Linguist vs. Machine: Rapid Development of Finite-State Morphological Grammars

Sarah Beemer and **Zak Boston** and **April Bukoski** and **Daniel Chen** and
Princess Dickens and **Andrew Gerlach** and **Torin Hopkins** and
Parth Anand Jawale and **Chris Koski** and **Akanksha Malhotra** and **Piyush Mishra** and
Saliha Muradoğlu[☼] and **Lan Sang** and **Tyler Short** and **Sagarika Shreevastava** and
Elizabeth Spaulding and **Tetsumichi Umada** and **Beilei Xiang** and **Changbing Yang** and
Mans Hulden
University of Colorado
The Australian National University[☼]

Abstract

Sequence-to-sequence models have proven to be highly successful in learning morphological inflection from examples as the series of SIGMORPHON/CoNLL shared tasks have shown. It is usually assumed, however, that a linguist working with inflectional examples could in principle develop a gold standard-level morphological analyzer and generator that would surpass a trained neural network model in accuracy of predictions, but that it may require significant amounts of human labor. In this paper, we discuss an experiment where a group of people with some linguistic training develop 25+ grammars as part of the shared task and weigh the cost/benefit ratio of developing grammars by hand. We also present tools that can help linguists triage difficult complex morphophonological phenomena within a language and hypothesize inflectional class membership. We conclude that a significant development effort by trained linguists to analyze and model morphophonological patterns are required in order to surpass the accuracy of neural models.

1 Introduction

Hand-written grammars for modeling derivational and inflectional morphology have long been seen as the gold standard for incorporating a word inflection aware component into NLP systems. However, the recent successes of sequence-to-sequence (seq2seq) models in learning morphological patterns, as seen in multiple shared tasks that address the topic (Cotterell et al., 2016, 2017, 2018; McCarthy et al., 2019), have raised the question whether there is any advantage in developing handwritten grammars for performance reasons. This question has special relevance with regard to low-resource languages when there is a desire to quickly develop fundamental NLP resources such as a morphological analyzer and generator with minimal resource expenditure (Maxwell and Hughes, 2006).

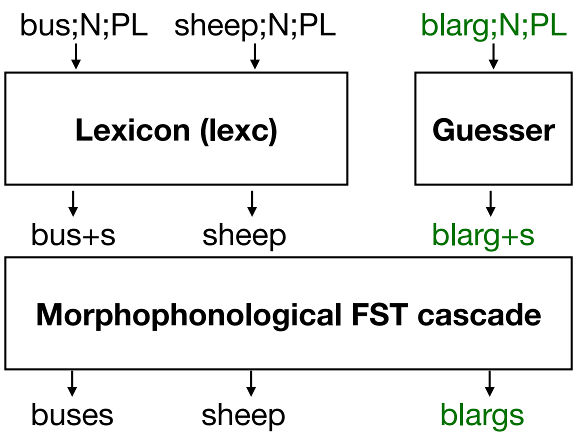

Figure 1: Basic FST grammar design used in this project which combines a lexicon-based model with a guesser to handle unseen lemmas.

It is clear that there is a need for hand-written morphological grammars, even if neural network models approach the performance of carefully hand-crafted morphologies. Normative and prescriptive language models, such as those needed by language academies in many countries—e.g. RAE in Spain, Académie Française in France, or the Council of the Cherokee Nation in the U.S.—would need to rely on explicitly designed models for providing guidance in word inflection, spelling rules, and orthography if they were to be implemented computationally. Currently, neural models trained on examples provide no verifiable guarantees that certain prescriptive phenomena have been learned by a trained model and can be reliably used.

In this paper[1] we document an experiment where a number of morphological grammars were handwritten by a group of 19 students enrolled in the class "LING 7565—Computational Phonology & Morphology" at the University of Colorado, each

[1]All tools and grammars developed are available on https://github.com/mhulden/7565tools.

