Deriving Morphological Analyzers from Example Inflections

Markus Forsberg,[1] **Mans Hulden**[2]
[1]Språkbanken, Department of Swedish, University of Gothenburg, Sweden
[2]Department of Linguistics, University of Colorado
markus.forsberg@gu.se, mans.hulden@colorado.edu

Abstract

This paper presents a semi-automatic method to derive morphological analyzers from a limited number of example inflections suitable for languages with alphabetic writing systems. The system we present learns the inflectional behavior of morphological paradigms from examples and converts the learned paradigms into a finite-state transducer that is able to map inflected forms of previously unseen words into lemmas and corresponding morphosyntactic descriptions. We evaluate the system when provided with inflection tables for several languages collected from the Wiktionary.

Keywords: learning morphology, paradigm induction, finite-state morphology

1. Introduction

Morphological analysis tools that provide detailed morphosyntactic descriptions (MSDs) and lemmatization of arbitrary word forms in some language are widely held to be fundamental for good performance of many higher-level NLP applications (Tseng et al., 2005; Spoustová et al., 2007; Avramidis and Koehn, 2008; Zeman, 2008; Hulden and Francom, 2012), particularly for languages with rich inflectional and derivational morphology. Hand-built systems, often modeled as finite-state transducers, offer very reliable morphological parses, but are time-consuming to create, and require significant linguistic expertise from the developers (Maxwell, 2015). In many cases, however, finding collated example inflections on a large scale for some language through resources such as the Wiktionary or simply by consulting a speaker of the language is far less laborious than the elaborate linguistic modeling required to produce a robust morphological analyzer.

In this paper we address this by describing a tool for automatic generation of finite-state morphological analyzers from collections of example word forms together with their MSDs. These morphological analyzers are constructed in the classical finite-state paradigm (Beesley and Karttunen, 2003), are non-probabilistic, and are designed to be high-recall, and hence to return all linguistically plausible analyses and lemmas, much like a hand-built morphological analyzer would which is extended with a 'guesser' module.

Our tool takes as input a set of words annotated with lemma and MSD, grouped into inflection tables, and produces as output a morphological analyzer using the Xerox regular expression formalism (Karttunen et al., 1996), which we compile into a transducer with the open-source finite-state toolkit *foma* (Hulden, 2009).[1]

2. Background

In this paper, we work with the idea that inflections and derivations of related word forms can be formally expressed as functions. Such a view has been commonly seen as an alternative to the finite-state morphology approach where functions that model inflectional behavior are built by hand (Forsberg and Ranta, 2004; Forsberg et al., 2006; Détrez and Ranta, 2012). As a starting point to the current work, we assume the approach of Ahlberg et al. (2014) and Ahlberg et al. (2015), which provide a mechanism to learn a specific type of function automatically from labeled data given to the algorithm in the form of inflected word forms grouped into inflection tables. In that work, different inflections and derived forms of a lemma are generalized into so-called 'abstract paradigms'.

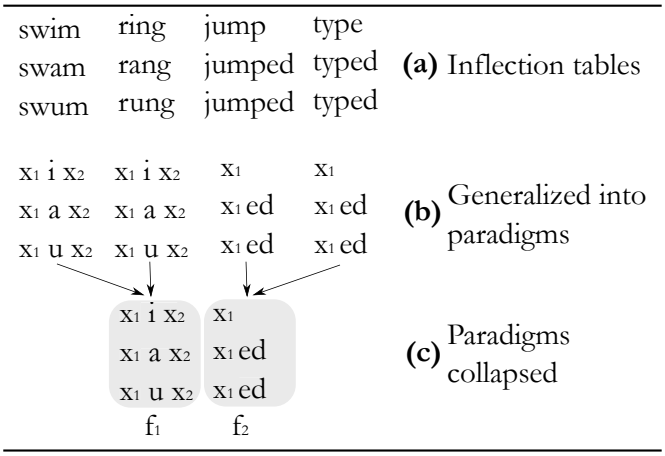

Figure 1: Illustration of generalizing inflection tables into abstract paradigms: (a) a number of inflection tables are given; (b) the aligned longest common subsequence is extracted; (c) resulting identical paradigms are merged. If the resulting paradigm is interpreted as a function, f_1*(shr, nk)* produces **shrink, shrank, shrunk**.

