictions on the dependent denotation to produce a more elaborate and less ambiguous denotation of the dependent expression.

It means that the semantic space consists of just two types of entities: word vectors and dependency-based functions. Vectors represent both word senses (direct vectors) and selectional preferences (indirect vectors), while head/dependent functions represent compositional operations. A dependency-based function takes as arguments a relation and a pair of vectors (direct + indirect), and returns a more elaborate direct vector.

2.3 *Dependencies and incremental interpretation*

Frameworks such as Discourse Representation Theory (Kamp and Reyle 1993) and Situation Semantics (Barwise 1987) make two basic assumptions about interpretation: that the meaning of a sentence is dependent of the meaning of the previous sentence in the discourse; and that a sentence modifies in turn the meaning of the following sentence. Sentence meaning is not isolated from discursive unfolding; rather, meaning is incrementally constructed at the same time as discourse information is processed.

We assume that incrementality is true not only at the inter-sentence level but also at the inter-word level, i.e., between dependent words. In order for a sentence-level interpretation to be attained, dependencies must be established between individual constituents as soon as possible. This claim is assumed by a great variety of research (Kempson *et al.* 2001, 1997; Milward 1992; Costa *et al.* 2001; Schlesewsky and Bornkessel 2004). The incremental hypothesis states that

information is built up on a left-to-right word-by-word basis in the interpretation process (Kempson *et al.* 2001). The meaning of an utterance is progressively built up as the words come in. The sense of a word is provided as part of the context for processing each subsequent word. Incremental processing assumes that humans interpret language without reaching the end of the input sentence; that is, they are able to assign a sense to the initial left fragment of an utterance. This hypothesis has received a large experimental support in the psycholinguistic community over the years (McRae *et al.* 1997; Tanenhaus and Carlson 1989; Truswell *et al.* 1994).

For instance, to interpret *the cat chased a mouse*, it is required to interpret *cat chased* as a fragment that restricts the type of nouns that can appear at the direct object position: *mouse*, *rat*, *bird*, etc.[1] In the same way *police chases* restricts the entities that can be chased by police officers: *thieves*, *robbers*, and so on. However, a left-to-right interpretation process cannot be easily assumed by a standard compositional approach. In a Montagovian model, *chase* is a transitive verb denoting the binary function $\lambda x \lambda y\, chase(x, y)$, *chased a mouse* is an intransitive verb denoting a unary predicate, while *the cat chased a mouse* is a sentence denoting a truth value. The standard compositional model does not provide any interpretation for *the cat chased* within the sentence *the cat chased a mouse*; consequently, it is unable to predict how the expression *the cat chased* restricts the type of nouns appearing at the direct object position.

By contrast, in our left-to-right incremental compositional strategy, *the cat chased* is a grammatical expression referring to two semantic objects: the compositional vectors of the two related lexical units.

In our approach, the iterative application of the syntactic dependencies found in a sentence is actually the recursive process of building the contextualized sense of all the content words which constitute the sentence. Thus, the whole sentence is not assigned only one meaning (which could be the contextualized sense of the *root* word), but one

[1] We do not consider the compositional meaning of determiners, auxiliary verbs, or tense affixes. Quantificational issues associated with them are beyond the scope of this work. An interesting work on determiners in compositional distributional semantics is reported by Bernardi *et al.* (2013).

sense per lemma, where the sense of the root is only one such sense considered.

This recursive and incremental process may have two directions: from left-to-right and from right-to-left.

The incremental left-to-right interpretation of *the cat chased a mouse* is illustrated in Equation 7 (without considering the meaning of determiners nor verbal tense):

$$
\begin{aligned}
& head_{\uparrow}(nsubj, \vec{chase}, \vec{cat}^{\circ}) \quad = \quad \vec{chase}_{nsubj\uparrow} \\
& dep_{\downarrow}(nsubj, \vec{chase}^{\circ}, \vec{cat}) \quad = \quad \vec{cat}_{nsubj\downarrow} \\
& head_{\uparrow}(dobj, \vec{chase}_{nsubj\uparrow}, \vec{mouse}^{\circ}) \quad = \quad \vec{chase}_{nsubj\uparrow+dobj\uparrow} \\
& dep_{\downarrow}(dobj, \vec{chase}^{\circ}_{nsubj\downarrow}, \vec{mouse}) \quad = \quad \vec{mouse}_{nsubj\downarrow+dobj\downarrow}
\end{aligned} \tag{7}
$$

First, the head and dependent functions are applied on the subject dependency *nsubj* to build the compositional vectors $\vec{chase}_{nsubj\uparrow}$ and $\vec{cat}_{nsubj\downarrow}$. Then, the head function is applied *dobj* to produce a more elaborate chasing event, $\vec{chase}_{nsubj\uparrow+dobj\uparrow}$, which stands for the full contextualized sense of the root verb. In addition, the dependent function takes *dobj* to yield a new nominal vector, $\vec{mouse}_{nsubj\downarrow+dobj\downarrow}$, whose internal information only can refer to a specific animal: *mouse chased by the cat*. In the context of a chasing event, *mouse* does not refer to a computer's device.

The contextualized selectional preferences, $\vec{chase}^{\circ}_{nsubj\downarrow}$, represent an indirect vector obtained as follows:

$$
\vec{chase}^{\circ}_{nsubj\downarrow} \quad = \quad \vec{cat}_{nsubj\downarrow} \odot \sum_{\vec{w} \in \boldsymbol{C}} \vec{w} \tag{8}
$$

where $\boldsymbol{C}$ is the vector set of those nouns that are in the direct object role of *chase* (except the noun *mouse*). The new vector resulting by adding the vectors of $\boldsymbol{C}$ is combined by multiplication (intersection) with the contextualized dependent vector, $\vec{cat}_{nsubj\downarrow}$, to build the contextualized selectional preferences. In more intuitive terms, the selectional preferences built in Equation 8 are constituted by selecting the contexts of the nouns appearing as direct object of *chase*, which are also part of *cat* after having been contextualized by the verb at the subject position.

The dependency-by-dependency functional application results in three contextualized word senses: $\vec{cat}_{nsubj\downarrow}$, $\vec{chase}_{nsubj\uparrow+dobj\uparrow}$ and

$\vec{mouse}_{nsubj\downarrow+dobj\downarrow}$. They all together represent the meaning of the sentence in the left-to-right direction. Notice that $\vec{cat}_{nsubj\downarrow}$ is not a fully contextualized vector: it was only contextualized by the verb, but not by the direct object noun. In order to fully contextualize the subject, we need to initialize the composition process in the other way around.

In the opposite direction, from right-to-left, the incremental process starts with the direct object dependency:

$$
\begin{aligned}
& head_{\uparrow}(dobj, \vec{chase}, \vec{mouse}^{\circ}) \quad = \quad \vec{chase}_{dobj\uparrow} \\
& dep_{\downarrow}(dobj, \vec{chase}^{\circ}, \vec{mouse}) \quad = \quad \vec{mouse}_{dobj\downarrow} \\
& head_{\uparrow}(nsubj, \vec{chase}_{dobj\uparrow}, \vec{cat}^{\circ}) \quad = \quad \vec{chase}_{dobj\uparrow+nsubj\uparrow} \\
& dep_{\downarrow}(nsubj, \vec{chase}^{\circ}_{dobj\downarrow}, \vec{cat}) \quad = \quad \vec{cat}_{dobj\downarrow+nsubj\downarrow}
\end{aligned}
\tag{9}
$$

In Equation 9, the verb *chase* is first restricted by *mouse* at the direct object position, and then by its subject *cat*. In addition, this noun is restricted by the vector $\vec{chase}^{\circ}_{dobj\downarrow}$, which represents the contextualized selectional preferences built by combining $\vec{mouse}_{dobj\downarrow}$ with the vectors of the nouns that are in the subject position of *chase* (except *cat*). This new compositional vector represents a very contextualized nominal concept: *the cat that chased a mouse*. The word *cat* and its specific sense can be related to anaphorical expressions by making use of co-referential relationships at the discourse level: e.g., pronoun *it*, other definite expressions (*that cat*, *the cat*, ...), and so on. Notice that this compositional vector might also be used to represent the contextualized sense of a nominal restricted by a relative clause. For this type of construction, it is worth paying special attention to the work reported in Sadrzadeh *et al.* (2013), where the authors describe a tensor-based method to represent the compositional meaning of relative pronouns.

The meaning of a sentence is ideally represented by the full contextualization of its constituent words. Yet, as has been said, not all words in a sentence can be fully contextualized using left-to-right combination. For instance, to fully contextualize the noun subject $\vec{cat}_{dobj\downarrow+nsubj\downarrow}$ within the subject-verb-object sentence *the cat chased a mouse*, the iterative process must follow the right-to-left direction: first, the noun vector $\vec{mouse}$ is combined with chasing preferences on the object ($\vec{chase}^{\circ}$). Then, the resulting vector of the previous combination is used to generate the restricted verb preferences on the subject ($\vec{chase}^{\circ}_{dobj\downarrow}$), which are combined with the noun vector $\vec{cat}$ to

eventually return the fully contextualized vector of the subject noun. As in standard compositional approaches, vectors are combined with pointwise multiplication. The main difference with regard to standard vector combination is that our compositional strategy also relies on vectors representing selection preferences. Both selection preferences and compositional (contextualized) vectors are generated dynamically during word combination.

The order of function application is flexible since it is not constrained by the type of dependencies or by the arity of function words (mainly verbs). A particular order may be applied by principles or constraints that are independent of the syntactic structure. The constraints that specify a particular order may be defined by external factors. For instance, if the objective is to simulate a psycholinguistic notion of incrementality, where the meaning of words is gradually elaborated as they are syntactically integrated into new dependencies, then the best option is to implement the left-to-right algorithm. However, nothing prevent us implementing the complementary right-to-left direction in order to compare the contextualized senses generated by using both directions (as we will show later in the experimental section). Instead of applying all possible orders, which has high computational cost, it would be possible to apply external constraints and principles to impose a very restricted order. One of these contraints might be, for instance, to consider the degree of ambiguity of lexical units: we could apply first those dependencies containing less ambiguous words with more semantically homogenous vectors; and then use these in a subsequent step to disambiguate more heteregenous word vectors (i.e., more ambiguous ones) (Gamallo 2008).

In the sentence *the coach drives the team*, this contraint should lead us to interpret *drives the team* before *the coach drives*, since *team* is less ambiguous than *coach*. By contrast, in *the team hired a coach*, the order should be the other way around following the same principle. In a more complex sentence such as *I lost my computer mouse*, the same principle would force us to interpret first the less ambiguous noun-noun dependency between *computer* and *mouse* before the more ambiguous relation between *lost* and *mouse*. This ambiguity-based constraint may be seen as a general procedure to word sense discrimination. Yet, the definition and implementation of specific contraints and principles restricting function application is beyond the scope of the current work.

Finally, it is worth noticing that the compositional objects we build using dependencies are not flat representations such as those derived from typical dependency-based analysis. The order of functional application is meaningful and allows us to build vectors at different constituency levels in terms of immediate constituent analysis. A criticism of dependency analysis is that it is not able to deal with the different interpretations obtained from expressions like *fastest American runner* and *American fastest runner*. As both expressions are analyzed with the same flat dependency-based structure (*fastest* and *American* are dependent of *runner*), it would not be possible to derive different semantic entailments from the same syntactic analysis. However, in our dependency-based model, the order in which the functions are applied allows us to build several compositional entities, which simulate the construction of different constituent units.

3 EXPERIMENTS

We have designed and developed a system, *DepFunc*, based on the method described in the previous section. Although the method can potentially be applied to any sentence, regardless of its syntactic structure, the limitations of the implementation and the complexity of the task have led us to apply it only to expressions with a predetermined and fixed structure: adjective-noun, noun-verb, and noun-verb-noun.

Two different types of experiments were carried out to evaluate the performance of our system. The specific objective of the first experiment (Section 3.1) is to compare the distributional similarity between compositional vectors of composite expressions and corpus-observed vectors of the same composite expressions. If they are similar, our vectors predicted by compositionality can be considered correct because they are close to standard vectors built with observed data. We compared our strategy with the one defined in Baroni and Zamparelli (2010), which also carried out a similar evaluation. Experiments were made with ADJ-NOUN (to abbreviate: AN) and NOUN-VERB expressions (to abbreviate: NV).

In the second type of experiments (Section 3.2), we use test datasets to measure the correlation between human similarity judgements and similarity coefficients computed with our compositional expressions. In Subsection 3.2.2, we measure the quality of composi-

tional vectors built from NV composite expressions, using as gold standard the test dataset provided by Mitchell and Lapata and described in (Mitchell and Lapata 2008). In Subsection 3.2.3, we check the quality of more complex composite expressions, namely NOUN-VERB-NOUN constructions (NVN) incrementally composed with *nsubj* and *dobj* dependencies.

3.1 *Compositional and corpus-observed vectors*

As in Baroni and Zamparelli (2010), the experiment consists in comparing the distributional similarity between two different types of vectors associated with composite expressions: *compositional vectors* and *corpus-observed vectors*. Compositional vectors are those built following the compositional method described in the previous section. They are thus model-generated vectors constructed according to the corpus-based observed frequencies of their constituents. Corpus-observed vectors of composite expressions are constructed with the frequencies associated with the whole expression. They are called *holistic* vectors by Turney (2013). We should expect that the compositional and the holistic vectors built for the same composite expression should be similar (Baroni and Zamparelli 2010). More precisely, we expect that the predicted distribution computed by our compositional approach should yield similar vectors to those built with real distributions calculated from real-world corpora. For instance, if we build a compositional vector for *red car* according to the frequency of its parts in a compositional way, the resulting vector should be similar to the vector constructed by just observing the co-occurrences of the composite expression as a whole. Notice that there are exceptions to that, namely those cases where the meaning of the compound expression is not compositional (e.g., collocations, frozen expressions, idioms, and so on).

3.1.1 Corpus and distributional models

In order to build the compositional and holistic (corpus-observed) vectors, we made two partitions from the English Wikipedia (dump file of November 2015), with 100M tokens each. The first partition was used to build the compositional vectors (and to train learning models) while the second partition was used for extracting corpus-observed vectors as well as for testing and evaluation. Word vec-

tors were built by computing their co-occurrences in lexico-syntactic contexts. We used the dependency parser DepPattern (Gamallo and Garcia 2018; Gamallo 2015) to perform syntactic analysis. Three different types of vectors were built from the corpus: nominal, verbal, and adjectival vectors. Then, for each word we filtered out irrelevant contexts using simple count-based techniques inspired by those described in Gamallo (2017a), where matrices are stored in hash tables with only non-zero values. More precisely, the association between words and their contexts were weighted with the Dunning's likelihood ratio (Dunning 1993) and then, for each word, only the N contexts with highest likelihood scores were stored in the hash table (where $N = 500$). So, the remaining contexts were removed from the hash (whereas in standard vector/matrix representations, instead of removing contexts we should assign them zero values). This filtering-based approach turned out to be more efficient than other strategies based on dimensionality reduction such as Singular Value Decomposition (Gamallo and Bordag 2011). In addition, our approach requires explicit vector spaces, which are more linguistically transparent than dense representations such as neural-based word embeddings.

Not all words were selected for computing similarity; in particular, we selected those nouns, verbs, and adjectives occurring in more than 100 different contexts. The experiments were made with lemmas.