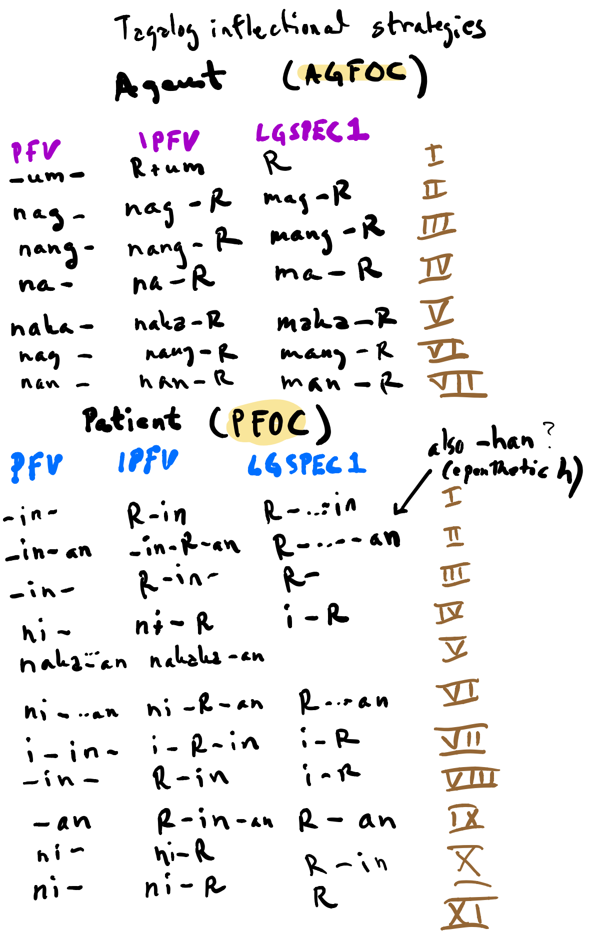

Figure 2: Old-school pencil-and-paper Linguistics: hypothesizing the possible inflectional patterns for Tagalog Actor Focus and Object Focus verb forms (from project notes). The symbol R represents reduplication of the first CV(V) in the stem.

student having training in either computer science or linguistics, and some previous training in writing finite-state morphological grammars. The languages were chosen from the 2020 SIGMORPHON shared task 0 (Vylomova et al., 2020), and the grammars were designed so as to be able to inflect unseen forms. The design was also such that the grammars were able to function as "guessers" and inflect lexemes never seen in the training data.

2 Finite-State Grammars

Finite-state Transducer (FST) solutions have long been the foremost paradigm in which to develop linguistically informed large-scale morphological grammars (Koskenniemi, 1983; Beesley and Karttunen, 2003; Hulden, 2009). The availability of a variety of tools (Hulden (2009); Riley et al. (2009); Beesley (2012) inter alia) has also supported this mode of development, and by now hundreds of languages have grammars developed by linguists in this paradigm.

The usual approach to developing morphological analyzers is to model the mapping from a lemma (citation form) and a morphosyntactic description (MSD) into an inflected form (target form) as a two-step process. The first step maps the lemma+MSD into an intermediate form that represents a combination of canonical morpheme representations, while the second step employs a cascade of transducers which handle morphophonological alternations. It is customary to handle inflectional classes by explicitly dividing lemmas into groups in the first step so that correct morphemes are chosen for each lemma. Analyzers built in such a way generally are not capable of inflecting lemmas that are not explicitly encoded in a lexicon. However, it is common to integrate an additional "guesser" component that can handle any valid lemma in a language, and pass it through the relevant morphophonological component only (Beesley and Karttunen, 2003). Basic finite-state calculus is then used to construct a single FST that "overrides" outputs from the guesser whenever a known lexeme is inflected, so conflicting outputs are avoided. The basic design is illustrated in Figure 1.

3 Approach

All of the grammars were built with the *foma* finite-state tool (Hulden, 2009). Before grammar writing commenced, the participants were urged to spend roughly 1 hour in groups of 3 to quickly analyze all the languages in the development and surprise groups as follows:

- Triage: the training sets for all languages in the shared task were rapidly analyzed for difficulty, and possible complex inflectional classes. Following this, a selection of languages were chosen by the participants to model. This was done once for the development languages, and through an additional round of triage for the surprise languages.

- Each language was scored for difficulty based on familiarity with the writing system, paradigm size, complexity, and the apparent number of inflectional classes; naturally the actual number was not known, and this represented an educated guess. Participants were asked to informally rate the difficulty of a language on a 1(easy)–5(very difficult) before choosing languages to work on. The participants were not explicitly instructed to pick an easy language, but rather, to choose one that would provide an interesting experience and would be feasible to complete.[2]

- Computational tools (discussed below) were used to reconstruct the partial paradigms given in the training data, to extract the alphabets used in the languages, to canonicalize the UniMorph tag order (Kirov et al., 2018) used in the data, and to provide a rapid development environment that could give instant feedback on accuracy on the training and dev sets after compilation of FSTs.