These paradigm functions essentially generalize concrete manifestations of word inflections for specific lemmas, allowing those inflections to be carried out for previously unseen words. This generalization is done by extracting the Longest Common Subsequence (LCS) for all related word forms and then declaring that the graphemes or phonemes that participate in the LCS are variables of the resulting function (Hulden, 2014). The new representation can be interpreted as a function which generates specific new inflection tables, given a specific set of variable values. We

[1]Our code is freely available together with the training/test setup employed in this paper at github.com/marfors/paradigmextract

refer the reader to Figure 1, which illustrates this process where paradigm functions are built from the extracted LCS from a number of inflected forms.

In general, the algorithm often produces much fewer unique functions compared with the number of inflection tables given as input. One of the advantages of the model is that it produces a human-readable structured output which can be inspected and also used as a starting point for other learning procedures, such as the production of finite-state morphological analyzers, which we describe below.

2.1. Variables in paradigms

The variables that participate in an abstract paradigm in essence capture all possible paradigmatic variation where substituting one set of variables in a paradigm function for another set of variables generates the inflectional pattern for a different word. For example, the paradigm function induced from **avenir** in Table 1 implicitly states that the original variables of the paradigm function were $x_1 =$ **av** and $x_2 =$ **n**. Substituting x_1 with e.g. **conv** yields another specific inflection table, the one for **convenir**. That is f(**conv**, **n**) maps to the inflection table:**conviniendo**, **convenido**, etc. Even though these variables can in principle be instantiated with arbitrary strings, morphophonological restrictions on their shape will inevitably come into play as variable parts alternate with fixed parts in the abstract paradigm. As we collect more data which are generalized into paradigms we also acquire more evidence about the nature of these restrictions. For example, Figure 2 shows some different x_1 and x_2 which have been seen as several concrete inflection patterns were all coalesced into one abstract paradigm.

3. Constraining variables

As we collect and generalize inflection tables we can examine the different instantiations of the variables $x_1, \ldots, x_n$. Not unsurprisingly, the content of these variables will not differ arbitrarily within a paradigm. Rather, most variables are constrained in that they reflect morphophonological phenomena and orthographic conventions. After collecting multiple inflection tables that yield many identical paradigms, we can reexamine the variables in a paradigm to produce more constrained definitions of the paradigms in question.

This can be illustrated through Figure 2, which shows the implicit mechanism by which to derive the lemma (the infinitive form) from the present participle and the first person singular present forms in two different learned paradigms in Spanish. Both paradigms contain two variables x_1 and x_2. In the first paradigm, called **avenir** we can see that x_1 always ends in the letter **v**, and that x_2 is always the string **n**.[2] In the second paradigm (**negar**), there is no clear pattern regarding the shape of x_1. However, x_2 in all 14 inflection tables that produced the paradigm, is always the string **eg**.

3.1. Estimating probabilities of new variable instantiations

This observation—that the variable parts of paradigms do not change arbitrarily—allows for the establishment of

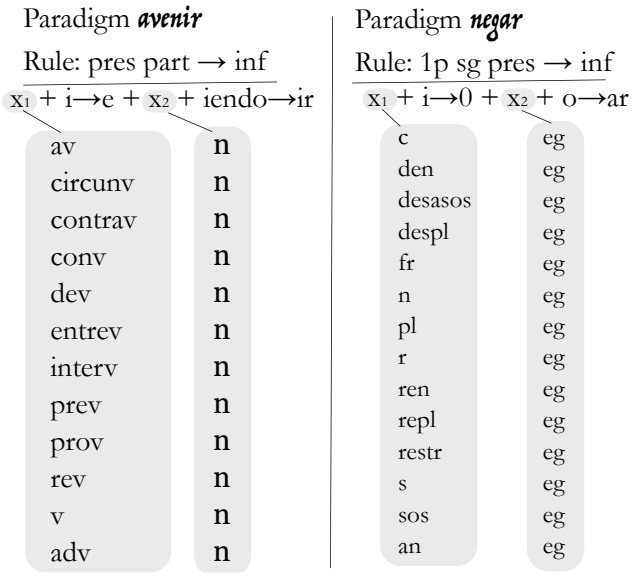

Figure 2: Abstract paradigms implicitly provide a mapping from any inflected form to any other form, a lemma for example. In the example we see two Spanish verbs from different paradigms together with a rule for mapping from the participle to the infinitive (left) and the first person singular present form to the infinitive (right). The different variable instantiations x_1 and x_2 seen in the training data are also shown.