Experiments were performed with two types of composite units: AN expressions in the nominal space and NV in the verbal space. Our specific task consists of selecting a list of both AN and NV composites, building their compositional and corpus-observed vectors, and checking whether each particular compositional vector is similar to its corresponding corpus-observed vector. To avoid possible bias between predicted and observed occurrences, corpus-observed vectors were derived from the second partition of the corpus, while compositional vectors were built from the first partition. To build compositional vectors, the strategy defined in the previous section was implemented in Perl giving rise to the software *DepFunc*.

3.1.2 Evaluation

The list of target composites for evaluation was created as follows. In the second partition with 100M tokens, we selected the composites

with more than 50 different contexts: 6676 ANs and 3004 NVs. Then, we filtered out those composites with at least one constituent word which does not appear in the matrices created from the first corpus partition (since these constituent words had fewer than 100 lexico-syntactic contexts in the first partition). Finally, we obtained a test list of 1,841 ANs and another of 767 NVs, which were subsequently manually revised in order to filter out ill-formed expressions. We obtained more AN composites than those of NVs because the nominal space has a higher number of entities and lexico-syntactic contexts than the verbal space.

Then, we built, on the one hand, the compositional vectors of the selected 1,841 ANs and 767 NVs using the first corpus partition and, on the other hand, the corpus-observed vectors of the same composites using the quantitative information of the second partition. The new vectors are added to both the nominal and verbal matrices. In total, the nominal matrix contains 22,025 single nouns and $1{,}841 \times 2$ AN composites, while the verbal matrix consists of 5,131 single verbs and 767×2 NV composites. Next, all possible pairs were generated and cosine similarity was computed in each matrix. For each corpus-observed composite, we created a ranked list of the N most similar expressions, and finally, we verified whether its corresponding compositional composite is found in the list and recorded its ranking.

We define *hit* to mean an instance of finding the compositional vector of a composite expression in the ranked list of its corresponding corpus-observed vector. For instance, if the compositional vector of *"red car"* is in the top N list of similar candidates of the corpus-observed vector associated to the same expression, we count one *hit*.

To compare our model with a state-of-the-art system, we used the software DISSECT (Dinu *et al.* 2013a)[2] The software was used to train and apply the compositional functions described in Baroni and Zamparelli (2010), taking as input the first (part-of-speech tagged) corpus partition and the lists of test composites introduced above. The training process was performed by selecting all the adjectives and verbs of the test list and all their occurrences with those nominal arguments

[2] `http://clic.cimec.unitn.it/composes/toolkit/introduction.html`

that are not in the test list. To compute word-context co-occurrences, we defined the contexts of a word as the bag-of-lemmas extracted from a window of size 5 (two context words both to the left and to the right of the target word). Co-occurrence matrices were reduced to 300 dimensions by making use of Singular Value Decomposition. Similarity between vectors was computed with the cosine measure. The function we have used for training the model is *LexFunc* (Lexical Function), which gave rise to the best results in the experiments described in Baroni and Zamparelli (2010) and Dinu *et al.* (2013b).

The final results are shown in Tables 3 and 4. Each system is evaluated with regard to different values of K: 10, 50 and 100. For each value, we count the proportion of hits to compute precision at K, noted P@K. For instance P@10 is the number of hits within the 10 most similar expressions divided by the total number of evaluated expressions. The other measure, *ranking_average*, stands for the average of the ranking positions of all hits within the ranked list with the 100 most similar expressions. For instance, if 3 hits were found in rankings 25, 50, and 75, the *ranking_average* is 50. This evaluation is inspired by standard information retrieval metrics.

Four strategies are compared: what we call *lower-bound* is just the by chance probability of finding hits at each K level. The hits found at $K = 100$ tend to occur at position 50 on average. The *baseline* strategy consists in associating the compositional vector to the head vector. For instance, the baseline compositional vector of *"red car"* would be the vector of the head noun *car*, while the baseline compositional vector of *"horses run"* would be the vector of the head verb *run*. This is a very reliable and sound baseline because there is a straight semantic relationship between any composite expression and its head: the concept designated by the head tends to be the direct hypernym of the concept designated by the composite expression. So, *"red car"* (hyponym) must be closely related to *car* (hypernym). In the experiments described by Baroni and Zamparelli (2010), this baseline was the third best strategy out of six evaluated systems, outperforming the approaches introduced by Mitchell and Lapata (2009) and Guevara (2010). The system denoted *LexFunc* represents the best compositional system, known as *alm* and based on the *Lexfunc* model, described in

Baroni and Zamparelli (2010).[3] These two systems are compared with our compositional approach: *DepFunc (head)*. It is worth noting that, in these experiments, the evaluation is just focused on the contextualized heads of compositional vectors. The reason for this is that the syntactic contexts of holistic expressions are found in the space of the heads: AN expressions are nouns and NVs are verbs. So, in order to compare compositional with holistic expressions, we have to consider that compositional ANs are contextualized nouns and compositional NVs are contextualized verbs.

system	*P@10*	*P@50*	*P@100*	*ranking_average*
lower-bound	0%	0.2%	0.4%	50
baseline (noun)	11.74%	31.36%	42.95%	33.90
DepFunc (head)	**36.39**%	**53.01**%	**60.32**%	**17.69**
LexFunc	21.36%	35.79%	42.87%	22.43

Table 3: Percentages of hits (precision at 10, 50 and 100) and ranking average in the ranked lists of AN expressions

system	*P@10*	*P@50*	*P@100*	*ranking_average*
random	0.1%	0.7%	1.5%	50
baseline (verb)	6.21%	23.16%	35.02%	37.74
DepFunc (head)	17.51%	**37.85**%	**45.76**%	25.64
LexFunc	**24.54**%	35.24%	39.81%	**24.23**

Table 4: Percentages of hits (precision at 10, 50 and 100) and ranking average in the ranked lists of NV expressions

Tables 3 and 4 show the results of AN and NV expressions. Our compositional approach, DepFunc (head), clearly outperforms the baseline strategies for both AN and NV. In addition, it also outperforms LexFunc for AN. However, the differences between LexFunc and DepFunc are not so sharp for NV. In fact, DepFunc finds more hits within larger ranked lists (50 and 100), but those found by LexFunc are in better ranks, being even more precise at $K = 10$.

The main problem of this evaluation is that it does not allow us to take advantage of the contextualization of the dependent word. This will be solved in the following experiments.

[3] Additive and multiplicative models implemented in DISSECT were also evaluated, but the results obtained were below the baseline.

3.2 *Correlation with human judgements*

In the following experiments, we compare the similarity between pairs of compositional vectors representing composite expressions with the similarity given by annotators to those expressions. In this case, we will compare all contextualized words of the expressions instead of only considering the word heads.

3.2.1 Corpus and distributional models

In these experiments, our working corpus consists of both the English Wikipedia (dump file of November 2015[4]) and the British National Corpus (BNC)[5]. In total, the corpus contains about 2.5 billion word tokens, which were analysed with DepPattern.

Word vectors were built by computing their co-occurrences in syntactic contexts. Two different types of vectors were built from the corpus: nominal and verbal vectors. Distributional matrices were built using the same strategy as the one defined for the previous experiment.

This process of matrix reduction resulted in the selection of 330 953 nouns (most of them proper names) with 236,708 different nominal contexts; and 6,618 verbs with 140,695 different verbal contexts. As the contexts of nouns and verbs are not compatible, we created two different vector spaces. Words and their contexts were stored in two hashes, one per vector space, which represent matrices containing only non-zero values. Cosine similarity was calculated for pairs of composite expressions.

3.2.2 NV composite expressions

The test dataset by Mitchell and Lapata (2008) comprises a total of 3600 human similarity judgements. Each item consists of an intransitive verb and a subject noun, which are compared to another NV pair combining the same noun with a synonym of the verb that is chosen to be either similar or dissimilar to the verb in the context of the given subject. For instance, *child stray* is related to *child roam*, *roam* being a synonym of *stray*. The dataset was constructed by extracting NV composite expressions from the British National Corpus (BNC) and verb synonyms from WordNet (Miller *et al.* 1990). To evaluate the results

[4] `https://dumps.wikimedia.org/enwiki/`

[5] `http://www.natcorp.ox.ac.uk`

of our systems, Spearman correlation is computed between individual human similarity scores and the systems' predictions.

As the objective of the experiment is to compute the similarity between pairs of NV composite expressions, we are able to compare the similarity not only between the contextualized heads of two NV composite expressions, but also between their contextualized dependent expressions. So, we built compositional vectors using not only the head function, but also the dependent one. For instance, we compute the similarity between *eye flare* vs. *eye flame* by comparing first the verbs *flare* and *flame* when combined with *eye* in the subject position (head function), and by comparing how (dis)similar the noun *eye* is when combined with both the verbs *flare* and *flame* (dependent function). In addition, as we are provided with two similarities (*head* and *dep*) for each pair of compared expressions, it is possible to compute a new similarity measure by averaging *head* and *dep*, and what we call *head* + *dep* system.

Table 5 shows the Spearman's correlation values (ρ) obtained by our compositional strategy (*DepFunc*). We compare our results to the *Lexfunc* algorithm (Baroni and Zamparelli 2010), which is the state-of-the-art method for this dataset according to the ρ score reported in Dinu *et al.* (2013b) using a corpus consisting of approximately 2.8 billion tokens compiled from Wikipedia, BNC and ukWaC (Baroni *et al.* 2009). In the first row of *DepFunc*, we show the ρ value obtained by our combinatorial similarity measure (*head* + *dep*). The ρ score reaches 0.32, which is higher than using only head similarity (*head*) or dep similarity (*dep*). This shows that the similarity obtained by combining the head and dependent functions is more accurate than that obtained by using only one type of compositional function. The *head* + *dep* similarity strategy based on *DepFunc* outperforms the *Lexfunc* system (0.26). The baseline method we have implemented (first

system	ρ	***size of training corpus***
non-compositional (V)	0.11	2.5B tokens: Wiki & BNC
DepFunc (head + dep)	**0.32**	2.5B tokens: Wiki & BNC
DepFunc (head)	0.27	2.5B tokens: Wiki & BNC
DepFunc (dep)	0.31	2.5B tokens: Wiki & BNC
Lexfunc (Dinu et al. 2013)	0.26	2.8B tokens: Wiki, BNC & ukWaC

Table 5: Spearman's correlation for intransitive expressions using the benchmark by Mitchell and Lapata (2008)

row in Table 5) is a non-compositional strategy just based on the similarity between the head verbs within the NV pairs. In this case, all the compositional methods clearly outperform this basic strategy. Finally, the non-compositional similarity between the noun subjects has not been computed because the nouns of each NV pair are identical in the current dataset.

3.2.3 NVN composite expressions

The last experiment consists of evaluating the quality of compositional vectors built by means of the consecutive application of head and dependency functions associated with nominal subject and direct object. The experiment is performed on the dataset developed in Grefenstette and Sadrzadeh (2011a). The dataset was built using the same guidelines as Mitchell and Lapata (2008), using transitive verbs paired with subjects and direct objects: NVN composites.

Given our compositional strategy, we are able to compositionally build several vectors that somehow represent the meaning of the whole NVN composite expression. Take the expression *the coach runs the team*. If we follow the left-to-right strategy (noted *nv-n*), at the end of the compositional process, we obtain two fully contextualized senses:

nv-n_head The sense of the head *run*, as a result of being contextualized first by the preferences imposed by the subject and then by the preferences required by the direct object. We note *nv-n_head* the final sense of the head in a NVN composite expression following the left-to-right strategy.

nv-n_dep The sense of the object *team*, as a result of being contextualized by the preferences imposed by *run* previously combined with the subject *coach*. We note *nv-n_dep* the final sense of the direct object in a NVN composite expression following the left-to-right strategy.

If we follow the right-to-left strategy (noted *n-vn*), at the end of the compositional process, we obtain two fully contextualized senses:

n-vn_head The sense of the head *run* as a result of being contextualized first by the preferences imposed by the object and then by the subject.

n-vn_dep The sense of the subject *coach*, as a result of being contextualized by the preferences imposed by *run* previously combined with the object *team*.

Table 6 shows the Spearman's correlation values (ρ) obtained by all the different variations built by our system *DepFunc*. The best score was achieved by averaging the head and dependent similarity values derived from the *n-vn* (right-to-left) strategy. Let us note that, for NVN composite expressions, the left-to-right strategy seems to build less reliable compositional vectors than the right-to-left counterpart. Note too that the broader model (*n-vn* + *nv-n*) resulting from combining the two strategies does not improve the results of the best one (*n-vn*). This model, *n-vn* + *nv-n*, is computed by averaging the similarities of both *n-vn_head* + *dep* and *nv-n_head* + *dep*. More precisely, it is the result of averaging the four fully contextualized vectors:

- *nv-n_head*: left-to-right full contextualization of the verb,
- *nv-n_dep*: left-to-right full contextualization of the object noun,
- *n-vn_head*: right-to-left full contextualization of the verb,
- *n-vn_dep*: right-to-left full contextualization of the subject noun.

system	ρ
non-compositional (V)	0.27
DepFunc (nv_head)	0.33
DepFunc (nv_dep)	0.19
DepFunc (vn_head)	0.36
DepFunc (vn_dep)	0.38
DepFunc (nv-n_head + dep)	0.35
DepFunc (nv-n_head)	0.33
DepFunc (nv-n_dep)	0.20
DepFunc (n-vn_head + dep)	**0.46**
DepFunc (n-vn_head)	0.36
DepFunc (n-vn_dep)	0.42
DepFunc (n-vn + nv-n)	0.44
Grefenstette and Sadrzadeh (2011)	0.28
Hashimoto and Tsuruoka (2014)	0.43
Polajnar et al. (2015)	0.35

Table 6:
Spearman's correlation
for transitive expressions
using the benchmark
by Grefenstette and Sadrzadeh (2011)

It is worth mentioning that the best fully contextualized vector is the subject noun generated with the right-to-left algorithm (*n-vn_dep* = 0.42), which outperforms the two contextualized verb senses: *n-vn_head* and *nv-n_head*. This result was not expected since the sense of the verb represents the meaning of the syntactic root of the sentence, which is the best connected word in the syntactic tree and, by extension, the best positioned word to represent the core meaning of the sentence. However, the fact that the subject noun works so well is conceptually possible since any fully contextualized vector may represent the meaning of the whole sentence from a specific point of view.

The score value obtained by our *n-vn_head+dep* right-to-left strategy outperforms the three other systems tested using this dataset: Grefenstette and Sadrzadeh (2011b) and Polajnar *et al.* (2015), which are two works based on the categorical compositional distributional model of meaning of Coecke *et al.* (2010), and the neural network strategy described in Hashimoto and Tsuruoka (2015).

At the top of Table 6, we show the non-contextual baseline we have created for this dataset: similarity between single verbs. No test has been made for subject and object nouns since they are identical in each pair of transitive clauses, as was the case with the subject nouns in the dataset of intransitive expressions. In the current experiment, the correlation ρ of the non-compositional baseline is much higher than in Table 5. This might explain why the best correlation value of the compositional strategy is also much higher for this dataset (0.46 vs. 0.32). The table also shows four intermediate values resulting from comparing partial compositional constructions: the noun-verb (*nv_head* and *nv_dep*) and the verb-noun (*vn_head* and *vn_dep*) combinations. Two interesting remarks can be made from these values when they are compared with the full compositional constructions.

First, there is no clear improvement of performance if we compare the full compositional information of the two transitive constructions with the partial combinations. On the one hand, the full *nv-n* construction does not improve the scores obtained by the partial intransitive *nv*. On the other hand, *n-vn* performs slightly better than *vn* but only in the case of the dependent function which makes use of contextualized selectional preferences: *n-vn_dep* = 0.42 / *vn_dep* = 0.38. The low performance at the second level of composition might call into

question the use of contextualized vectors to build still more contextualized senses. The scarcity problem derived from the recursive combination of contextualized vectors is an important issue which could be resolved by incorporating more text/additional corpora, and which we should analyze with more complex evaluation tests.