- A template grammar was used as a starting point; it provided both the possibility of developing a morphophonology-only grammar, or a grammar where all lemmas needed to be divided into inflectional classes.

Through the above process, a number of languages were selected as the primary targets, and development was launched for some 40 languages in total—roughly 20 for the development languages and a similar number for the surprise languages, as they were published. In the end, the output of 25 languages was submitted to the shared task. The criterion for actually submitting a language was that the grammar was mature enough, judged by examining whether accuracy on the development set was within 5% of the neural baseline models (Wu et al., 2020) provided by the organizers.

4 Tools

As mentioned above, a number of tools for the support of rapid grammar writing were also developed. These included the tools to reconstruct the partial inflection tables from the data and various analysis tools for accuracy and error reporting.

Apart from that, a separate tool for inflection table clustering and a non-neural tool for hypothesizing forms for missing slots in paradigms were also developed. This latter tools' output was also submitted as a second system (**CU-7565-02**) to the shared task for nearly all languages. These two tools were more involved and are discussed in detail below.

4.1 Inflection Table Clustering

Crucial in the development of a grammar from raw, partial inflection table data is the ability to hypothesize if lexemes fall into different inflectional classes quickly, and if so, how. This is non-trivial to determine, especially with large amounts of lexemes represented in the various data sets. It is also essential to disentangle phonological regularity from inflectional classes which may be significant red herrings in the analysis of a language. For example, while **cat** in English pluralizes as **cats**, **bus** pluralizes as **buses**—by an epenthetic **e** inserted between sibilants. A naive analysis would postulate that the two lexemes behave differently and place them in separate inflectional classes, although a properly designed phonological component could avoid this unnecessary complexity in the morphological component.

4.1.1 Lexeme similarity measure

To facilitate providing a linguist with a quick overview, we developed a model to perform rapid hierarchical clustering of all lexemes in a language's data set. To this end, we developed a metric for lexeme similarity with respect to inflectional behavior. This metric is calculated by a two-step process. First, all pairs of word forms for a lexeme (within a paradigm) are aligned using an out-of-the-box Monte Carlo aligner (Cotterell et al., 2016) written by the last author. This is shown in figure 3 (a). Following this alignment procedure, we automatically produce a crude approximation of the string transformation implied by the alignment as a regular expression, which is then compiled into an FST.

In the conversion process, matching input sequences in the alignment are modeled by `?+` (repeat one or more symbols[3]) and non-matching symbols are replaced by the symbol-pair found in the alignment: i:o. For example, the aligned pair **runs** $\leftrightarrow$ **ran** in Figure 3 (b) is converted into the regular expression

```
?+   u:a   ?+   s:0
```
(1)

which can be compiled into a transducer in Figure 3 (c). This transducer generalizes over the matched elements in the input-output pair and can be applied to other third-person present forms, such as **outruns** to produce **outran**. Obviously, this example transformation only applies to this particular

[2]On average, the surprise languages were deemed considerably more difficult, largely because of paradigm size.

[3]We use foma regular expression notation.

inflectional class and will give incorrect transformations such as **pulls** → **pall** for words that do not have the same inflectional behavior. The purpose of calculating all-known-pairs mappings for each lexeme is to provide a *similarity measure* between lexemes. In particular, we use the following measure for two lexemes l_1 and l_2, which compares the overlap of all transformation rules found between the forms in l_1 with the transformation rules in l_2:

$$\mathrm{sim}(l_1, l_2) = \frac{2 \times \#\mathrm{shared}(l_1, l_2)}{\#\mathrm{shared}(l_1, l_1) + \#\mathrm{shared}(l_2, l_2)} \tag{2}$$

Here, $\#\mathrm{shared}(l_1, l_2)$ is the simple count reflecting how many of the slot-to-slot transformation rules in l_1 are identical for l_2.