constraints on their graphemic shape. As is seen intuitively from the contents of Figure 2, the x_2 variables seem to be entirely determined, while the **avenir** paradigm's x_1 is subject to variation except for the final letter, which is always **v**.

We can formalize a simple probability measure which effectively quantifies our belief in seeing novel instantiations of a variable in the future, apart from those already witnessed. We do so as follows: first, we assume, having witnessed t types of instantiations for a variable x_i, that there are in fact $t+1$ types, and that we have simply not yet seen the evidence for the $t+1$th type. We also assume that all types are drawn from a uniform distribution. Under these assumptions, the likelihood of witnessing the data where the class is not closed, but a $t+1$th member never happened to be witnessed, becomes

$$p_{\text{unseen}} = (1 - \frac{1}{t+1})^n \quad (1)$$

where n is the number of tokens witnessed for the variable. For example, the probability of the x_2 variable in **avenir** in Figure 2 becomes $(1 - 1/2)^{12} \approx 0.0002$.[3] This provides us with a parameter that can be used to refine how much evidence we require to declare the class of values that the variable can assume closed. We shall henceforth, unless otherwise stated, assume that if $p_{\text{unseen}} \leq 0.05$, the class is closed.

[2]The name of the paradigm is arbitrarily chosen among all the verbs that belong to it.

[3]The question of estimating the probability of a previously unseen type is addressed in multiple ways in the literature, the most-well known of which is the popular Good-Turing estimator (Good, 1953). Other less known, specifically linguistics discussions on the matter include Ogino (1999) and Kageura and Sekine (1999).

Inflection table	Paradigm	MSD	Inflection table	Paradigm	MSD
avenir	x_1+e+x_2+ir	infinitive	**neg**ar	x_1+x_2+ar	infinitive
aviniendo	x_1+i+x_2+iendo	pres part	**neg**ando	x_1+x_2+ando	pres part
avenido	x_1+e+x_2+ido	past part	**neg**ado	x_1+x_2+ado	past part
avengo	x_1+e+x_2+go	1sg pres ind	**nieg**o	x_1+i+x_2+o	1sg pres ind
avienes	x_1+ie+x_2+es	2sg pres ind	**nieg**as	x_1+i+x_2+as	2sg pres ind

Table 1: An illustration of generalizing two (partial) Spanish verb inflection tables into paradigms. The segments that are part of the longest common subsequence, which are cast as variables in the generalization, is shown in boldface in the inflection tables.

3.2. Expressing constraints through regular expressions

The above measure allows to constrain those variables where the entire variable assumes a limited number of strings, as in x_2, which is always **n** in the paradigm for **avenir** Table 2. However, x_1, for example, in the same paradigm appears to not be a closed class in this sense, but does appear to have limited variability in its last letter. To model this intuition, we can use the same scoring method as above, and extend it to investigate the seen prefixes and suffixes of all lengths for some variable x_i. We do as follows. First we examine all strings seen as instantiations of x_i. If $p_{\text{unseen}} \leq 0.05$, we declare the class closed. If the class cannot be assumed to be closed, we find the longest prefix and suffix which can be assumed to be closed by the same measure. After doing this, we can construct a regular expression that models the possible variation seen for a variable, which is:

1. $(w_1 \cup w_2 \cup \ldots \cup w_n)$ if the class is closed, where the w_is are the complete strings seen as instantiations.
2. $(p_1 \cup \ldots \cup p_n)\Sigma^* \cap \Sigma^*(s_1 \cup \ldots \cup s_n)$, if both prefixes and suffixes can be constrained; here the p_is correspond to the prefixes of the maximal length that can be assumed to be drawn from a closed class, and the s_is the suffixes.
3. $(p_1 \cup \ldots \cup p_n)\Sigma^*$ if only prefixes can be constrained.
4. $\Sigma^*(s_1 \cup \ldots \cup s_n)$ if only suffixes can be constrained.
5. Σ^+ otherwise.