The second remark is about the difference between the two algorithms: left-to-right and right-to-left. The scores achieved by the left-to-right algorithm (*nv*, *nv-n*) are clearly below those achieved by right-to-left (*vn*, *n-vn*) . This might be due to the weak semantic motivation of the selectional preferences involved in the subject dependency of transitive constructions in comparison to the direct object. In fact, right-to-left and left-to-right function application produces substantially different vectors because each algorithm corresponds to a particular hierarchy of constituents. Change of constituency implies different semantic entailments; for example, consider the different levels of constituency of noun modifiers (e.g. *fastest American runner* $\neq$ *American fastest runner*). Finally, the poor results of *nv* in this dataset compared with those obtained in Table 5 is explained because the subject role is less meaningful in transitive clauses than in intransitive ones. The subject of intransitive clauses is assigned a complex semantic role that tends to merge the notions of agent and patient. By contrast, the subject of transitive constructions tends to be just the agent of an action with an external patient.

4 RELATED WORK

Several models for compositionality in vector spaces have been proposed in the last decade, and most of them use bag-of-words as basic representations of word contexts. As has been said in the introduction, the basic approach to composition, explored by Mitchell and Lapata (2008, 2009, 2010), is to combine vectors of two syntactically related words with arithmetic operations: addition and component-wise multiplication. The additive model produces a sort of union of word contexts, whereas multiplication has an intersective effect. According to Mitchell and Lapata (2008), component-wise multiplication performs better than the additive model. However, in Mitchell and Lapata (2009, 2010), these authors explore weighted additive models giving more weight to some constituents in specific word combinations. For

instance, in a noun-subject-verb combination, the verb is assigned a higher weight because the whole construction is closer to the verb than to the noun. Other weighted additive models are described in Guevara (2010) and Zanzotto *et al.* (2010). Another work using these basic composition operations is reported in Reddy *et al.* (2011). In this work, the compositional model is enriched with a notion close to our concept of contextualization, which the authors call *dynamic prototype*, but only applied to noun-noun compounds. The model represents each constituent by a prototype vector which is built dynamically by activating only the contexts considered to be relevant with regard to the other constituent. All these models share a common trait: they define composition operations solely for pairs of words. Their main drawback is that they do not propose a more systematic model accounting for all types of semantic composition. They do not focus on the logical aspects of the functional approach underlying compositionality.

As has been said before, other distributional approaches develop sound compositional models of meaning where functional words are represented as high-dimensional tensors (Coecke *et al.* 2010; Baroni and Zamparelli 2010; Grefenstette *et al.* 2011; Krishnamurthy and Mitchell 2013; Kartsaklis and Sadrzadeh 2013; Baroni 2013; Baroni *et al.* 2014). This idea is mostly based on Combinatory Categorial Grammar and typed functional application inspired by Montagovian semantics. The functional approaches relying on Categorial Grammar distinguish the words denoting atomic types, which are represented as vectors, from those that denote compositional functions applied to vectors. By contrast, in our compositional approach, we show that function application is not associated with predicate words such as adjectives or verbs, but rather with binary dependencies. Our semantic space does not map the syntactic structure of Combinatory Categorial Grammar but that of Dependency Grammar. This way, we avoid the troublesome high-order tensor representations of verbs with n-arity arguments.

Some of the approaches cited above induce the compositional meaning of the functional words from examples adopting regression techniques commonly used in machine learning (Baroni and Zamparelli 2010; Krishnamurthy and Mitchell 2013; Baroni 2013; Baroni *et al.* 2014). In our approach, by contrast, functions associated with dependencies are just basic arithmetic operations on vectors, as in the case

of the arithmetic approaches to composition described above (Mitchell and Lapata 2008). Arithmetic approaches are easy to implement and produce high-quality compositional vectors, which makes them a good choice for practical applications (Baroni *et al.* 2014).

The other compositional approaches based on Categorial Grammar use tensor products for composition (Coecke *et al.* 2010; Grefenstette *et al.* 2011). As has been said in the introduction, at least two problems arise with tensor products. First, they result in an information scalability problem, since tensor representations grow exponentially as the phrases grow longer (Turney 2013). And second, tensor products did not perform as well as component-wise multiplication in the experiments made by Mitchell and Lapata (2010). To improve the performance of the composition process, the tensor-based approach reported in Kartsaklis *et al.* (2014) is provided with an explicit disambiguation step prior to composition. In Paperno *et al.* (2014), the authors try to partially overcome the scalability problem of tensors by representing a functional word as a vector plus an ordered set of matrices, with one matrix for each argument the function takes.

There are a few works using vector spaces structured with syntactic information which, as in our approach, are not based on n-order tensors. Thater *et al.* (2010) distinguish between *first-order* and *second-order* vectors in order to allow two syntactically incompatible vectors to be combined. Similarly, in Melamud *et al.* (2015) the second-order vectors are called "substitute vectors". The notion of second-order (or substitute) vector is close to our concept of *indirect vector*, while their first-order vector corresponds to our *direct vector*. However, there are important differences with regard to our approach. In (Thater *et al.* 2010), the combination of a first-order with a second-order vector returns a second-order vector, which can be combined with other second-order vectors. This could require the resort to third-order (or n-order) vectors at further levels of vector composition. By contrast, in our approach, any vector combination always returns a first-order (i.e. *direct*) vector, and we only permit compositional combinations between a direct vector and an indirect one. This simplifies the compositional process at any level of analysis.

The work by Thater *et al.* (2010) is inspired by that described in Erk and Padó (2008) and Erk and Padó (2009). Erk and Padó (2008) proposes a method in which the combination of two words, a and b,

returns two vectors: a vector *a'* representing the sense of *a* given the selectional preferences imposed by *b*, and a vector *b'* standing for the sense of *b* given the (inverse) selectional preferences imposed by *a*. The main problem is that this approach does not propose any compositional model for sentences. Its objective is to simulate word sense disambiguation, but not to model semantic composition at any level of analysis. In Erk and Padó (2009), the authors briefly describe an extension of their model by proposing a recursive application of the compositional function. However, they only formalize the recursive application when the composite expression consists of two dependents linked to the same head. So, they explain how the head is contextualized by its dependents, but not the other way around. They do not model the influence of context on the selectional preferences. In other terms, their recursive model does not make use of contextualized selectional preferences. By contrast, in our approach, selectional preference are contextualized recursively. This is formalized in Equation 8 (Section 2.3).

Thater *et al.* (2010) took up the basic idea from Erk and Padó (2008) which consists in exploiting selectional preference information for contextualization and disambiguation. However, they did not borrow the idea of splitting the output of a word combination into two different vectors (one per word). As far as we know, no fully and coherent compositional approach has been proposed on the basis of the interesting idea of returning two contextualized vectors per combination. Our approach is an attempt to join the main ideas of these syntax-based models (namely, selectional preferences as indirect vectors and two returning senses per word combination) into an entirely compositional model. In sum, we generalize the model introduced by Erk and Padó (2008) to include dependencies as compositional operations allowing us to interpret any composite expression with any number of word constituents. Finally, it is important to point out that there is another relevant difference between our work and that reported in Erk and Padó (2008), Thater *et al.* (2010), and Melamud *et al.* (2015). While they tested their systems on a task of determining word meaning in context by lexical substitution, to evaluate our system we performed experiments in the task of measuring phrase similarity.

A very similar work to our compositional approach has been reported in Weir *et al.* (2016). The authors also state that distributional

composition is a matter of integrating the meaning of each of the words in the phrase. The main difference is the type of context they use to build word vectors. Each word occurrence is modelled by what they call "anchored packed dependency tree", which is a dependency-based graph that captures the full sentential context of the word. The main drawback of this context approach is its critical tendency to build very sparse word representations.

Finally, recent works make use of deep learning strategies to build compositional vectors, such as recursive neural network models (Socher *et al.* 2012; Hashimoto and Tsuruoka 2015), which share with our model the idea that in the composition of two words both words modify each other's meaning. Similarly, the bidirectional recursive neural network reported in Irsoy and Cardie (2014) computes a context vector for each word. It is also worth noting the deep learning syntax-based compositional version of the C-BOW algorithm (Pham *et al.* 2015).

5 CONCLUSIONS

In this paper, we described a distributional model to contextualize word meaning in composite expressions based on a syntactically structured vector space. To avoid the different syntactic environments associated with two syntactically dependent words, we proposed to combine direct with indirect vectors, which are compatible and can be merged into a new direct vector. An indirect vector represents the selectional preferences that one word uses to contextualize the direct vector of the other word. The combination of two related words gives rise to two vectors which represent the senses of the two contextualized words. This process can be repeated until no syntactic dependency is found in the analyzed composite expression. The compositional interpretation of a composite expression builds the sense of each constituent word in a recursive and incremental way.

Syntactic dependencies are endowed with a combinatorial meaning. Characterizing dependencies as compositional devices has important consequences on the way in which the process of semantic interpretation is considered. First, dependencies are binary functions on vectors while all content words are vectors. Vectors of content words (as well as collocations and idioms) can be constructed from a cor-

pus directly, while vectors of composite expressions are the result of composition operations driven by dependencies. Second, the contextualization process is performed in an incremental way dependency by dependency. It starts with very ambiguous vectors associated with the constituent words before composition and results in more compact and less ambiguous vectors associated with the contextualized words. And third, as syntactic dependencies are conceived here as semantic operations, syntax becomes a semantic participant involved in the interpretation process (Langacker 1991).

Our compositional model tackles the problem of *information scalability*. This problem states that the size of semantic representations should grow in proportion to the amount of information that they are representing. If the size of the contextualized vectors is fixed, eventually there will be information loss. Besides, the size of vector representations should not grow exponentially. In our approach, even if the size of the contextualized vectors is fixed, there is no information loss since each word of the composite expression is associated to a compositional vector representing its context-sensitive sense. In addition, the contextualized vectors do not grow exponentially since their size is fixed by the vector space: they are all first-order tensors.

Substantial problems still remain unsolved. For instance there is no clear boundary between compositional and non-compositional expressions (collocations, compounds, or idioms). It seems to be obvious that vectors of full compositional units should be built by means of compositional operations and predictions based on their constituent vectors. It is also evident that vectors of entirely frozen expressions should be totally derived from corpus co-occurrences of the whole expressions without considering internal constituency. However, there are many expressions, in particular collocations (such as *save time*, *go mad*, *heavy rain*, etc.) which can be considered as both compositional and non-compositional. In those cases, it is not clear which is the best method to build their distributional representation: predicted vectors by compositionality; or corpus-observed vectors of the whole expression.

Another problem that has not been considered is how to represent the semantics of some grammatical words, namely determiners and auxiliary verbs (i.e., noun and verb specifiers). This might require a different functional approach, probably closer to the work described

by Baroni *et al.* (2014), which defines functions as linear transformations on vector spaces. A solution might be similarly inspired by Gupta *et al.* (2015), where the authors analyze the distributional features associated with referential expressions.

An obvious drawback of the recursive strategy is the scarcity that results from the iterative application of several contextualizations to the same word vector. The more complex the dependency structure, the fewer occurrences there will be to compute the in-context selectional preferences. This problem also underlies other similar approaches based on transparent and interpretable distributional models, such as that reported in Weir *et al.* (2016). Kober *et al.* (2016) proposed a solution to this problem. Their proposal involves explicitly inferring un-observed co-occurrences using the distributional neighborhood. More precisely, in order to transform a sparse word vector w into a new enriched vector w', the algorithm iterates over all word vectors w in a given distributional model M, and adds the vector representations of the nearest neighbors, determined by cosine similarity, to the representation of the new enriched word vector w'. In future work, we will carry out new experiments by using this strategy on similarity datasets containing phrases or sentences with more complex syntactic structures.

Among the most fundamental applications of compositional models are paraphrasing and textual entailment. For instance, by making use of sentence similarity, we should be able to infer that the sentence *A stadium craze is sweeping the country* entails *A craze is covering the nation*, but not *A craze is brushing the nation* (Garrette *et al.* 2014). These applications build compositional vectors from co-occurrences observed in monolingual corpora. However, if the same methodology is applied to acquire phrase and sentence similarity from comparable corpora, it could be possible to learn translation equivalents of composite units. This could lead to new machine translation techniques.

In future work, we will try to define more complex semantic word models by combining relation-based features (from WordNet or other lexical resources) with distributional-based representations. We will also explore the link between distributional representations and model-theoretical objects (entities, events, and so on), by considering bridging concepts such as *ideal distributions*.

The code for DepFunc and the distributional models used in the experiments are made freely available.[6]

ACKNOWLEDGMENTS

This work received financial support from project DOMINO (PGC2018-102041-B-I00, MCIU/AEI/FEDER, UE), eRisk (RTI2018-093336-B-C21), the Consellería de Cultura, Educación e Ordenación Universitaria (accreditation 2016-2019, ED431G/08), and the European Regional Development Fund (ERDF).

REFERENCES

Marco BARONI (2013), Composition in Distributional Semantics, *Language and Linguistics Compass*, 7:511–522.

Marco BARONI, Raffaella BERNARDI, and Roberto ZAMPARELLI (2014), Frege in Space: A Program for Compositional Distributional Semantics, *Linguistic Issues in Language Technology (LiLT)*, 9:241–346.

Marco BARONI, Silvia BERNARDINI, Adriano FERRARESI, and Eros ZANCHETTA (2009), The WaCky Wide Web: A Collection of Very Large Linguistically Processed Webcrawled Corpora, *Language Resources and Evaluation*, 43(3):209–226.

Marco BARONI and Roberto ZAMPARELLI (2010), Nouns Are Vectors, Adjectives Are Matrices: Representing Adjective-noun Constructions in Semantic Space, in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP'10, pp. 1183–1193, Stroudsburg, PA, USA.

Jon BARWISE (1987), Recent Developments in Situation Semantics, in M. NAGAO, editor, *Language and Artificial Intelligence*, pp. 387–399, North Holla.

Raffaella BERNARDI, Georgiana DINU, Marco MARELLI, and Marco BARONI (2013), A Relatedness Benchmark to Test the Role of Determiners in Compositional Distributional Semantics, in *The 51st Annual Meeting of the Association for Computational Linguistics ACL-2013*, pp. 53–57, The Association for Computational Linguistics.

Chris BIEMANN and Martin RIEDL (2013), Text: Now in 2D! A Framework for Lexical Expansion with Contextual Similarity, *Journal of Language Modelling*, 1(1):55–95.

[6] `http://fegalaz.usc.es/DepFunc.tgz`

B. COECKE, M. SADRZADEH, and S. CLARK (2010), Mathematical Foundations for a Compositional Distributional Model of Meaning, *Linguistic Analysis*, 36(1–4):345–384.

Ann COPESTAKE and Aurelie HERBELOT (2012), Lexicalised Compositionality, in *http://www.cl.cam.ac.uk/˜ah433/lc-semprag.pdf*, Unpublished article.

Fabrizio COSTA, Vincenzo LOMBARDO, Paolo FRASCONI, and Giovanni SODA (2001), Wide Coverage Incremental Parsing by Learning Attachment Preferences, in *Conference of the Italian Association for Artificial Intelligence (AIIA)*, pp. 297–307.

Georgiana DINU, Nghia PHAM, and Marco BARONI (2013a), DISSECT: DIStributional SEmantics Composition Toolkit, in *ACL 2013 Workshop on Continuous Vector Space Models and their Compositionality (CVSC 2013)*, pp. 31–36, East Stroudsburg PA.