We subsequently convert this similarity score into a distance for the purposes of clustering:

$$\mathrm{distance}(l_1, l_2) = 1 - \mathrm{sim}(l_1, l_2) \tag{3}$$

Note that the denominator in the similarity calculation in effect expresses the maximum possible similarity scores for l_1 and l_2 by calculating the similarity with themselves, resulting in a range of $[0, 1]$ for the overall similarity and distance measures. Since many given paradigms contain missing forms and are therefore missing pair-transformations as well, this maximum score will vary from lexeme to lexeme.

With this similarity in hand between all lexemes, we can perform a (single-link) agglomerative hierarchical clustering of all lexemes in the training data of a language.

Example results of the clustering are shown in Figure 4 for Ingrian (the full training set which contained partial inflectional tables for 50 lexemes), and English (a small subset). Included in the Ingrian clustering are our final linguist-hypothesized inflectional class numbers for each lexeme for comparison.

4.2 Inflection with transformation FSTs

As a byproduct of the clustering distance measure that uses slot-to-slot transformation FSTs, we can also address the shared task itself. Since the development and test sets largely contain unknown inflections from lexemes where *some* forms have been seen, we can make use of the learned transformation rules from other lexemes that target an unknown form asked for in the development or

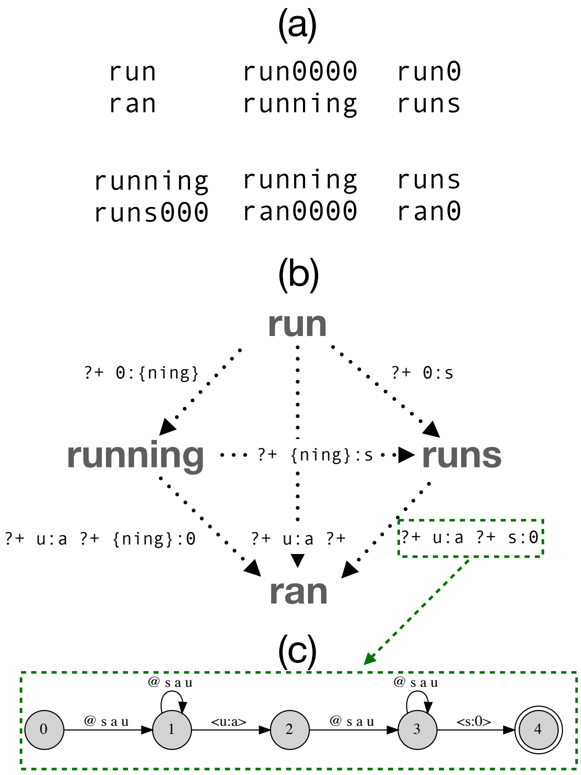

Figure 3: Generating transformation rules for each pairwise slot for a lexeme: (a) we perform alignment of all pairs, (b) a regular expression is issued to model the transformation which is compiled into an FST (c).

test sets. To this end, we collect all known *source* → *target* transformation rules from all other tables where the target form is the desired slot (MSD). We then apply all of these transformations, generating potentially hundreds of inflection candidates for the missing target slot of a lexeme. From among the candidates, we perform a majority vote. For all languages, we experimented with weighting the majority vote so that transformation rules that come from paradigms that share many transformation rules with the target lexeme's paradigm get a multiplier for the vote using the similarity measure in (2). This strategy produced slightly superior results throughout, as analyzed by performance on the development set, and was hence used in the final submission for our system **CU-7565-02**.

5 Results

The results for the hand-written grammars (**CU-7565-01**) and the non-neural paradigm completion model (**CU-7565-02**) are given in Table 1. We note that we were able to match or surpass the strongest neural participant in the task on 13 languages with the hand-written grammars. Several of these, how-