Here, we have collected all the symbols seen in all paradigms into the alphabet, Σ. For example, variables in the **avenir** paradigm in Figure 2 generalize to the following regular expressions:

$$x_1 = (\Sigma^* v) \;\; x_2 = n \tag{2}$$

i.e. x_1 must end in **v** and x_2 must always be the single letter **n**.[4]

These constraints allow us to capture many linguistic generalizations within a paradigm, and also to directly construct finite-state transducers that map word forms to their analyses in a constrained way.

[4]Our tool also provides different export formats for these generalizations, including Python-compatible regular expressions, and foma-compatible ones.

4. Deriving morphological analyzers

The above generalization of related word forms into paradigms and the subsequent restriction of the variable parts of each paradigm allows to directly construct individual transducers that perform the specific mapping of word form to its lemma and MSD. To achieve this, we construct a regular expression for each form in a paradigm that alternately repeats the variable parts in a paradigm form and maps the other parts to their corresponding lemma form.

For example, to map the **aviniendo** form (in Table 1) to its lemma form **avenir**, we note that we have generalized **aviniendo** as x_1 + **i** + x_2 + **iendo**, while the lemma form is generalized as x_1 + **e** + x_2 + **ir**. Taking into account the fact that we have witnessed many different words that follow the same pattern, and so further constrained x_1 to the regular expression $(\Sigma^* v)$ and x_2 to the regular expression n, we can construct a regular expression for this entire mapping:

$$(\Sigma^* v)\ (i\!:\!e)\ n\ (iendo : ir[\text{type=participle}]) \tag{3}$$

The resulting single transducer of this is shown in Figure 3 (top).

Each word form of each paradigm is thus converted into a single transducer that analyzes a single type of inflection. All these transducers can be combined into one monolithic transducer by simply calculating a union of them all:

$$f_1 \cup f_2 \cup \ldots \cup f_1 \cup \ldots \cup f_m \tag{4}$$

5. Prioritizing analyses

The above directly allows us to construct a morphological analyzer that is both guaranteed to handle all inflections provided as training data and also generalizes this to unseen word forms in a constrained way through inferring morphophonological restrictions on the variables in a paradigm. However, for practical use, such an analyzer may return multiple analyses for any word form given to it if some paradigms are sufficiently unconstrained. This is often undesirable for analyzing those word forms already seen in the training data (since we know what paradigm they belong to). Additionally, it may also be the case that we have overconstrained some variable and would thus be missing analyses for some unseen word forms.

To address both these issues, we generate three separate analyzers as follows:

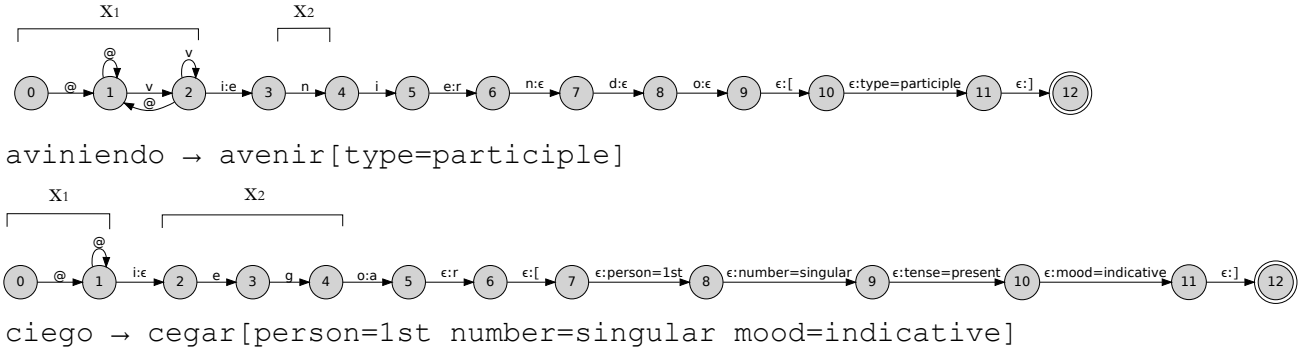

Figure 3: Examples of two single generalized word forms mapped to lemmas followed by morphosyntactic description. The parts that correspond to constraints of the variables x_1 and x_2 are marked. Transitions marked @ are identity transduction 'elsewhere' cases, matching any symbol not explicitly mentioned in the state.