Georgiana DINU, Nghia PHAM, and Marco BARONI (2013b), General Estimation and Evaluation of Compositional Distributional Semantic Models, in *ACL 2013 Workshop on Continuous Vector Space Models and their Compositionality (CVSC 2013)*, pp. 50–58, East Stroudsburg PA.

Ted DUNNING (1993), Accurate Methods for the Statistics of Surprise and Coincidence, *Computational Linguistics*, 19(1):61–74.

Katrin ERK (2013), Towards a Semantics for Distributional Representations, in *10th International Conference on Computational Semantics (IWCS 2013)*, pp. 95–106.

Katrin ERK and Sebastian PADÓ (2008), A Structured Vector Space Model for Word Meaning in Context, in *2008 Conference on Empirical Methods in Natural Language Processing (EMNLP-2008*, pp. 897–906, Honolulu, HI.

Katrin ERK and Sebastian PADÓ (2009), Paraphrase Assessment in Structured Vector Space: Exploring Parameters and Datasets, in *Proceedings of the EACL Workshop on Geometrical Methods for Natural Language Semantics*, pp. 57–65, Athens, Greece.

John Rupert FIRTH (1957), A synopsis of linguistic theory 1930-1955, *Studies in Linguistic Analysis*, pp. 1–32.

Pablo GAMALLO (2008), The Meaning of Syntactic Dependencies, *Linguistik OnLine*, 35(3):33–53.

Pablo GAMALLO (2015), Dependency Parsing with Compression Rules, in *Proceedings of the 14th International Workshop on Parsing Technology (IWPT 2015)*, pp. 107–117, Association for Computational Linguistics, Bilbao, Spain.

Pablo GAMALLO (2017a), Comparing Explicit and Predictive Distributional Semantic Models Endowed with Syntactic Contexts, *Language Resources and Evaluation*, 51(3):727–743.

Pablo GAMALLO (2017b), The Role of Syntactic Dependencies in Compositional Distributional Semantics, *Corpus Linguistics and Linguistic Theory*, 13(2):261–289.

Pablo GAMALLO (2017c), Sense Contextualization in a Dependency-Based Compositional Distributional Model, in *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pp. 1–9, Association for Computational Linguistics, doi:10.18653/v1/W17-2601, `http://aclweb.org/anthology/W17-2601`.

Pablo GAMALLO, Alexandre AGUSTINI, and Gabriel LOPES (2005), Clustering Syntactic Positions with Similar Semantic Requirements, *Computational Linguistics*, 31(1):107–146.

Pablo GAMALLO and Stefan BORDAG (2011), Is Singular Value Decomposition Useful for Word Simalirity Extraction, *Language Resources and Evaluation*, 45(2):95–119.

Pablo GAMALLO and Marcos GARCIA (2018), Dependency Parsing with Finite State Transducers and Compression Rules, *Information Processing & Management*, 54(6):1244–1261.

Pablo GAMALLO and Martín PEREIRA-FARIÑA (2017), Compositional Semantics using Feature-Based Models from WordNet, in *Proceedings of the 1st Workshop on Sense, Concept and Entity Representations and their Applications*, pp. 1–11, Association for Computational Linguistics, `http://aclweb.org/anthology/W17-1901`.

Dan GARRETTE, Katrin ERK, and Raymond MOONEY (2014), A Formal Approach to Linking Logical Form and Vector-Space Lexical Semantics, in H. BUNT, J. BOS, and S. PULMAN, editors, *Text, Speech and Language Technology: Computing Meaning*, pp. 27–48, Springer.

Edward GREFENSTETTE and Mehrnoosh SADRZADEH (2011a), Experimental Support for a Categorical Compositional Distributional Model of Meaning, in *Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pp. 1394–1404.

Edward GREFENSTETTE and Mehrnoosh SADRZADEH (2011b), Experimenting with Transitive Verbs in a DisCoCat, in *Workshop on Geometrical Models of Natural Language Semantics (EMNLP 2011)*.

Edward GREFENSTETTE, Mehrnoosh SADRZADEH, Stephen CLARK, Bob COECKE, and Stephen PULMAN (2011), Concrete Sentence Spaces for Compositional Distributional Models of Meaning, in *Proceedings of the Ninth International Conference on Computational Semantics*, IWCS '11, pp. 125–134.

Gregory GREFENSTETTE (1995), Evaluation Techniques for Automatic Semantic Extraction: Comparing Syntatic and Window Based Approaches, in Branimir BOGURAEV and James PUSTEJOVSKY, editors, *Corpus processing for Lexical Acquisition*, pp. 205–216, The MIT Press.

Emiliano GUEVARA (2010), A Regression Model of Adjective-Noun Compositionality in Distributional Semantics, in *Proceedings of the 2010 Workshop on GEometrical Models of Natural Language Semantics*, GEMS '10, p. 33–37.

Abhijeet GUPTA, Gemma BOLEDA, Marco BARONI, and Sebastian PADÓ (2015), Distributional Vectors Encode Referential Attributes, in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 12–21, Association for Computational Linguistics, Lisbon, Portugal, `http://aclweb.org/anthology/D15-1002`.

Kazuma HASHIMOTO and Yoshimasa TSURUOKA (2015), Learning Embeddings for Transitive Verb Disambiguation by Implicit Tensor Factorization, in *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pp. 1–11, Association for Computational Linguistics, Beijing, China, `http://www.aclweb.org/anthology/W15-4001`.

Richard HUDSON (2003), The Psychological Reality of Syntactic Dependency Relations, in *Proceedings of the First International Conference on Meaning-Text Theory*, pp. 181–192, Paris.

Ozan IRSOY and Claire CARDIE (2014), Deep Recursive Neural Networks for Compositionality in Language, in *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 2096–2104, `http://papers.nips.cc/paper/5551-deep-recursive-neural-networks-for-compositionality-in-language`.

Sylvain KAHANE (2003), Meaning-Text Theory, in *Dependency and Valency: An International Handbook of Contemporary Reseach*, Berlin: De Gruyter.

Hans KAMP and Uwe REYLE (1993), *From Discourse to Logic: Introduction to Model-theoretic Semantics of Natural Languge. Formal Logic and Discourse Representation Theory*, Kluwer Academic Publisher.

Dimitri KARTSAKLIS (2014), Compositional Operators in Distributional Semantics, *Springer Science Reviews*, 2(1–2):161–177.

Dimitri KARTSAKLIS, Nal KALCHBRENNER, and Mehrnoosh SADRZADEH (2014), Resolving Lexical Ambiguity in Tensor Regression Models of Meaning, in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Vol. 2: Short Papers)*, pp. 212–217, Association for Computational Linguistics, Baltimore, USA.

Dimitri KARTSAKLIS and Mehrnoosh SADRZADEH (2013), Prior Disambiguation of Word Tensors for Constructing Sentence Vectors, in *Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, pp. 1590–1601.

Ruth M. KEMPSON, Wilfried MEYER-VIOL, and Dov GABBAY (1997), Language Understanding: A Procedural Perspective, in C. RETORE, editor, *First*

International Conference on Logical Aspects of Computational Linguistics, pp. 228–247, Lecture Notes in Artificial Intelligence Vol. 1328. Springer Verlag.

Ruth M. KEMPSON, Wilfried MEYER-VIOL, and Dov GABBAY (2001), *Dynamic Syntax: The Flow of Language Understanding*, Blackwell, Oxford.

Thomas KOBER, Julie WEEDS, Jeremy REFFIN, and David J. WEIR (2016), Improving Sparse Word Representations with Distributional Inference for Semantic Composition, in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1–4, 2016*, pp. 1691–1702, http://aclweb.org/anthology/D/D16/D16-1175.pdf.

Jayant KRISHNAMURTHY and Tom MITCHELL (2013), Vector Space Semantic Parsing: A Framework for Compositional Vector Space Models, in *Proceedings of the Workshop on Continuous Vector Space Models and their Compositionality*, pp. 1–10, Association for Computational Linguistics.

Germán KRUSZEWSKI and Marco BARONI (2014), Dead Parrots Make Bad Pets: Exploring Modifier Effects in Noun Phrases, in *Proceedings of the Third Joint Conference on Lexical and Computational Semantics, *SEM@COLING 2014, August 23-24, 2014, Dublin, Ireland.*, pp. 171–181, http://aclweb.org/anthology/S/S14/S14-1021.pdf.

Ronald W. LANGACKER (1991), *Foundations of Cognitive Grammar: Descriptive Applications*, volume 2, Stanford University Press, Stanford.

Ken MCRAE, Todd R. FERRETI, and Liane AMYOTE (1997), Thematic Roles as Verb-specific Concepts, in M.C. MACDONALD, editor, *Lexical Representations and Sentence Processing*, pp. 137–176, Psychology Press.

Oren MELAMUD, Ido DAGAN, and Jacob GOLDBERGER (2015), Modeling Word Meaning in Context with Substitute Vectors, in *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pp. 472–482, http://aclweb.org/anthology/N/N15/N15-1050.pdf.

George A. MILLER, Richard BECKWITH, Christiane FELLBAUM, Derek GROSS, and Katherine J. MILLER (1990), Introduction to Wordnet: an On-Line Lexical Database, *International Journal of Lexicography*, 3(4):235–244.

David MILWARD (1992), Dynamics, Dependency Grammar and Incremental Interpretation, in *14th Conference on Computational Linguistics (COLING 1992)*, pp. 1095–1099, Nantes.

Jeff MITCHELL and Mirella LAPATA (2008), Vector-Based Models of Semantic Composition, in *Proceedings of the Association for Computational Linguistics: Human Language Technologies (ACL-08: HLT)*, pp. 236–244, Columbus, Ohio.

Jeff MITCHELL and Mirella LAPATA (2009), Language Models Based on Semantic Composition, in *Proceedings of Empirical Methods in Natural Language Processing (EMNLP-2009)*, pp. 430–439.

Jeff MITCHELL and Mirella LAPATA (2010), Composition in Distributional Models of Semantics, *Cognitive Science*, 34(8):1388–1439.

Richard MONTAGUE (1970), Universal Grammar, *Theoria*, 36(3):373–398.

Muntsa PADRÓ, Marco IDIART, Aline VILLAVICENCIO, and Carlos RAMISCH (2014), Nothing like Good Old Frequency: Studying Context Filters for Distributional Thesauri, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pp. 419–424.

Denis PAPERNO, Nghia The PHAM, and Marco BARONI (2014), A Practical and Linguistically-Motivated Approach to Compositional Distributional Semantics, in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 90–99, Association for Computational Linguistics, Baltimore, Maryland, `http://www.aclweb.org/anthology/P/P14/P14-1009`.

Nghia The PHAM, Germán KRUSZEWSKI, Angeliki LAZARIDOU, and Marco BARONI (2015), Jointly Optimizing Word Representations for Lexical and Sentential Tasks with the C-PHRASE Model, in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pp. 971–981, `http://aclweb.org/anthology/P/P15/P15-1094.pdf`.

Tamara POLAJNAR, Laura RIMELL, and Stephen CLARK (2015), An Exploration of Discourse-Based Sentence Spaces for Compositional Distributional Semantics, in *Proceedings of the First Workshop on Linking Computational Models of Lexical, Sentential and Discourse-level Semantics*, pp. 1–11, Association for Computational Linguistics, Lisbon, Portugal, `http://aclweb.org/anthology/W15-2701`.

James PUSTEJOVSKY (1995), *The Generative Lexicon*, MIT Press, Cambridge.

Siva REDDY, Ioannis P. KLAPAFTIS, Diana MCCARTHY, and Suresh MANANDHAR (2011), Dynamic and Static Prototype Vectors for Semantic Composition, in *Fifth International Joint Conference on Natural Language Processing, IJCNLP 2011, Chiang Mai, Thailand, November 8–13, 2011*, pp. 705–713, `http://aclweb.org/anthology/I/I11/I11-1079.pdf`.

Mehrnoosh SADRZADEH, Stephen CLARK, and Bob COECKE (2013), The Frobenius Anatomy of Word Meanings I: Subject and Object Relative Pronouns, *Journal of Logic and Computation*, 23(6):1293–1317, doi:10.1093/logcom/ext044, `http://dx.doi.org/10.1093/logcom/ext044`.

Matthias SCHLESEWSKY and Ina BORNKESSEL (2004), On Incremental Interpretation: Degrees of Meaning Accessed During Sentence Comprehension, *Lingua*, 114:1213–1234.

Richard SOCHER, Brody HUVAL, Christopher D. MANNING, and Andrew Y. NG (2012), Semantic Compositionality Through Recursive Matrix-vector Spaces, in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pp. 1201–1211, Association for Computational Linguistics, Stroudsburg, PA, USA, `http://dl.acm.org/citation.cfm?id=2390948.2391084`.

Mark STEEDMAN (1996), *Surface Structure and Interpretation*, The MIT Press.

Michael K. TANENHAUS and Greg CARLSON (1989), Lexical Structure and Language Comprehension, in William MARSLEN-WILSON, editor, *Lexical Representation and Process*, pp. 530–561, The MIT Press.

Stefan THATER, Hagen FÜRSTENAU, and Manfred PINKAL (2010), Contextualizing Semantic Representations Using Syntactically Enriched Vector Models, in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 948–957, Stroudsburg, PA, USA.

John TRUSWELL, Michael K. TANENHAUS, and Susan M. GARNSEY (1994), Semantic Influences on Parsing: Use of Thematic Role Information in Syntactic Ambiguity Resolution, *Journal of Memory and Language*, 33:285–318.

Peter D. TURNEY (2013), Domain and Function: A Dual-Space Model of Semantic Relations and Compositions, *Journal of Artificial Intelligence Research (JAIR)*, 44:533–585.

David J. WEIR, Julie WEEDS, Jeremy REFFIN, and Thomas KOBER (2016), Aligning Packed Dependency Trees: A Theory of Composition for Distributional Semantics, *Computational Linguistics*, 42(4):727–761.

Fabio Massimo ZANZOTTO, Ioannis KORKONTZELOS, Francesca FALLUCCHI, and Suresh MANANDHAR (2010), Estimating Linear Models for Compositional Distributional Semantics, in *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pp. 1263–1271.

Character-based recurrent neural networks for morphological relational reasoning

Olof Mogren[1] [*] *and Richard Johansson*[2]
[1] olof@mogren.one, RISE Research institutes of Sweden
[2] richard.johansson@gu.se, University of Gothenburg, Sweden

ABSTRACT

We present a model for predicting inflected word forms based on morphological analogies. Previous work includes rule-based algorithms that determine and copy affixes from one word to another, with limited support for varying inflectional patterns. In related tasks such as morphological reinflection, the algorithm is provided with an explicit enumeration of morphological features which may not be available in all cases. In contrast, our model is feature-free: instead of explicitly representing morphological features, the model is given a *demo pair* that implicitly specifies a morphological relation (such as *write:writes* specifying *infinitive:present*). Given this demo relation and a *query word* (e.g. *watch*), the model predicts the target word (e.g. *watches*). To address this task, we devise a character-based recurrent neural network architecture using three separate encoders and one decoder.

Our experimental evaluation on five different languages shows that the exact form can be predicted with high accuracy, consistently beating the baseline methods. Particularly, for English the prediction accuracy is 94.85%. The solution is not limited to copying affixes from the demo relation, but generalizes to words with varying inflectional patterns, and can abstract away from the orthographic level to the level of morphological forms.

Keywords: morphological analogies, morphological inflection, morphological reinflection, recurrent neural network, character-based modelling

[*] A large portion of this work was done while O.M. was a PhD student at Chalmers university of technology, Sweden.