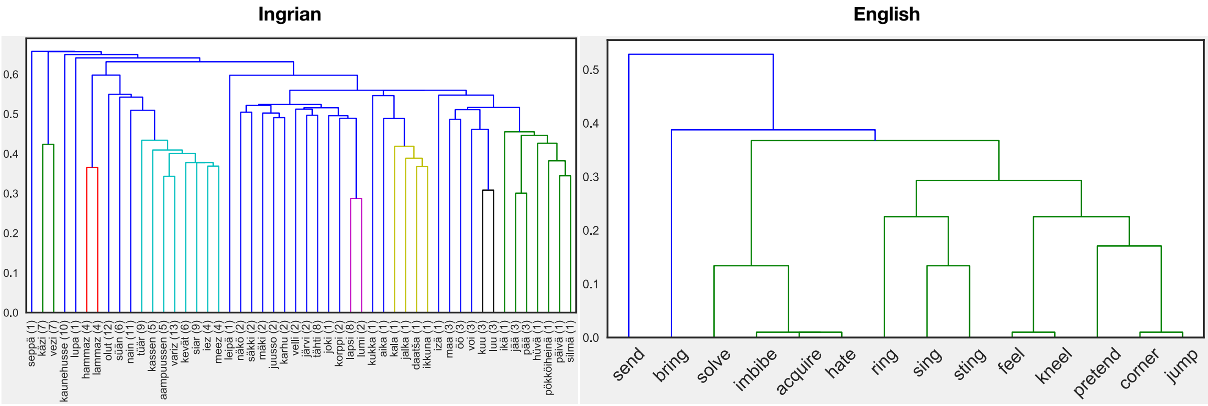

Figure 4: Hierarchical clustering of lexemes by apparent inflectional behavior based on string transformations between inflectional slots for Ingrian (left) and English (right). The numbers in parentheses in Ingrian refer to the Linguist-derived inflectional class number after developing a grammar. The Ingrian data is the output from the full training data while the English is a small selection of verbs to illustrate clustering behavior.

ever, were relatively "easy" languages and often did not contain any significant morphophonology at all. On two languages, Ingrian (izh) and Tagalog (tgl), we were able to significantly improve upon the other models participating in the task. These languages had a fairly large number of inflectional classes and very complex morphophonology. Ingrian features a large variety of consonant gradation patterns common in Uralic languages, and Tagalog features intricate reduplication patterns (see Figure 2).

We include results for train, dev, and test as we used tools to continuously evaluate our progress during development on the training set. It is worth noting that the linguist-driven development process does not seem to be prone to overfitting—accuracy for several languages on the test set was actually higher than on the training set.

The non-neural paradigm completion model (**CU-7565-02**), which was submitted for nearly all 90 languages performed reasonably well, and is to our knowledge the best-performing non-neural model available for morphological inflection. Never outperforming the strongest neural models; it nevertheless represents a strong improvement over the baseline non-neural model provided by the organizers. Additionally, it provides another tool to quickly see reasonable hypotheses for missing forms in inflection tables.

6 Discussion

6.1 Earlier work

To our knowledge, no extensive comparison between well-designed manual grammars and neural

Language	**trn**[1]	**dev**[1]	**tst**[1]	**tst**[2]
aka	100.0	100.0	**100.0**	89.8
ceb	85.2	86.2	86.5	84.7
crh	97.5	97.0	96.4	97.7
czn	79.0	76.0	72.5	76.1
dje	100.0	100.0	**100.0**	**100.0**
gaa	100.0	100.0	**100.0**	**100.0**
izh	93.4	91.1	**92.9**	77.2
kon	100.0	100.0	98.7	97.4
lin	100.0	100.0	**100.0**	**100.0**
mao	85.5	85.7	66.7	57.1
mlg	100.0	100.0	**100.0**	-
nya	100.0	100.0	**100.0**	**100.0**
ood	81.0	87.5	71.0	62.4
orm	99.6	100.0	**99.0**	93.6
ote	91.2	93.5	90.9	91.3
san	88.5	89.7	89.0	88.3
sna	100.0	100.0	**100.0**	99.3
sot	100.0	100.0	**100.0**	99.0
swa	100.0	100.0	**100.0**	**100.0**
syc	89.3	87.3	88.3	89.1
tgk	100.0	100.0	**93.8**	**93.8**
tgl	77.9	75.0	**77.8**	-
xty	81.1	80.0	81.7	70.3
zpv	84.3	77.9	78.9	81.1
zul	82.9	88.1	83.3	88.5

Table 1: Results for the train, dev, and test sets with our handwritten grammars ([1]) and our non-neural learner ([2]). The non-neural model also participated in additional languages not shown here. Languages with accuracies on par with or exceeding the best shared task participants are shown in boldface.

network models for morphology have been proposed. Pirinen (2019) reports on a small experiment that compares an earlier SIGMORPHON shared task winner's results to a Finnish hand-written morphological analyzer (Pirinen, 2015), with the seq2seq-based participant's model yielding higher precision than the rule-based FST analyzer. In another related experiment, Moeller et al. (2018) train neural seq2seq models from an existing hand-designed transducer acting as an oracle and note that the seq2seq model begins to converge to the FST with around 30,000 examples in a very complex language, Arapaho (arp).