	Analysis: **peleaste**	
O	**pelear**	**[pers=2 number=sg tense=past mood=ind]**
C	peleastar	[pers=1 num=sg tense=pres mood=subj]
	peleastar	[pers=3 num=sg tense=pres mood=subj]
	peleastir	[pers=3 num=sg tense=pres mood=ind]
	pelear	**[pers=2 num=sg tense=past mood=ind]**
U	peleaster	[pers=3 num=sg tense=pres mood=ind]
	peleastar	[pers=1 num=sg tense=pres mood=subj]
	peleastar	[pers=3 num=sg tense=pres mood=subj]
	peleastir	[pers=3 num=sg tense=pres mood=ind]
	pelear	**[pers=2 num=sg tense=past mood=ind]**
	pelastir	[pers=3 num=sg tense=pres mood=ind]
	pleastir	[pers=3 num=sg tense=pres mood=ind]
	Analysis: **aceleran**	
O	???	???
C	**acelerar**	**[pers=3 num=pl tense=pres mood=ind]**
	acelerir	[pers=3 num=pl tense=pres mood=subj]
U	**acelerar**	**[pers=3 num=pl tense=pres mood=ind]**
	aceler	[pers=3 num=pl tense=imp-ra mood=subj]
	acelerir	[pers=3 num=pl tense=pres mood=subj]
	acelerer	[pers=3 num=pl tense=pres mood=subj]
	acelrir	[pers=3 num=pl tense=pres mood=subj]
	acelir	[pers=3 num=pl tense=imp-ra mood=subj]
	aclerir	[pers=3 num=pl tense=pres mood=subj]

Table 2: Illustration of the behavior of the three subgrammars learned on two Spanish words; **Original** = **O**, **Constrained** = **C**, **Unconstrained** = **U**. The first word to be parsed, **peleaste** 'quarrel', is seen in the training data and hence gets a parse from **O**, while the second word **aceleran** 'accelerate' is not, and only gets parses from **C** and **U** with increasing tolerance of variable instantiation.

- **Original**: this is an analyzer where no variables are generalized; any x_i must be exactly one of those seen in the training data.
- **Constrained**: this is an analyzer where variables are constrained as described above in section 3.
- **Unconstrained**: this is an analyzer where all variables are completely unconstrained, i.e. match the regular expression Σ^+.

These analyzers can be combined into one large transducer by, e.g., an operation commonly called *priority union* (Kaplan, 1987):

$$\textbf{Original} \cup_P \textbf{Constrained} \cup_P \textbf{Unconstrained} \quad (5)$$

This in effect leads to an analyzer that can be thought of as first consulting **Original**, and that failing to produce an analysis, consults **Constrained**, and if that also fails, consults **Unconstrained**.[5] This results in a tri-level analysis where the most robust analyses have priority over those which can be produced by more lax generalization. Table 2 shows the parses given with two Spanish words using the differently constrained transducers when trained on Wiktionary training data to illustrate the increasing number of accepted parses with decreasing constraints.

6. Evaluation

To evaluate the system we used the data set published by Durrett and DeNero (2013) (D&DN13). This includes fully inflected forms for thousands of lemmas in three languages: German (nouns and verbs), Spanish (verbs), and Finnish (nouns+adjectives and verbs). These forms are organized into inflection tables. We used the same train/test splits as in the source, and set aside all the word forms from 200 inflection tables for testing. The task then consisted of providing analyses for each unseen form. We test on each part-of-speech separately, and also provide an evaluation on a combined test set where the part-of-speech is not known. The evaluation setup is the one described above where we have a three-part priority system—and a union of these systems in the combined test case—that returns preferably only the most constrained analyses first, and resorting to the more lax variant if the more constrained system fails to produce output. Since the sets of tables of train+dev and test are disjoint, the contribution of **Original** is ranging from modest to non-existent.