1 INTRODUCTION

Analogical reasoning is an important part of human cognition (Gentner *et al.* 2001). Resolving analogies by mapping unknown data points to known analogous examples allows us to draw conclusions about the previously unseen data points. This is closely related to zero-shot and one-shot learning: strategies that are useful when training data is very limited, such as when explicit labels may not be available, or only sparsely available. In linguistics, analogies have been studied extensively, e.g. phonetic analogies (Yvon 1997) and semantic analogies (Mikolov *et al.* 2013a). In general, an analogy is defined as a quadruple of objects A, B, C, and D having the analogical relation: A is to B as C is to D, and the problem is to predict D given A, B, and C. In this work, we study morphological analogies where A, B, C, and D are words. The pair (A, B) represents a *demo relation* representing some morphological transformation between two word forms, and the problem is to transform the *query word* C from the source form to the target form as specified by the demo relation. The task may be illustrated with a simple example: *see* is to *sees* as *eat* is to what?

A good solver for morphological analogies can be of practical help as writing aids for authors, suggesting synonyms in a form specified by examples rather than using explicitly specified forms. Furthermore, models that can generate words with correct inflection are important building blocks for many tasks within natural language processing. Studying how systems can learn the right abstraction to generate inflected words using limited supervision, can help us learn how to create systems for more complex language generation tasks, such as machine translation, automatic summarization, and dialogue systems.

Previous work has tackled the problem of predicting the target form by identifying the string transformation (insertions, deletions, or replacements of characters) in the demo relation, and then trying to apply the same transformation to C (Lepage 1998). For instance, this algorithm correctly solves the example given above, since it just needs to add an *s* to the query word.

However, such solutions are brittle, as they are unable to abstract away from the raw strings, failing when the given relation is realized differently in A and B than in C and D. On a basic level, the model needs to take into account phonological processes such as umlaut and

vowel harmony, as well as orthographic quirks such as the rule in English that turns *y* into *ie* in certain contexts. Furthermore, an even more challenging problem is that the model will need to take into account that words belong to groups whose inflectional patterns are different – morphological *paradigms*. In all these cases, to be successful, a solution needs to abstract away from the raw character-based representation to a higher level representation of the relations.

In this work, we propose a supervised machine learning approach to the problem of predicting the target word in a morphological analogy. The model is based on character-level recurrent neural networks (RNNs), which have recently seen much success in a number of morphological prediction tasks (Faruqui *et al.* 2016; Kann and Schütze 2016). This model is able to go beyond the simple string substitutions handled by previous approaches: it picks up contextual string transformations including orthograpic and phonological rules, and is able to generalize between inflection paradigms.

Machine learning approaches, including character-based RNNs, have been successfully applied in several types of prediction problems in morphology, including lemmatization, inflection and reinflection (see Section 2.2). However, those tasks have either been more restricted than ours (e.g. lemmatization), or relied on an explicit enumeration of morphological features, which may not be available in all cases. In contrast, our model is a completely feature-free approach to generating inflected forms, which can predict any form in a morphological paradigm.

The fact that our model does not rely on explicit features makes it applicable in scenarios with under-resourced languages where such annotations may not be available. However, since the model is trained using a weaker signal than in the traditional feature-based scenario, it needs to learn a latent representation from the analogies that play the same role as the morphological features otherwise would, making the task more challenging.

2 RELATED WORK

Analogical reasoning is useful in many different tasks. In this section we will limit the survey to work that is relevant to morphological applications.

2.1 *Morphological analogies*

Lepage (1998) presented an algorithm to solve morphological analogies by analyzing three input words, determining changes in prefixes, infixes, and suffixes, and adding or removing them to or from the query word, transforming it into the target:

$$\textit{reader:}\underline{\textit{un}}\textit{read}\underline{\textit{able}} = \overline{\textit{do}}\textit{er:}\boldsymbol{x} \rightarrow \boldsymbol{x} = \underline{\textit{un}}\overline{\textit{do}}\underline{\textit{able}}.$$

Stroppa and Yvon (2005) presented algebraic definitions of analogies and a solution for *analogical learning* as a two-step process: learning a mapping from a memorized situation to a new situation, and transferring knowledge from the known to the unknown situation. The solution takes inspiration from k-nearest neighbour (k-NN) search, where, given a query q, one looks for analogous objects A, B, C from the training data, and selects a suitable output based on a mapping of A, B, C from input space to output space. The task studied in these papers is the same as in the current paper. The solutions, are however much limited in the generality. Our solution can learn very flexible relations and different inflectional patterns.

2.2 *Character based modeling for morphology*

The 2016 and 2017 SIGMORPHON shared tasks on *morphological reinflection* (Cotterell *et al.* 2016a, 2017) have spurred some recent interest in morphological analysis. In this task, a word is given in one form, and should be transformed into a form specified by an explicit feature representation. These features represent number, gender, case, tense, aspect, etc. In comparison, the problem of morphological analogies is more difficult, as no explicit tags are provided: the forms must instead be inferred from a demo relation.

While morphological inflection tasks have previously been studied using rule-based systems (Koskenniemi 1984; Ritchie *et al.* 1991) and learned string transducers (Yarowsky and Wicentowski 2000; Nicolai *et al.* 2015a; Ahlberg *et al.* 2015; Durrett and DeNero 2013), they have more recently been dominated by character-level neural network models (Faruqui *et al.* 2016; Kann and Schütze 2016) as they address the inherent drawbacks of traditional models that represent words as atomic symbols. This offers a number of advantages: the vocabulary in a character-based model can be much smaller, as it only

needs to represent a finite and fairly small alphabet, and as long as the characters are in the alphabet, no words will be out-of-vocabulary (OOV). Character-level models can capture distributional properties, not only of frequent words but also of words that occur rarely (Luong and Manning 2016), and they need no tokenization, freeing the system from one source of errors. Neural models working on character- or subword-level have been applied in several natural language processing (NLP) tasks, ranging from relatively basic tasks such as text categorization (Zhang *et al.* 2015) and language modelling (Kim *et al.* 2016) to complex prediction tasks, such as translation (Luong and Manning 2016; Sennrich *et al.* 2016). Because they can recognize patterns on a subword level, character-based neural models are attractive in NLP tasks that require an awareness of morphology.

2.3 *Other morphological transformations*

Lemmatization is the task of predicting the base form (lemma) of an inflected word. A lemmatizer may make use of the context to get (implicit) information about the source form of the word (Koskenniemi 1984; Kanis and Müller 2005; Chrupała *et al.* 2008; Jongejan and Dalianis 2009; Chakrabarty *et al.* 2017). In comparison, our task does not offer contextual information, but instead provides the (similarly implicit) cues for the forms from the demo relation. With this in mind, predicting the lemma is just a special case of the morphological analogy problem. *Paradigm filling* is the more general task of predicting all unknown forms in a paradigm (Dreyer and Eisner 2011).

2.4 *Morphological relations in word embedding models*

Word analogies have been proposed as a way to demonstrate the utility of neural word embeddings and to evaluate their quality (Mikolov *et al.* 2013a; Mnih and Kavukcuoglu 2013; Nicolai *et al.* 2015b; Pennington *et al.* 2014). Such embeddings show simple linear relationships in the resulting continuous embedding space that allow for finding impressive analogous relations such as

$$v(king) - v(man) + v(woman) \approx v(queen)$$

where $v(w)$ is the word embedding of the word w. Analogies have been categorized as either semantic or syntactic. (The example with “king” and “queen” is a semantic analogy, while syntactic analogies relate

different morphological forms of the same words). Google's dataset for syntactic analogies (Mikolov *et al.* 2013a) was proposed as a task to evaluate word embedding models on English.

Cotterell *et al.* (2016b) presented an approach using a Gaussian graphical model to process word embeddings computed using a standard toolkit such as Word2Vec to improve the quality of embeddings for infrequent words, and to construct embeddings for morphological forms that were missing in the training data (but belonging to a paradigm that had some form or forms in the data).

3 THE NEURAL MORPHOLOGICAL ANALOGY SYSTEM

In this paper, we present the Neural morphological analogy system (NMAS), a neural approach for morphological relational reasoning. We use a deep recurrent neural network with gated recurrent unit (GRU) cells that take words represented by their raw character sequences as input.

3.1 *Morphological relational reasoning with analogies*

We define the task as follows. Given a query word q and a demo word in two forms w_1 and w_2, demonstrating a transformation from one word form to another, and where q is another word in the same form as w_1, the task is to transform q into the form represented by w_2.

3.2 *Recurrent neural networks*

A recurrent neural network (RNN) is an artificial neural network that can model a sequence of arbitrary length. Gated RNNs were proposed to solve some issues of basic "vanilla" RNNs (the difficulty to capture long dependencies and vanishing gradients) (Hochreiter 1998; Bengio *et al.* 1994). The long short term memory (LSTM) (Schmidhuber and Hochreiter 1997) is one of the most famous types. At every step in the sequence, it has a cell with three learnable gates that controls what parts of the internal memory vector to keep (the forget gate), what parts of the input vector to store in the internal memory (the input gate), and what to include in the output vector (the output gate). The gated recurrent unit (GRU) (Cho *et al.* 2014a) is a simplification of this approach, having only two gates by replacing the input and

forget gates with an update gate that simply erases memory whenever it is updating the state with new input. Hence, the GRU has fewer parameters, and still obtains similar performance as the original LSTM.

An RNN can easily be trained to predict the next token in a sequence, and when applied to words this essentially becomes a language model. A sequence-to-sequence model is a neural language model conditioned on another input sequence. Such a model can be trained to translate from one sequence to another (Sutskever *et al.* 2014; Cho *et al.* 2014b). This is the major building block in modern neural machine translation systems, where they are combined with an attention mechanism to help with the alignment (Bahdanau *et al.* 2015).

In language settings it is common to have a linear input layer that learns embeddings for a vocabulary of words. However, these models suffer from the limitations of having fixed word vocabularies, and being unable to learn subword patterns. As an alternative, an RNN can work either using a vocabulary of subword units, or a vocabulary of characters, as is the case in this paper.

3.3 *Model layout*

The proposed model has three major parts, the relation encoder, the query encoder, and the decoder, all working together to generate the predicted target form given the three input words: the demo relation (w_1, w_2), and the query word q. The whole model is trained end-to-end and requires no other input than the raw character sequences of the three input words w_1, w_2, and q.

A. The relation encoder. The first part encodes the demo relation $R_{demo} = (w_1, w_2)$ using an encoder RNN for each of the two words w_1 and w_2. The relation encoder RNNs share weights but have separate internal state representations. The outputs of the relation encoders are fed into a fully connected layer with tanh activation *FC relation*:

$$\mathbf{h}_{rel} = \tanh(W_{rel}[g_{rel}(\mathbf{0}, \mathbf{w}_1), g_{rel}(\mathbf{0}, \mathbf{w}_2)]),$$

where g_{rel} is the output from the relation encoder RNN (using zero vectors as initial hidden states), $\mathbf{w}_1, \mathbf{w}_2$ are sequences of one-hot encodings for the characters of w_1 and w_2, W_{rel} is the weight matrix for

the *FC relation* layer, and tanh is the element-wise nonlinearity. Here, $[\mathbf{x}, \mathbf{y}]$ means the concatenation of the vectors $\mathbf{x}$ and $\mathbf{y}$.

B. The query encoder. The query word q is encoded separately using a distinct encoder RNN. The final output from the query encoder is fed together with the output from *FC relation* (A) through a second fully connected layer (with tanh activation) *FC combined*:

$$\mathbf{h}_{comb} = \tanh(W_{comb}[\mathbf{h}_{rel}, g_q(\mathbf{0}, \mathbf{q})]),$$

where $\mathbf{h}_{rel}$ is the output from *FC relation*, g_q is the output from the query RNN encoder, $\mathbf{q}$ is a sequence of embeddings of the characters of the query word, W_{comb} is the weight matrix for the *FC combined* layer, and tanh is the element-wise nonlinearity. The result $\mathbf{h}_{comb}$ is fed as the initial hidden state into the RNN decoder.

C. The decoder. The decoder RNN employs a standard attention mechanism (Bahdanau *et al.* 2015), computing a weighted sum of the sequence of outputs of the query encoder at every step t_d in the generation process. For each step t_e in the query encoder, the attention weight is computed using a multi-layer perceptron taking the decoder state at t_d and the query encoder state at t_e as inputs. For each decoder step t_d, the output character is decided by computing a distribution over the alphabet using the softmax output layer, and then sampling greedily from this distribution; this is fast and has yielded good results. The distribution $p(y_{t_d} = i) = \mathbf{h}^{(i)}_{dec;t_d}$ for each character i in the alphabet and for each step t_d in the decoder is modelled using:

$$\mathbf{h}_{dec;t_d} = s(W_{dec}[g_{dec}(\mathbf{h}_{comb}, \mathbf{y}_{(0:t_d-1)}), \mathbf{a}]),$$

where $\mathbf{h}_{comb}$ is the output from *FC combined* (used as the initial hidden state for the decoder RNN), g_{dec} is the output from the decoder RNN, $\mathbf{y}_{(0:t_d-1)}$ is a sequence of embeddings of the characters generated by the decoder until step $t_d - 1$, W_{dec} is the weight matrix for the decoder output layer, $\mathbf{a}$ is the weighted sum of hidden states from the query encoder RNN computed by the attention mechanism, and s is the softmax activation function: $s(\mathbf{z}) = \frac{e^{\mathbf{z}}}{\sum_i e^{\mathbf{z}(i)}}$. The result $\mathbf{h}_{dec;t_d}$ is a vector that sums to one, defining the distribution over the alphabet at time t_d.

The whole model is similar to a sequence-to-sequence model used for translation, with the addition of the relation encoder. Figure 1 shows the architecture of the model pictorially.

4 EXPERIMENTAL SETUP

This section explains the setup of the empirical evaluation of our model: how it is designed, trained, and evaluated.

The model was implemented using Pytorch;[1] all source code is freely available.[2] With the final hyperparameter settings (see Section 4), the model contains approximately 155,000 parameters, and it can be trained in a few hours on a modern GPU. In the experiments reported in this paper, the code was executed on a desktop PC with an NVIDIA Titan X GPU using Ubuntu 18.04.

The hyperparameters relevant to the proposed model are presented in Table 1. The RNN hidden size parameter decides the dimensionality of all four RNNs in the model, as we noticed no performance gain from varying them individually.