The non-neural inflection model (**CU-7565-02**) builds upon paradigm generalization work by Forsberg and Hulden (2016), which in turn is an extension of Hulden et al. (2014) and Ahlberg et al. (2015). An earlier non-neural model for paradigm generalization is found in Dreyer and Eisner (2011).

6.2 Human Resources

We did not record the exact amounts of time spent on the project individually for each participant. However, we can estimate this based on previous years' class surveys in the same course (LING 7565—Computational Phonology and Morphology) as regards the number of hours per week students spend working on course projects. Each student on average in the course spends 6.6 hours per week; as the project ran for 5 weeks with 19 participants, we roughly estimate a total of 627 person-hours spent on the task of developing grammars. As reflected in the results, we considered 13–15 languages to have largely completed grammars, or very nearly completed. The remainder of the 25 languages submitted were known to require further work, but very little work to reach accuracies beyond or at the best-performing neural models for the task. These estimates do not include student training in morphology, finite-state machines, and grammar writing. Likewise, some languages with very large number of forms per lexeme—such as Erzya (myv) with 1,597 forms and Meadow Mari (mhr) with 1,597 forms—were deemed outside the realm of realistic analysis and linguist-driven grammar writing within a scope of 5 weeks that were allotted to the work.

6.3 Neural or Human?

Given the above estimates, we can provide a conservative estimate of at least 40 person-hours of work on average—not counting infrastructure development and strategizing—to develop a hand-written morphological analyzer and generator that is on par with a model learned by state-of-the-art neural approaches. There is large variance around this figure, however, as some very regular languages only required 30 minutes of work and a dozen-or-so lines of code to produce a model that captures all the morphology and morphophonology involved. Others required a much greater and more intense effort in analyzing the partial inflection tables given in the training data, classifying lemmas into inflectional classes and modeling morphophonological rules as FSTs. Additionally, we note that all the participants had already been trained in this kind of analysis and grammar writing, a factor that our estimate does not take into account.

6.4 Language Notes

In the course of the development of the grammars, we observed that many languages had a skewed selection of data, or inconsistencies that would not be fruitful to model in a hand-written grammar. This also meant that in such cases it was unlikely that the hand-written grammar would ever attain the performance of a neural model, which can better handle the inconsistencies described below. We hope to be able to clean up the data as the test data is released to re-evaluate our grammars for these languages, without this additional noise.

Maori (mao) is an example of a language where the given data set provides a hard ceiling on how much can be inferred either by a linguist or a machine learning model. The data provided contains only maximally two forms for each verb—the active and the passive. Some examples of active-passive alternations include: **neke** ∼ **nekehia**, **nehu** ∼ **nehua**, **kati** ∼ **katia**. In this data set, the passive form is utterly unpredictable from the active form (but not vice versa). The standard phonological analysis of the data (Kiparsky, 1982; Harlow, 2007)—familiar to many from phonology textbooks—is that the underlying stem contains a consonant which is removed by a phonological rule that deletes word-final consonants in the language. The traditional phonological analysis is that the lemma listed as **neke**, for example, is underlyingly **/nekeh/**, and the passive suffix is regularly **-ia**, while the active suffix is the zero morpheme **-0**. The consonant-deletion rule applies to the active form, which surfaces as **neke**, but not to the