We ran our tool on the word forms and corresponding MSDs in the training data and generated a finite-state transducer as described previously. We then used the transducers

[5] The analyzers can of course be stored as separate transducers without combining them, and the 'priority union' strategy simulated at run-time.

Language		L-recall	L+M-recall	L/W	L+M/W
German	nouns	95.30%	95.06%	2.08	9.52
	verbs	91.18%	92.44%	4.16	9.57
	nouns+verbs	92.11%	93.04%	4.91	14.10
Spanish	verbs	98.06%	97.98%	1.93	2.20
Finnish	nounadj	88.69%	88.48%	4.10	5.30
	verbs	94.52%	94.47%	3.77	4.60
	nounadj+verbs	92.63%	92.43%	12.56	16.40

Table 3: The result of the evaluation, where we report separately on the recall of just the lemma (L-recall), and the recall of the lemma and corresponding MSD (L+M-recall). Also shown are the average number of unique lemmas returned per word form to be analyzed (L/W), and the average number of lemmas and MSDs returned (L+M/W).

Language		Tables	Paradigms
German	nouns	2564	70
	verbs	1827	139
	nouns+verbs	4391	209
Spanish	verbs	3855	96
Finnish	nounadj	6200	259
	verbs	7049	276
	nounadj+verbs	13249	535

Table 4: The D&DN13 train+dev set. **Paradigms** is the corresponding number of induced abstract paradigms.

Language		Tables	Unique word forms	Ambiguity
German	nouns	200	553	2.89
	verbs	200	2324	2.32
	nouns+verbs	400	2877	2.43
Spanish	verbs	200	10003	1.14
Finnish	nounadj	200	5198	1.08
	verbs	200	10466	1.03
	nounadj+verbs	400	15664	1.05

Table 5: The D&DN13 test set. **Ambiguity** is the average number of lemma-MSD pairs per unique word form.

to map unseen word forms in the test data to analyses. Table 4 shows the number of inflection tables used as training data, together with the number of resulting unique paradigms (or paradigm functions) the longest-common-subsequence extraction produces. Table 5 shows the number of tables and unique word forms in the test set. Also, we provide an inherent 'ambiguity' measure of the test data which shows the average number of different morphosyntactic descriptions that correspond to one word form in the test data (e.g. a Spanish verb **tenga** is inherently ambiguous as it can either be the first person singular present subjunctive of **tener** 'to have' or the third person singular present subjunctive). Because of this significant inherent ambiguity, we focus on the recall of the system, rather than precision, reporting the ambiguity separately.[6]

[6] An ideal evaluation metric would be to mimic the task where a human is asked to enumerate all plausible lemmas and their MSDs for an unseen/previously unknown word, where only the truly impossible interpretations should be ruled out. In such a way, one could provide both precision and recall figures for the task. Unfortunately, no such data set exists and it seems one would be rather challenging to produce because of the difficulty of exhaustively enumerating all morphophonologically plausible analyses and lemmatizations.

6.1. Results

Table 3 shows the main results of the evaluation.

In some cases, L-recall is lower than the corresponding L+M-recall, which might seem counter-intuitive at first. To understand why this could happen, consider these two cases: $w_1 \rightarrow \{l_1 : msd_1, l_1 : msd_2\}$ and $w_2 \rightarrow \{l_2 : msd_3\}$. If we get w_1 completely right and w_2 completely wrong, that would give us an L-recall of 50% and an L+M-recall of 66.6%.

It might be tempting to estimate precision by comparing Ambiguity with L+M/W, but this should be done with care. Even if we disregard the requirement that only impossible analyses should be ruled out in this evaluation, the test data does not give us a complete picture; to illustrate this with an English example, it might be the case that **call (noun)** is in the test set, but not **call (verb)**, which would give us an Ambiguity score of 1 but a (strict) L+M/W score of 4, i.e., a conclusion that the precision is 25% would be incorrect.

7. Conclusion and future work

The tool we have described allows for a supervised method to produce morphological analyzers and guessers in the finite-state paradigm from labeled word forms. The method offers an avenue to quickly produce high-recall morphological analysis from limited amounts of labeled data with no linguistic development necessary.