Hyperparameter	Explored	Selected
Embedding size	50–350	100
FC relation size	50–350	100
FC combined size	50–350	200
RNN hidden size	25–350	100
RNN depth	1–3	2
Learning rate		1×10^{-3}
L2 weight decay		5×10^{-5}
Drop probability	0.0, 0.2, ..., 0.8	0.0

Table 1:
Hyperparameters in the model

Training was done with backpropagation through time (BPTT) and minibatch learning with the Adam optimizer (Kingma and Ba 2015). For each example in a minibatch, a relation type is selected uniformly randomly. Then two word pairs are selected randomly from that relation type; one of these will be the demo relation, and one will be the query–target pair. The output from the decoder (see

[1] `http://pytorch.org/`

[2] `https://github.com/olofmogren/char-rnn-wordrelations`

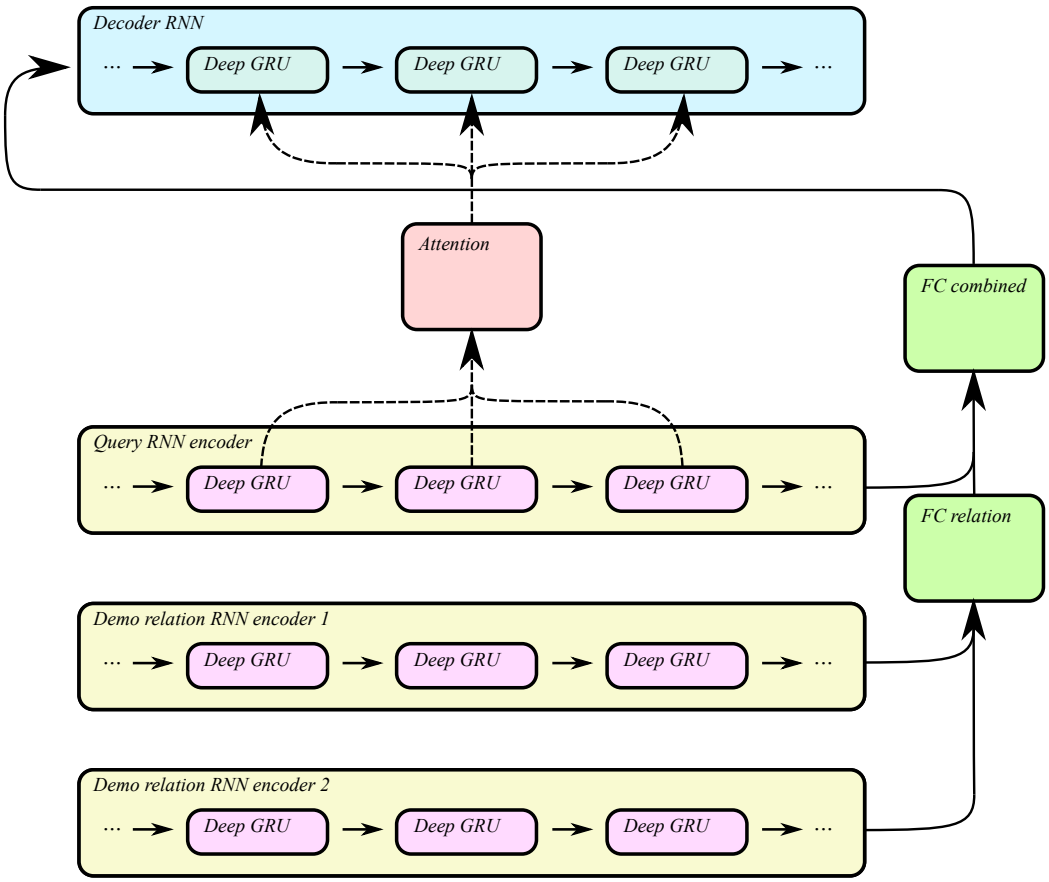


Figure 1: The layout of the proposed model. The demo relation is encoded using two encoder RNNs with shared weights for the two demo word forms. A fully connected (FC) layer *FC relation* follows the demo relation pair. The query word is encoded separately, and its embedding is concatenated with the output from the fully connected (FC) layer *FC relation*, and fed as the initial hidden state into the RNN decoder which generates the output while using an attention pointer to the query encoder. In the example *see* is to *sees* as *eat* is to *what*, *see* is fed into the *demo relation RNN encoder 1*, *sees* is fed into the *demo relation RNN encoder 2*, *eat* is fed into the *query RNN encoder*, and the whole model is trained to generate the correct output *eats* at the *decoder RNN*

Section 3.3 C) is a categorical distribution over the alphabet. We use the cross-entropy loss function for each character at position t_d in the output word as the learning objective for all parameters θ in the model:

$$\mathcal{L}(\theta) = \frac{-\sum_{t_d} \mathbf{y}_{(t_d)} \cdot \log \mathbf{h}_{\theta;t_d}}{N},$$

where N is the length of the target word, $\mathbf{y}_{(t_d)}$ is the one-hot encoding of the true target at position c and $\mathbf{h}_{\theta;t_d}$ is the model output distribution at position c. Training duration was decided using early stopping (Wang *et al.* 1994).

One model with separate parameters was trained per language. The parameters are shared between the two encoding RNNs in the relation encoder, but the query encoder RNN and the decoder RNN have separate weights, as the model search showed best performance using this configuration. Ensembling did not improve the results.

4.1 *Baselines*

The task considered in this work is closely related to morphological reinflection. Systems trained for the latter task generally obtain higher absolute numbers of prediction accuracy than ours, because more information is given through the explicit enumeration of morphological tags. Our task is also related to the syntactic analogy task used to evaluate word embeddings (Mikolov *et al.* 2013c), and we also include the word embedding-based word-level analogy solution as a baseline.

Lepage. This baseline was implemented from the description in (Lepage 1998). The algorithm is rule-based, and uses information collected when computing edit distance between w_1 and w_2, as well as between w_1 and q (Wagner and Fischer 1974). It can handle changes in prefix, infix, and suffix, but fails when words exhibit different inflectional patterns.

Word embedding baseline. This baseline uses pre-trained word embeddings using Word2Vec CBOW (continuous bag-of-words) (Mikolov *et al.* 2013b) and FastText (Bojanowski *et al.* 2017), referred to as W2V and FT, respectively. The FastText embeddings are designed to take subword information into account, and they performed better

than Word2Vec CBOW-based vectors in our experiments. The prediction is selected by choosing the word in the vocabulary that has an embedding with the highest cosine similarity compared to

$$v(q) - v(w_1) + v(w_2),$$

where $v(w)$ is the word embedding of the word w, q is the query word, and w_1, w_2 are the two demo words in the demo relation.

Word embeddings have been used in previous work for this task (then called syntactic analogies), but the solution is limited by a fixed vocabulary, and needs retraining to incorporate new words. Although it is trained without supervision, training requires much data and comparing the resulting vector above with all words in the vocabulary is expensive.

In this work, pretrained embeddings were downloaded and used. The utilized Word2Vec CBOW embeddings were downloaded from Kyubyong Park's repository.[3] The embeddings were trained using data from Wikipedia. The FastText embeddings used were downloaded from the FastText authors' website.[4] The embeddings were trained using data from CommonCrawl and Wikipedia. All the embeddings used have 300 dimensions.

To make the word embedding baseline stronger, we used the Lepage baseline as a fallback whenever any of the three input words are missing in the vocabulary.

4.2 *Datasets*

One model was trained and evaluated on each of five different languages. Data for all languages except for English and Swedish was taken from the SIGMORPHON 2016 dataset (Cotterell *et al.* 2016a). The code for downloading the data and performing dataset split is available for download.[5]

English. A total of 10 relations and their corresponding inverse relations were considered:

- nouns:
  - singular–plural, e.g. *dog–dogs*

[3] `https://github.com/Kyubyong/wordvectors/`

[4] `https://fasttext.cc/`

[5] `https://github.com/olofmogren/char-rnn-wordrelations/`

- adjectives:
  - positive–comparative, e.g. *high–higher*
  - positive–superlative, e.g. *high–highest*
  - comparative–superlative, e.g. *higher–highest*
- verbs:
  - infinitive–past, e.g. *sit–sat*
  - infinitive–present, e.g. *sit–sits*
  - infinitive–progressive, e.g. *sit–sitting*
  - past–present, e.g. *sit–sits*
  - past–progressive, e.g. *sit–sitting*
  - present–progressive, e.g. *sits–sitting*

For English, the dataset was constructed using the word list with inflected forms from the SCOWL project.[6] In the English data, 25,052 nouns, 1,433 adjectives, and 7,806 verbs were used for training. 1000 word pairs were selected randomly for validation and 1000 for testing, evenly distributed among relation types.

Swedish. Words were extracted from SALDO (Borin *et al.* 2013). In the Swedish data, 64,460 nouns, 12,507 adjectives, and 7,764 verbs were used for training. The division into training, validation, and test sets were based on the same proportions as in English. The same forms were used as in English, except that instead of the progressive form for verbs, the passive infinitive was used, e.g. *äta:ätas* 'eat:be eaten'.

Finnish, German, and Russian. For these languages, data from task1 and task2 in SIGMORPHON 2016 was used for training, and task2 data was used for evaluation. In this dataset, each word pair is provided along with morphological tags for the source and target words. We define a relation $\boldsymbol{R}$ as the combination of two sets of morphological tags, for which there exist words in the data.

The SIGMORPHON datasets consist of word pairs along with the corresponding morphological tags, specifying properties such as gender, number, case, and tense. For training set and validation set, we generate analogies from this as follows. First, we read each word pair

[6] See `http://wordlist.aspell.net/`.

(w_1, w_2) from the dataset, building tables of paradigms by storing the word pairs together with their tags. If w_1 or w_2 with the exact same set of tags has already been stored in a table (it may be part of another word pair (u_1, u_2)), then w_1 and w_2 is stored in the same table as u_1 and u_2. Second, when all words are stored in tables, we go through them and consider each pair of word forms members of a *morphological relation*. All words having a given source form and target form make up the set of word pairs for that relation. This procedure allows us to get more training data, as some new pairs can be generated from the tables (e.g. (w_1, u_1) from the example above). For the test set, we do not enhance the data in any such way, but use the exact relations provided from the original SIGMORPHON data set.

As the task described in this paper differs from the original SIGMORPHON task, with the additional requirement that every query–target word pair needs to be accompanied by a demo relation with the same forms, all relations with only one word pair were discarded. Of the SIGMORPHON datasets, we did not include Arabic, Georgian, Hungarian, Maltese, Navajo, Spanish, and Turkish, either because of the sparsity problem mentioned above,[7] or because the morphological features used in the language made it difficult to generate query–target pairs. The percentage of test set word pairs from the SIGMORPHON data being discarded in the remaining languages: Finnish: 1.2%, German: 2.1%, and Russian: 0.5%. Details about dataset sizes can be found in Table 2.

4.3 *Evaluation*

To evaluate the performance of the model, the datasets for English and Swedish were randomly split into training, validation, and test sets. Exact dataset split is defined within the openly shared code repository.[8] For the SIGMORPHON languages (Finnish, German, and Russian), the provided dataset split was used, and the test was performed as specified in the dataset, ignoring the specified morphological tags. For English and Swedish, each word pair was tested in both directions (switching the query word and the target word). Within one relation type, each word pair was randomly assigned another word pair as

[7] We decided on a threshold of at most 3% of the word pairs that could be discarded for the evaluation to be meaningful.

[8] `https://github.com/olofmogren/char-rnn-wordrelations/`

Table 2: Number of relations ("Rels", after discarding size-1 relations) and word pairs ("WPs") in the data set. *English and Swedish word pairs are all used exactly twice, once in original order, and once reversed. This means that the effective number of word pairs for these two languages are double the numbers in this table

Language	Training set		Validation set		Test set		Total	
	Rels	WPs	Rels	WPs	Rels	WPs	Rels	WPs
English	10	74,187	10	1,000	10	1,000	10	76,187*
Finnish	1,291	49,312	427	1,246	1,092	11,471	1,322	62,029
German	1,571	58,651	400	1,174	1,249	7,768	1,571	67,593
Russian	830	51,939	285	1,399	666	11,492	834	64,830
Swedish	10	146,551	10	1,000	10	1,000	10	148,551*

demo relation. Each word pair was used exactly once as a demo relation, and once as a query–target pair. Both word pairs in each analogy were selected from the same data partition; i.e. the test set for the evaluation. Relations having only one word pair were dropped from the test set, this is the only difference between the original SIGMORPHON test data and the test data used here (for more information, see Section 4.2). Where nothing else is specified, reported numbers are the prediction accuracy. This is the fraction of predictions that exactly match the target words.

4.4 *Data ambiguity*

As noted in Section 1, different words can have different inflectional patterns, and some words may also have the same expression for several forms. We note that there are such examples in the training data and in the validation data, but no such examples were detected in any of the test sets, see Table 3. This may affect the training, but not the evaluation of the system. When such ambiguities are presented as the target in demo relations, there is of course no way for a system with this setup to know which form to pick. However, the aim of our study was to keep the setup realistic, and hence, such ambiguous expressions were not removed from the datasets. Since no ambiguities were found in the test sets, there is always exactly one correct target for each query, but with a corresponding amount of

Table 3: Number of ambiguities detected in the data. This is the number of words that occur twice or more as a source word or as a target word in respective dataset partition

Language	Training set		Validation set		Test set		Total
	Twice	More	Twice	More	Twice	More	Twice or more
English	407	6	0	0	0	0	413
Finnish	20	0	0	0	0	0	20
German	415	0	2	0	0	0	417
Russian	186	0	0	0	0	0	186
Swedish	2,852	41	0	0	0	0	2,893

Table 4: Prediction accuracy and average Levenshtein distance of the proposed model (NMAS) trained using one language. Baseline: Lepage (1998)

Language	Accuracy		AVG Levenshtein	
	NMAS	Lepage	NMAS	Lepage
English	94.85%	56.05%	0.08	0.67
Finnish	85.64%	31.39%	0.22	1.76
German	87.96%	76.63%	0.20	0.39
Russian	75.85%	48.19%	0.36	1.01
Swedish	91.40%	64.80%	0.15	0.60

ambiguous data in the training set, the model may learn robustness and the noise provided by the ambiguities may also help to regularize the training.

5 RESULTS

This section presents the results of the experimental evaluation of the system.

5.1 *Language-wise performance*

The prediction accuracy results for the test set can be seen in Table 4, reaching an accuracy of 94.85% for English. While Finnish is a morphologically rich language, with 1323 distinct relations in the dataset, and with the lowest *Lepage* baseline score of all evaluated languages

(31.39%), NMAS is able to learn its relations rather well, with a prediction accuracy of 85.64%. For German and Swedish, the performance is 87.96% and 91.40%, respectively. They both have more complex morphologies with more inflectional patterns for nouns and verbs. On Russian, NMAS obtains an accuracy of 75.85%. This may be explained by its complex morphology and phonology, and is consistent with the results of top scoring systems on the SIGMORPHON tasks.

5.1.1 Detailed analysis of phonological and orthographic regularities

To successfully predict inflected word forms, our model must take the inflectional paradigms into account, as well as the orthographic and the phonological regularities that sometimes cut across the paradigms. We will now consider a number of examples of such regularities for all five languages and investigate how well they are handled by our model.

English. A basic textbook example of an orthographic rule conditioned on the immediate context is the *y/ie* alternation in English, such as *fry/fries*, *hurry/hurried*, etc. This simple regularity poses no difficulty for our model which predicted the correct form in all cases where such alternations occurred in the data.

Finnish. A more interesting case is the phonology of Finnish, which is well-known for its *vowel harmony* and *consonant gradation*. To investigate how well the model handles vowel harmony, we considered instances where the final vowel (which typically corresponds to the inflection) in the gold-standard output is either in the back-vowel group (*a*, *o*, or *u*) or the front-vowel group (*ä*, *ö*, or *y*). In these cases, the model predicts the correct vowel of the output form in 97% of the cases. This figure is identical for the subset of instances where vowel in the output is in a different group from the corresponding vowel in the demo output, which occurs in about 18% of the instances. It is notable that several of the model's vowel harmony errors correspond to exceptions where the usual rules of vowel harmony do not apply: (1) in a compound such as *hiuspinnit* ('hairpins'), the model is confused by the back vowel *u* in the compound prefix, and incorrectly predicts an *a* instead of *ä* in inflections; (2) a non-native word such as *desideratiivi* ('desiderative') may take an "unexpected" vowel (in this case the inflections use front vowels despite the preceding *a*).

In consonant gradation, consonants or consonant clusters may appear in either the *strong* or the *weak* grade, depending on the phonological context. For instance, the cluster *rt* in *parta* ('beard' in the nominative singular) is in the strong grade, which in the weak grade becomes *rr*, as in *parrat* (nominative plural). We selected the occurrences where the query word and the gold-standard output differed in grade (about 15% of the instances). The correct grade is predicted by the model in 84% of these cases, and this does not seem to be affected by whether the demo pair involves consonant gradation or not.