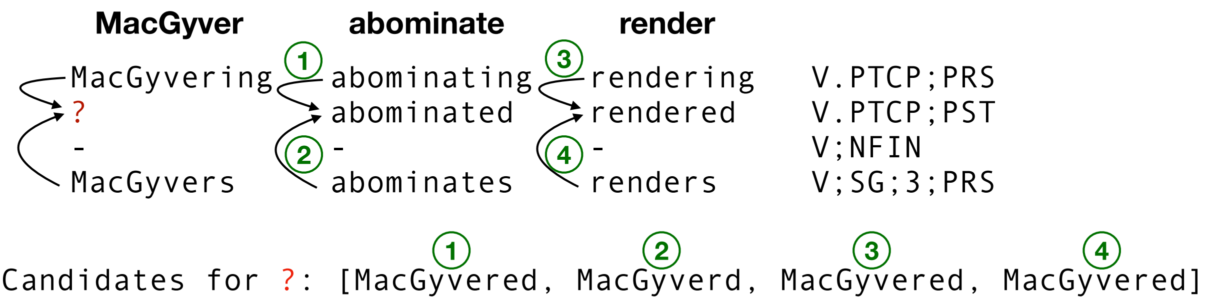

Figure 5: Generating candidate inflections for `V.PTCP.PST` for the verb “to **MacGyver**”. We use all the candidates generated by known transformation rules from all other tables (only 2 other tables shown here). A list of candidate inflections is produced, where the final inflection is decided by majority vote.

passive form **nekehia**, where the added suffix prevents the consonant from deleting. There is also an additional hiatus-avoiding rule—deleting a vowel—seen in e.g. **/nehu/+/ia/** → **nehua**. Obviously, the consonant which is not seen in the active form given in the training data can not be used to predict the passive form. The best one can do is to guess the most likely consonant in the language as being present in the underlying stem. Had the training data contained a third form which maintains the consonant—e.g. the Maori gerundive suffix **/-aŋa/**—the missing consonant of the passive could be predicted from the gerundive and vice versa.[4]

Hiligaynon (hil) contained several lemmas listed with multiple alternate forms, such as:

```
bati/batian/pamatian ginpamantian V;PROG;PST
```

It is very challenging to account for the occasional lemma being listed in two or three parts in a standard FST design, and so this kind of transformation was not attempted.

Syriac, Sanskrit, Oromo, Tohono O’odham (syc,san,orm,ood) contained multiple lines where the lemma and MSD were identical, but the output was not. In some languages this was pervasive enough to cause us to exclude them (ctp,pei) from our selection of attempted languages.

Chichicapan Zapotec (zpv) contained several inflected forms where the target form actually contained two alternatives separated by a slash. Predicting and modeling when this happens was deemed to be irregular and was not attempted.

[4]“If we wanted an A on our [phonology] exam, we would of course say the underlying forms are [the ones with the consonant] . . . If someone were to say that the underlying forms are [consonantless] he’d flunk.” (Kiparsky, 1982)

Zenzontepec Chatino (czn) contained a mixture of hyphens (-) and en-dashes (–) where presumably only one of them should have been used. Again, this was deemed hard to predict manually and no obvious pattern was found.

7 Conclusion

We have done a preliminary investigation in pitting neural inflection models against more traditional hand-written grammars, designed by non-naive grammar developers with some training in the field of linguistics and computational modeling. The results point to two main directions.

First, it is very difficult in many cases to outperform a state-of-the-art neural network model without significant development effort and attention to nuanced morphophonological patterns. Indeed, some data sets in the task were very simple, and in such cases, it is quite trivial to develop a high-accuracy grammar. This advantage is somewhat nullified by the apparent ability of neural seq2seq models to also model such morphologies with high accuracy, despite little data.

The second observation is the following: for languages where the group was able to significantly outperform neural models (such as Tagalog and Ingrian), success did not come cheaply. We estimate that for any language with high morphophonological complexity and a variety of inflectional classes, possibly hundreds of hours of development effort is required even by a trained linguist to surpass the performance of a current state-of-the-art seq2seq model. But it is also precisely in this latter case of high-complexity languages where linguists can still prevail with a margin.

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*, pages 1024–1029, Denver, Colorado. Association for Computational Linguistics.

Kenneth R. Beesley. 2012. Kleene, a free and open-source language for finite-state programming. In *Proceedings of the 10th International Workshop on Finite State Methods and Natural Language Processing*, pages 50–54, Donostia–San Sebastián. Association for Computational Linguistics.

Kenneth R. Beesley and Lauri Karttunen. 2003. *Finite State Morphology*. CSLI Publications, Stanford, CA.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Arya D. McCarthy, Katharina Kann, Sebastian Mielke, Garrett Nicolai, Miikka Silfverberg, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. The CoNLL–SIGMORPHON 2018 shared task: Universal morphological reinflection. In *Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 1–27, Brussels. Association for Computational Linguistics.

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*, pages 1–30, Vancouver. Association for Computational Linguistics.