The tool can be used as is or for semi-supervised lexicon expansion if access to raw text corpora is available. Some immediate further extensions appear useful. For example, weighted finite-state transducers could be used to impose language models—also learned from the data—for the variable parts in a paradigm instead of the constraint method used here. Such a tool could automatically rank the different lemma/MSD candidates that the system currently provides as analyses and could also combine seamlessly with higher-level statistical analyses such as part-of-speech tagging and syntactic parsing.

Acknowledgements

This work has been funded by the Swedish Research Council under grant number 2012-5738, *Towards a knowledge-based culturomics* and the University of Gothenburg through its support of the Centre for Language Technology and its support of Språkbanken.

8. Bibliographical References

Ahlberg, M., Forsberg, M., and Hulden, M. (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*, pages 569–578, Gothenburg, Sweden. Association for Computational Linguistics.

Ahlberg, M., Forsberg, M., and Hulden, M. (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*, pages 1024–1029, Denver, Colorado, May–June. Association for Computational Linguistics.

Avramidis, E. and Koehn, P. (2008). Enriching morphologically poor languages for statistical machine translation. *NAACL-HLT 2008*, pages 763–770.

Beesley, K. R. and Karttunen, L. (2003). *Finite State Morphology*. CSLI Publications, Stanford, CA.

Détrez, G. and Ranta, A. (2012). Smart paradigms and the predictability and complexity of inflectional morphology. In *Proceedings of the 13th EACL*, pages 645–653. Association for Computational Linguistics.

Durrett, G. and DeNero, J. (2013). Supervised learning of complete morphological paradigms. In *Proceedings of NAACL-HLT*, pages 1185–1195.

Forsberg, M. and Ranta, A. (2004). Functional morphology. *ACM SIGPLAN Notices*, 39(9):213–223.

Forsberg, M., Hammarström, H., and Ranta, A. (2006). Morphological lexicon extraction from raw text data. In *Advances in Natural Language Processing*, pages 488–499. Springer.

Good, I. J. (1953). The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3-4):237–264.

Hulden, M. and Francom, J. (2012). Boosting statistical tagger accuracy with simple rule-based grammars. In *Proceedings of the Language Resources and Evaluation Conference (LREC 2012)*, pages 2114–2117.

Hulden, M. (2009). Foma: a finite-state compiler and library. In *Proceedings of the 12th Conference of the European Chapter of the European Chapter of the Association for Computational Linguistics: Demonstrations Session*, pages 29–32, Athens, Greece. Association for Computational Linguistics.

Hulden, M. (2014). Generalizing inflection tables into paradigms with finite state operations. In *Proceedings of the 2014 Joint Meeting of SIGMORPHON and SIGFSM*, pages 29–36. Association for Computational Linguistics.

Kageura, K. and Sekine, S. (1999). A note on Ogino's "method to estimate probability of new appearance". *Journal of Mathematical Linguistics*, 22(3).

Kaplan, R. M. (1987). Three seductions of computational psycholinguistics. In P. Whitelock, et al., editors, *Linguistic Theory and Computer Applications*, London. Academic Press.

Karttunen, L., Chanod, J.-P., Grefenstette, G., and Schiller, A. (1996). Regular expressions for language engineering. *Natural Language Engineering*, 2(4):305–328.

Maxwell, M. (2015). Grammar debugging. In *Systems and Frameworks for Computational Morphology*, pages 166–183. Springer.

Ogino, T. (1999). How many examples are required in language research—a proposal of a method to estimate probability of new appearance. *Mathematical Linguistics*, 22(1):11–17.

Spoustová, D., Hajič, J., Votrubec, J., Krbec, P., and Květoň, P. (2007). The best of two worlds: Cooperation of statistical and rule-based taggers for Czech. In *Proceedings of the Workshop on Balto-Slavonic Natural Language Processing*, pages 67–74.

Tseng, H., Jurafsky, D., and Manning, C. (2005). Morphological features help POS tagging of unknown words across language varieties. In *Proceedings of the fourth SIGHAN workshop on Chinese language processing*, pages 32–39.

Zeman, D. (2008). Reusable tagset conversion using tagset drivers. In *Proceedings of the Language Resources and Evaluation Conference (LREC 2008)*.