German. For German, we considered umlaut alternations such as *Baum/Bäume*. We selected the instances where there is an umlaut vowel in either the query word or in the gold-standard output, but not both. This is about 2.7% of the instances. In such cases, the model predicts the vowel correctly just 27% of the time. This can be contrasted with results we saw for Finnish, where vowel harmony was handled almost perfectly. This difference is probably due to the unpredictability and rarity of the German umlaut, while the Finnish vowel harmony is very common and almost perfectly regular.

Swedish. Some words in Swedish exhibit the process of syncope where the unstressed vowel *e* is lost in some contexts. For instance, the word *nyckel* ('key') has the plural form *nycklar*. We found 36 instances involving a syncope of the query word or the gold-standard output. This corresponds to 1.8% of the total set. The model predicts the correct form in 94% of these cases.

Russian. This is the language which has proven to be the most problematic for our model, due to the rich system of its inflectional paradigms, as well as the complex phonological processes (e.g. palatalization) affecting some of the inflections. While irregularities are challenging for the model, it successfully handles phenomena that are more predictable. As an example of a regular pattern that the model handles well, we can consider the clitic used with reflexive verbs, which takes the form *-sya* or *-s'* depending on the phonological context. The form of this clitic is predicted correctly by our model 98% of the time.

5.2 Model variants

Attend to relation. Kann and Schütze (2016) explicitly feeds the morphological tags as special tokens being part of the input sequence, and the attention mechanism learns when to focus on the forms during output generation. Inspired by this we decided to evaluate a variant of our model where the embedding of the relation encoder is appended to the query encoder output sequence, allowing the decoder to attend to the whole query as well as the relation embedding, see Figure 2. The per-

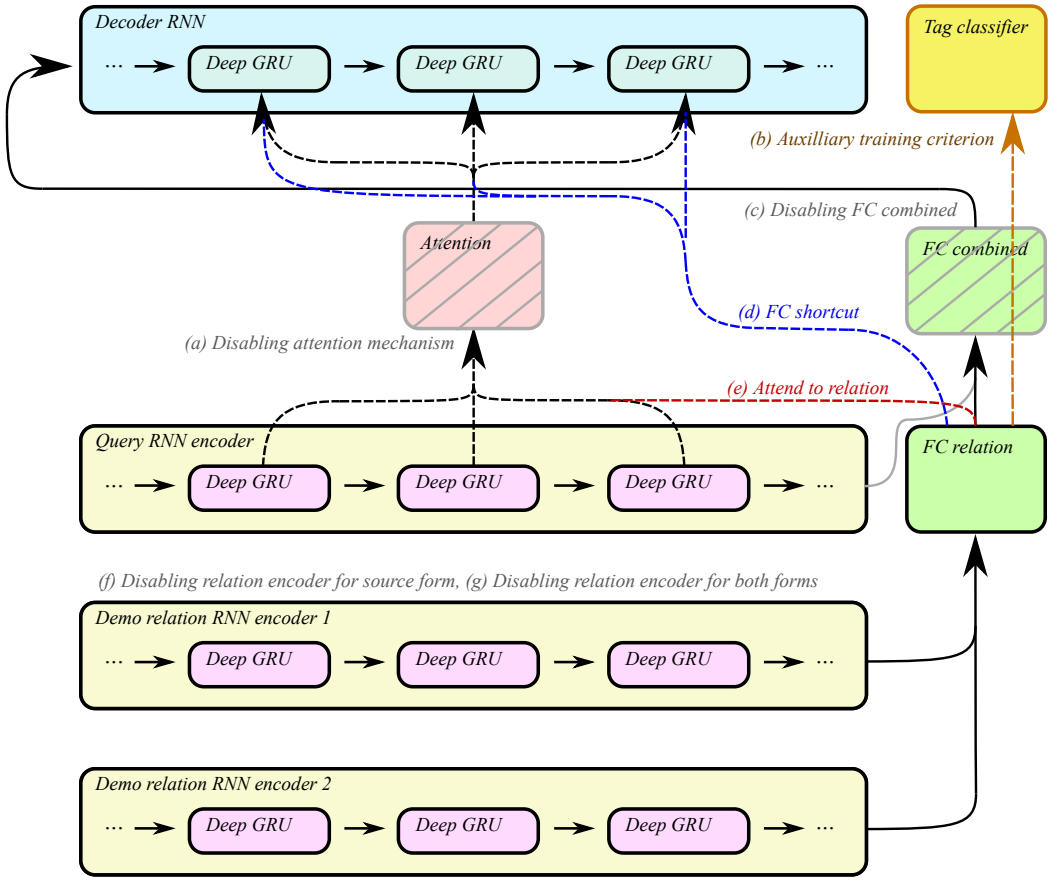


Figure 2: The evaluated variants of the proposed model. (a) disabling attention mechanism, (b) auxiliary training criterion, (c) disabling FC combined, (d) FC shortcut, (e) attend to relation, (f) disabling relation encoder for source form, and (g) disabling relation encoder for both forms

Table 5: Prediction accuracy of the proposed model trained with "attend to relation", with and without the relation embedding fed to initial hidden state (*FC combined*), all words reversed, feeding the relation embedding using a shortcut to each step in the decoder RNN, and using auxiliary tags classification criterion, respectively. English validation set

Variant	Validation accuracy
Full model	**96.40%**
Attend to relation	96.20%
Attend to relation & No *FC combined*	95.35%
Reversed words	96.00%
Relation shortcut	95.40%
Auxiliary tags classification	95.25%

formance of the model was not affected by this change (see Table 5), and there was no clear trend spanning over different languages. When also disabling *FC combined*, and thus feeding the relation embedding directly as input to the decoder, there was a noticeable decrease in performance: 95.35% accuracy on the English validation set.

Relation shortcut. In the layout of the proposed model, the information from the relation encoder is available to the decoder only initially. To explore if it would help to have the information available at every step in the decoding process, a shortcut connection was added from *FC relation* to the final layer in the decoder. This helped the model to start learning fast (see Figure 3), but then resulted in a slight decrease in accuracy (95.25% on English validation set). (See Table 5).

Auxiliary training criterion. Multi-task learning using a related *auxiliary task* can lead to stronger generalization and better regularized models (Caruana 1998; Collobert and Weston 2008; Bingel and Søgaard 2017). We evaluated a model that used an auxiliary training task: the model had to predict the morphological tags as an output from the relation encoder. This addition gave a slight initial training speedup (see Figure 3), but did not give a better performing model once the model finished training. This indicates a strength in the originally proposed solution: the model can learn to differentiate the morphological forms of the words in the demo relation, even

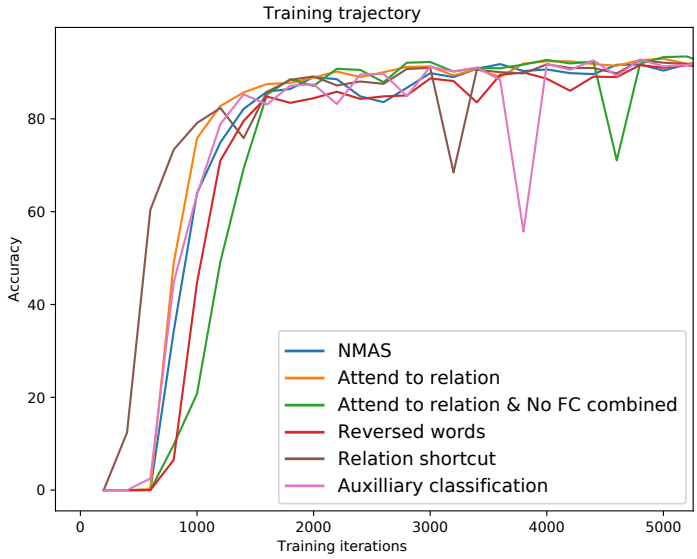


Figure 3: Prediction accuracy on the English validation set during training for some variations of the model

without having this explicit training signal, something that is also demonstrated by the visualized relation embeddings (see Figure 4).

Disabling model components. The relation encoder learns to represent the different morphological relations with nicely disentangled embeddings (see Figure 4). The fact that the prediction accuracy drops as far as to 39.35% when disabling the relation input (see Table 6) indicates that the setup is useful, and that the model indeed learns to utilize the information from the demo relation. Disabling only the first word in the demo relation allows the model to perform much better (94.60% validation accuracy), but it does not reach the accuracy of the full model with both demo words (96.40%). Disabling the attention mechanism is a small modification of our model, but it substantially degrades performance, resulting in 91.90% accuracy on the English validation set.

5.3 *Mechanisms of word inflection*

As English (and many other languages) forms inflections mainly by changing suffixes, an experiment was performed where every word was reversed (e.g. *“requirement”* → *“tnemeriuqer”*), to evaluate

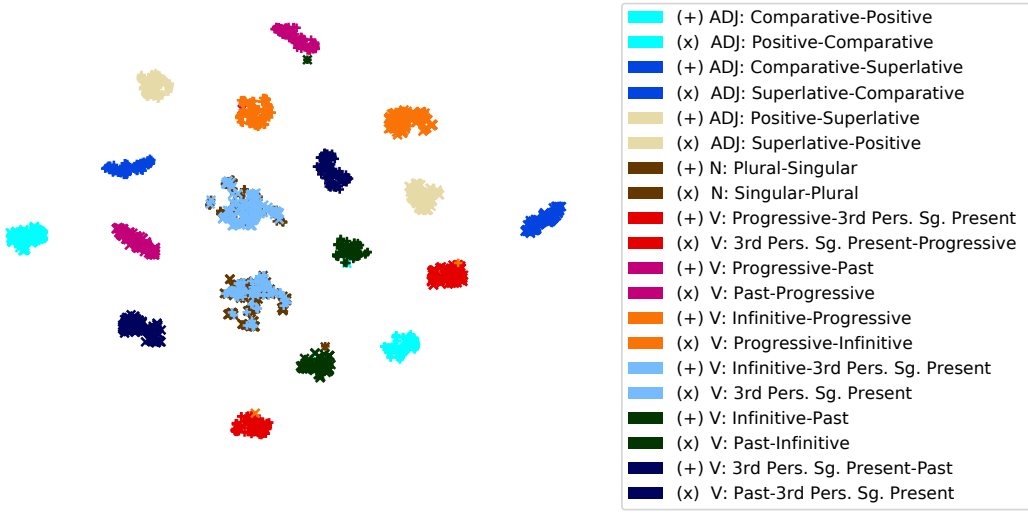


Figure 4: t-SNE visualization of all demo relation pairs from English validation set embedded using the relation encoder. Each point is colored by the relation type that it represents

Table 6: Prediction accuracy of the proposed model without attention mechanism, without the first (source) word in the demo relation, and completely without demo relation encoder, respectively. English validation set

Variant	Validation accuracy
Full model	**96.40%**
Disable attention mechanism	91.90%
Disable relation source	94.60%
Disable relation input	39.35%

whether the model can cope with other mechanisms of word inflection. On this data, NMAS obtains a prediction accuracy that is only slightly worse than the original version (96.00% on English validation set). This indicates that the model can cope with different kinds of inflectional patterns (i.e. suffix and prefix changes). As can be noted in the example outputs (see Table 7), the model does handle several different kinds of inflections (including orthographic variations such as *y/ie*), and it does not require the demo relation to show

Table 7: Correct (top), and incorrect (bottom) example outputs from the model. Samples from English validation set

Correct:

Demo word 1	Demo word 2	Query	Target	Output
misidentify	misidentifies	bottleneck	bottlenecks	bottlenecks
obliterate	obliterated	prig	prigged	prigged
ventilating	ventilates	disorganizing	disorganizes	disorganizes
crank	cranker	freckly	frecklier	frecklier
debauchery	debaucheries	bumptiousness	bumptiousnesses	bumptiousnesses

Incorrect:

Demo word 1	Demo word 2	Query	Target	Output
repackage	repackaged	outrun	outran	outrunned
misinformed	misinform	gassed	gas	gass
julep	juleps	catfish	catfish	catfishes
cedar	cedars	midlife	midlives	midlifes
affrays	affray	buzzes	buzz	buzze

the same inflectional pattern as the query word. In fact, often when the system fails, it does so by inflecting irregular words in a regular manner, suggesting that patterns with less data availability poses the major problem.

5.4 *Relation embeddings*

Figure 4 shows a t-SNE visualization of the embeddings from the relation encoder (*"FC relation"*) of all data points in the English validation set. One can see that most relations have been clearly separated into one distinct cluster each, with the exception of two clusters, both containing points from two relations each. The first such cluster contains the two relation types *"N: Singular-Plural"* and *"V: Infinitive-3 Pers. Sg. Present"*; both of these are realized in English by appending the suffix *-s* to the query word. The second cluster contains the relation types *"N: Plural-Singular"* and *"V: 3 Pers. Sg. Present-Infinitive"*; both of these are realized by the removal of the suffix *-s*. It is worth noting that no ex-

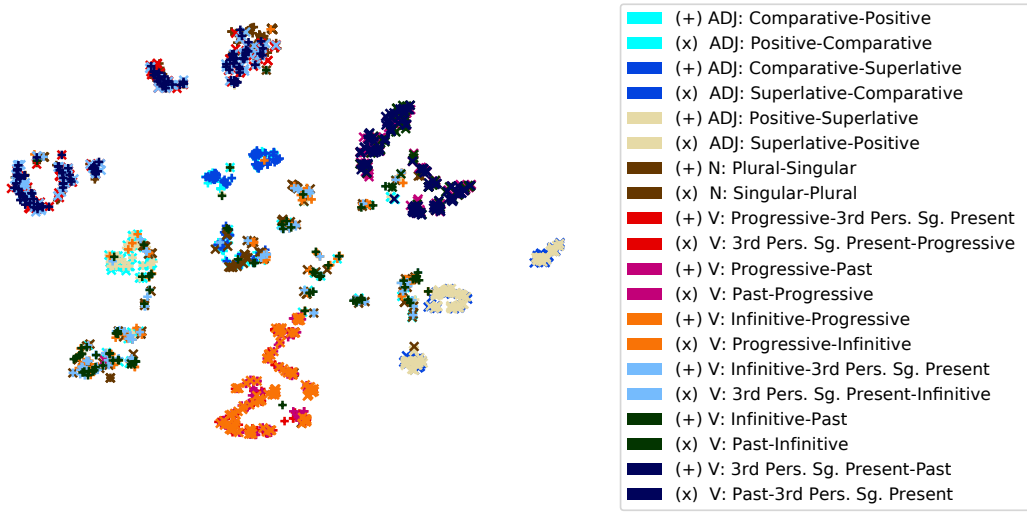


Figure 5: t-SNE visualization of all query words from English validation set embedded using the query encoder. Each point is coloured by the relation type that it represents

plicit training signal has been provided for this to happen. The model has learned to separate different morphological relations to help with the downstream task.

Similar clustered representations can be seen when analysing the embeddings computed by the relation encoder RNN also for other languages. We refer interested readers to the supplemental material[9] for a complete list of these plots.

Figure 5 shows a t-SNE visualization of the embeddings from the query encoder. As we saw with the relation encoder, query embeddings seem to encode information about morphology as similar morphological forms cluster together, albeit with more internal variation and more inter-cluster overlaps. The task for the query encoder is more complex as it needs to encode all information about the query word and provide information on how it may be transformed. To solve the task, and be able to correctly transform query words with the same relation type but with different inflection patterns, it needs to be able to deduce what subcategory of a relation a given query word belongs to.