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*, pages 10–22, Berlin, Germany. Association for Computational Linguistics.

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*, pages 616–627, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Markus Forsberg and Mans Hulden. 2016. Learning transducer models for morphological analysis from example inflections. In *Proceedings of the SIGFSM Workshop on Statistical NLP and Weighted Automata*, pages 42–50, Berlin, Germany. Association for Computational Linguistics.

Ray Harlow. 2007. *Maori: A Linguistic Introduction*. Cambridge University Press.

Mans Hulden. 2009. Foma: a finite-state compiler and library. In *Proceedings of the Demonstrations Session at EACL 2009*, pages 29–32, Athens, Greece. Association for Computational Linguistics.

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*, pages 569–578, Gothenburg, Sweden. Association for Computational Linguistics.

Paul Kiparsky. 1982. *Explanation in Phonology*, volume 4. Walter de Gruyter.

Christo Kirov, Ryan Cotterell, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sebastian Mielke, Arya McCarthy, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. UniMorph 2.0: Universal morphology. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, Miyazaki, Japan. European Languages Resources Association (ELRA).

Kimmo Koskenniemi. 1983. *Two-level morphology: A general computational model for word-form recognition and production*. Publication 11, University of Helsinki, Department of General Linguistics, Helsinki.

Mike Maxwell and Baden Hughes. 2006. Frontiers in linguistic annotation for lower-density languages. In *Proceedings of the Workshop on Frontiers in Linguistically Annotated Corpora 2006*, pages 29–37, Sydney, Australia. Association for Computational Linguistics.

Arya D. McCarthy, Ekaterina Vylomova, Shijie Wu, Chaitanya Malaviya, Lawrence Wolf-Sonkin, Garrett Nicolai, Christo Kirov, Miikka Silfverberg, Sebastian J. Mielke, Jeffrey Heinz, Ryan Cotterell, and Mans Hulden. 2019. The SIGMORPHON 2019 shared task: Morphological analysis in context and cross-lingual transfer for inflection. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 229–244, Florence, Italy. Association for Computational Linguistics.

Sarah Moeller, Ghazaleh Kazeminejad, Andrew Cowell, and Mans Hulden. 2018. A neural morphological analyzer for Arapaho verbs learned from a finite state transducer. In *Proceedings of the Workshop on Computational Modeling of Polysynthetic Languages*, pages 12–20, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Tommi A Pirinen. 2015. Omorfi — free and open source morphological lexical database for Finnish. In *Proceedings of the 20th Nordic Conference* *of Computational Linguistics (NODALIDA 2015)*, pages 313–315, Vilnius, Lithuania. Linköping University Electronic Press, Sweden.

Tommi A Pirinen. 2019. Neural and rule-based Finnish NLP models—expectations, experiments and experiences. In *Proceedings of the Fifth International Workshop on Computational Linguistics for Uralic Languages*, pages 104–114, Tartu, Estonia. Association for Computational Linguistics.

Michael Riley, Cyril Allauzen, and Martin Jansche. 2009. OpenFst: An open-source, weighted finite-state transducer library and its applications to speech and language. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Tutorial Abstracts*, pages 9–10, Boulder, Colorado. Association for Computational Linguistics.

Ekaterina Vylomova, Jennifer White, Elizabeth Salesky, Sabrina J. Mielke, Shijie Wu, Edoardo Ponti, Rowan Hall Maudslay, Ran Zmigrod, Joseph Valvoda, Svetlana Toldova, Francis Tyers, Elena Klyachko, Ilya Yegorov, Natalia Krizhanovsky, Paula Czarnowska, Irene Nikkarinen, Andrej Krizhanovsky, Tiago Pimentel, Lucas Torroba Hennigen, Christo Kirov, Garrett Nicolai, Adina Williams, Antonios Anastasopoulos, Hilaria Cruz, Eleanor Chodroff, Ryan Cotterell, Miikka Silfverberg, and Mans Hulden. 2020. The SIGMORPHON 2020 Shared Task 0: Typologically diverse morphological inflection. In *Proceedings of the 17th Workshop on Computational Research in Phonetics, Phonology, and Morphology*.

Shijie Wu, Ryan Cotterell, and Mans Hulden. 2020. Applying the transformer to character-level transduction. *arXiv:2005.10213 [cs.CL]*.