[9] `http://bit.ly/2oyPEtX`

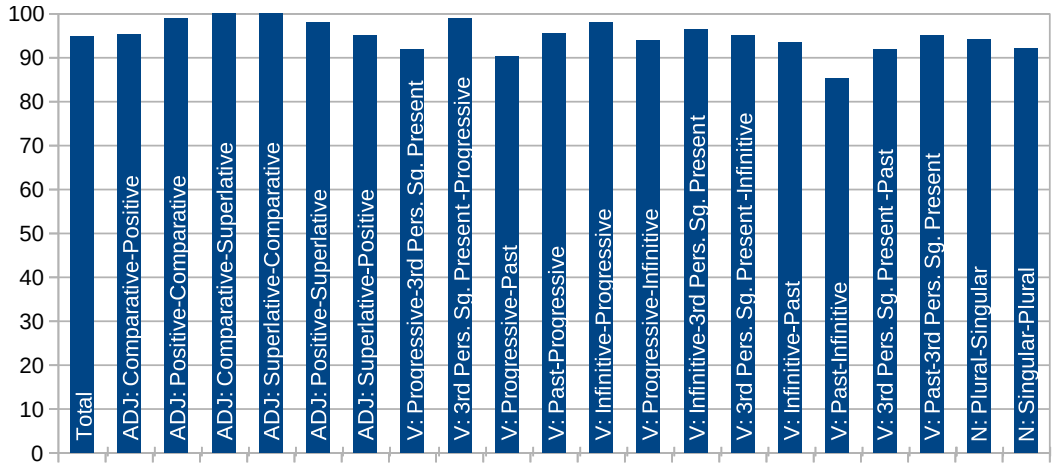


Figure 6: Results for all relations (total), and for each specific relation of the English test set

5.5 *Word embedding analogies*

The Lepage baseline proved to be the strongest baseline for all languages. For instance, for English it obtains prediction accuracy of 56.05%, compared to 40.75% for the Word2Vec baseline, and 45.00% for the FastText baseline. Without the Lepage fallback, the Word2Vec baseline scored 14.45%, and the FastText baseline scored 22.75%. For other languages, the results were even worse. The datasets in our study contain a rather large vocabulary, not only including frequent words. While the fixed vocabulary is one of the major limitations (explaining the difference between the embedding baselines and the corresponding ones without fallback), the word embedding baseline predictions were often incorrect even when the words were included in the vocabulary. This led us to use the Lepage baseline in the result tables.

5.6 *Relation-wise performance*

Figure 6 shows the performance for each relation type, showing that our model obtains 100% test set accuracy for the transforms between *comparative–superlative*. It obtains the lowest accuracy (85.19%) for *past–infinitive*, 94.17%, and 92.23% for *plural–singular*, and *singular–plural*, respectively. From Figure 4 we have learned that these very relations are the most difficult ones for the relation encoder to distin-

guish between. One difficulty of *plural–singular* seems to be to determine how many character to remove, while the patterns for adding the *-s* suffix is generally simpler. An example demonstrating this can be seen in Table 7: *buzzes:buzz*, where the model incorrectly predicted *buzze*.

5.7 *Example outputs*

We have collected some examples from the English validation set where our model succeeds and where it fails (see Table 7). Examples of patterns that can be observed in the failed examples are (1) words with irregular inflections that the model incorrectly inflects using regular patterns, e.g. *outrun:outran*, where the model predicted *outrunned*; (2) words with ambiguous targets, e.g. *gassed:gas*, where the model predicted *gass*. If there existed a verb *gass*, it could very well have been *gassed* in its past-tense form. Tables with example output for the other studied languages are provided in the supplemental material.[10] In general: the model can learn different inflectional patterns. Suffixes, infixes, and prefixes do not pose problems. The query word does not need to have the same inflectional pattern as the demo relation. When the model does fail, it is often due to an inflection that is not represented in the training data, such as irregular verbs.

6 DISCUSSION AND CONCLUSIONS

In this paper, we have presented a neural model that can learn to carry out *morphological relational reasoning* on a given query word q, given a demo relation consisting of a word in two different forms (source form and desired target form). Our approach uses a character based encoder RNN for the demo relation words, and one for the query word, and generates the output word as a character sequence. The model is able to generalize to unseen words as demonstrated by good prediction accuracy on the held-out test sets in five different languages: English, Finnish, German, Russian, and Swedish. It learns representations that separate the relations well provided only with the training signal given by the task of generating the words in correct form.

[10] `http://bit.ly/2oyPEtX`

Our solution is more general than existing methods for morphological inflection and reinflection, in the sense that they require explicit enumeration of the morphological tags specifying the transformation; our solution instead learns to build its own internal representation of this information by observing an analogous word pair demonstrating the relation.

ACKNOWLEDGMENTS

RJ was supported by the Swedish Research Council under grant 2013–4944. OM was supported by Swedish Foundation for Strategic Research (SSF) under grant IIS11-0089.

REFERENCES

Malin AHLBERG, Markus FORSBERG, and Mans HULDEN (2015), Paradigm classification in supervised learning of morphology, in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1024–1029, Association for Computational Linguistics, Denver, United States, doi:10.3115/v1/N15-1107.

Dzmitry BAHDANAU, Kyunghyun CHO, and Yoshua BENGIO (2015), Neural machine translation by jointly learning to align and translate, in *Proceedings of the 3rd International Conference on Learning Representations, ICLR, Conference Track Proceedings*, San Diego, United States.

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, doi:10.1109/72.279181.

Joachim BINGEL and Anders SØGAARD (2017), Identifying beneficial task relations for multi-task learning in deep neural networks, in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pp. 164–169, Association for Computational Linguistics, Valencia, Spain.

Piotr BOJANOWSKI, Edouard GRAVE, Armand JOULIN, and Tomas MIKOLOV (2017), Enriching word vectors with subword information, *Transactions of the Association for Computational Linguistics*, 5:135–146, doi:10.1162/tacl_a_00051.

Lars BORIN, Markus FORSBERG, and Lennart LÖNNGREN (2013), SALDO: a touch of yin to WordNet's yang, *Language Resources and Evaluation*, 47(4):1191–1211, doi:10.1007/s10579-013-9233-4.

Rich CARUANA (1998), Multitask learning, in *Learning to Learn*, pp. 95–133, Springer US, Boston, MA, doi:10.1007/978-1-4615-5529-2_5.

Abhisek CHAKRABARTY, Onkar Arun PANDIT, and Utpal GARAIN (2017), Context sensitive lemmatization using two successive bidirectional gated recurrent networks, in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1481–1491, Association for Computational Linguistics, Vancouver, Canada, doi:10.18653/v1/P17-1136.

Kyunghyun CHO, Bart VAN MERRIËNBOER, Dzmitry BAHDANAU, and Yoshua BENGIO (2014a), On the properties of neural machine translation: Encoder–decoder approaches, in *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pp. 103–111, Association for Computational Linguistics, Doha, Qatar, doi:10.3115/v1/W14-4012.

Kyunghyun CHO, Bart VAN MERRIËNBOER, Caglar GULCEHRE, Dzmitry BAHDANAU, Fethi BOUGARES, Holger SCHWENK, and Yoshua BENGIO (2014b), Learning phrase representations using RNN encoder–decoder for statistical machine translation, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, Association for Computational Linguistics, Doha, Qatar, doi:10.3115/v1/D14-1179.

Grzegorz CHRUPAŁA, Georgiana DINU, and Josef VAN GENABITH (2008), Learning morphology with Morfette, in *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, pp. 2362–2367, European Language Resources Association (ELRA), Marrakech, Morocco.

Ronan COLLOBERT and Jason WESTON (2008), A unified architecture for natural language processing: Deep neural networks with multitask learning, in *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pp. 160–167, ACM, Helsinki, Finland, doi:10.1145/1390156.1390177.

Ryan COTTERELL, Christo KIROV, John SYLAK-GLASSMAN, Géraldine WALTHER, Ekaterina VYLOMOVA, Patrick XIA, Manaal FARUQUI, Sandra KÜBLER, David YAROWSKY, Jason EISNER, and Mans HULDEN (2017), CoNLL-SIGMORPHON 2017 shared task: universal morphological reinflection in 52 languages, in *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pp. 1–30, Association for Computational Linguistics, Vancouver, Canada.

Ryan COTTERELL, Christo KIROV, John SYLAK-GLASSMAN, David YAROWSKY, Jason EISNER, and Mans HULDEN (2016a), The SIGMORPHON 2016 shared task – morphological reinflection, in *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pp. 10–22, Association for Computational Linguistics, Berlin, Germany, doi:10.18653/v1/W16-2002.

Ryan COTTERELL, Hinrich SCHÜTZE, and Jason EISNER (2016b), Morphological smoothing and extrapolation of word embeddings, in *Proceedings*

of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1651–1660, Association for Computational Linguistics, Berlin, Germany.

Markus DREYER and Jason EISNER (2011), Discovering morphological paradigms from plain text using a Dirichlet process mixture model, in *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pp. 616–627, Association for Computational Linguistics, Edinburgh, United Kingdom.

Greg DURRETT and John DENERO (2013), Supervised learning of complete morphological paradigms, in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1185–1195, Association for Computational Linguistics, Atlanta, United States.

Manaal FARUQUI, Yulia TSVETKOV, Graham NEUBIG, and Chris DYER (2016), Morphological inflection generation using character sequence to sequence learning, in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 634–643, Association for Computational Linguistics, San Diego, United States, doi:10.18653/v1/N16-1077.

Dedre GENTNER, Keith James HOLYOAK, and Boicho N. KOKINOV (2001), *The analogical mind: Perspectives from cognitive science*, MIT press.

Sepp HOCHREITER (1998), The vanishing gradient problem during learning recurrent neural nets and problem solutions, *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 6(2):107–116, doi:10.1142/S0218488598000094.

Bart JONGEJAN and Hercules DALIANIS (2009), Automatic training of lemmatization rules that handle morphological changes in pre-, in- and suffixes alike, in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pp. 145–153, Association for Computational Linguistics, Suntec, Singapore.

Jakub KANIS and Luděk MÜLLER (2005), Automatic lemmatizer construction with focus on OOV words lemmatization, in *Text, Speech and Dialogue*, pp. 132–139, Springer Berlin Heidelberg, Berlin, Heidelberg.

Katharina KANN and Hinrich SCHÜTZE (2016), MED: The LMU System for the SIGMORPHON 2016 shared task on morphological reinflection, in *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pp. 62–70, Association for Computational Linguistics, Berlin, Germany, doi:10.18653/v1/W16-2010.

Yoon KIM, Yacine JERNITE, David SONTAG, and Alexander M. RUSH (2016), Character-aware neural language models, in *Proceedings of the Thirtieth AAAI*

Conference on Artificial Intelligence, AAAI'16, pp. 2741–2749, AAAI Press, Phoenix, United States.

Diederik KINGMA and Jimmy BA (2015), Adam: a method for stochastic optimization, in *Proceedings of the 3rd International Conference on Learning Representations, ICLR, Conference Track Proceedings*, San Diego, United States.

Kimmo KOSKENNIEMI (1984), A general computational model for word-form recognition and production, in *10th International Conference on Computational Linguistics and 22nd Annual Meeting of the Association for Computational Linguistics*, pp. 178–181, Association for Computational Linguistics, Stanford, United States, doi:10.3115/980491.980529.

Yves LEPAGE (1998), Solving analogies on words: an algorithm, in *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pp. 728–734, Association for Computational Linguistics, Montreal, Canada, doi:10.3115/980845.980967.

Minh-Thang LUONG and Christopher D. MANNING (2016), Achieving open vocabulary neural machine translation with hybrid word-character models, in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1054–1063, Association for Computational Linguistics, Berlin, Germany, doi:10.18653/v1/P16-1100.

Tomas MIKOLOV, Kai CHEN, Greg CORRADO, and Jeffrey DEAN (2013a), Efficient estimation of word representations in vector space, in *Proceedings of the International Conference on Learning Representations (ICLR), Workshop Track*, Scottsdale, United States.

Tomas MIKOLOV, Ilya SUTSKEVER, Kai CHEN, Greg CORRADO, and Jeffrey DEAN (2013b), Distributed representations of words and phrases and their compositionality, in *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pp. 3111–3119, Curran Associates Inc., Lake Tahoe, United States.

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

Andriy MNIH and Koray KAVUKCUOGLU (2013), Learning word embeddings efficiently with noise-contrastive estimation, in *Advances in Neural Information Processing Systems 26*, pp. 2265–2273, Curran Associates, Inc.

Garrett NICOLAI, Colin CHERRY, and Grzegorz KONDRAK (2015a), Inflection generation as discriminative string transduction, in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational*

Linguistics: Human Language Technologies, pp. 922–931, Association for Computational Linguistics, Denver, Colorado, doi:10.3115/v1/N15-1093.

Garrett NICOLAI, Colin CHERRY, and Grzegorz KONDRAK (2015b), Morpho-syntactic regularities in continuous word representations: A multilingual study, in *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pp. 129–134, Association for Computational Linguistics, Denver, United States, doi:10.3115/v1/W15-1518.

Jeffrey PENNINGTON, Richard SOCHER, and Christopher MANNING (2014), GloVe: global vectors for word representation, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, Association for Computational Linguistics, Doha, Qatar, doi:10.3115/v1/D14-1162.

Graeme D. RITCHIE, Graham J. RUSSELL, Alan W. BLACK, and Stephen G. PULMAN (1991), *Computational morphology: practical mechanisms for the English lexicon*, ACL-MIT Series in Natural Language Processing, MIT Press, Cambridge, United States.

Jürgen SCHMIDHUBER and Sepp HOCHREITER (1997), Long short-term memory, *Neural Computation*, 9(8):1735–1780, doi:10.1162/neco.1997.9.8.1735.

Rico SENNRICH, Barry HADDOW, and Alexandra BIRCH (2016), Neural machine translation of rare words with subword units, in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1715–1725, Association for Computational Linguistics, Berlin, Germany, doi:10.18653/v1/P16-1162.

Nicolas STROPPA and François YVON (2005), An analogical learner for morphological analysis, in *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pp. 120–127, Association for Computational Linguistics, Ann Arbor, United States.

Ilya SUTSKEVER, Oriol VINYALS, and Quoc V. LE (2014), Sequence to sequence learning with neural networks, in *Advances in Neural Information Processing Systems 27*, pp. 3104–3112, Curran Associates, Inc.

Robert A. WAGNER and Michael J. FISCHER (1974), The string-to-string correction problem, *J. ACM*, 21(1):168–173, doi:10.1145/321796.321811.

Changfeng WANG, Santosh S. VENKATESH, and J. Stephen JUDD (1994), Optimal stopping and effective machine complexity in learning, in *Advances in Neural Information Processing Systems 6*, pp. 303–310, Morgan-Kaufmann.

David YAROWSKY and Richard WICENTOWSKI (2000), Minimally supervised morphological analysis by multimodal alignment, in *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pp. 207–216, Association for Computational Linguistics, Hong Kong, doi:10.3115/1075218.1075245.

François YVON (1997), Paradigmatic cascades: a linguistically sound model of pronunciation by analogy, in *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pp. 428–435, Association for Computational Linguistics, Madrid, Spain, doi:10.3115/976909.979672.

Xiang ZHANG, Junbo ZHAO, and Yann LECUN (2015), Character-level convolutional networks for text classification, in C. CORTES, N. D. LAWRENCE, D. D. LEE, M. SUGIYAMA, and R. GARNETT, editors, *Advances in Neural Information Processing Systems 28*, pp. 649–657, Curran Associates, Inc.