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 developed 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 hand-written 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).

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 we document an experiment where a number of morphological grammars were hand-written 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.
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.  

- 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 ↔ ran in Figure 3 (b) is converted into the regular expression

\[
?+ \ u:a \ ?+ \ s:0
\]

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 $l_2$:

$$\text{sim}(l_1, l_2) = \frac{2 \times \#\text{shared}(l_1, l_2)}{\#\text{shared}(l_1, l_1) + \#\text{shared}(l_2, l_2)}$$  \hspace{1cm} (2)$$

Here, $\#\text{shared}(l_1, l_2)$ 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:

$$\text{distance}(l_1, l_2) = 1 - \text{sim}(l_1, l_2)$$ \hspace{1cm} (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 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-
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

<table>
<thead>
<tr>
<th>Language</th>
<th>trn(^1)</th>
<th>dev(^1)</th>
<th>tst(^1)</th>
<th>tst(^2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>aka</td>
<td>100.0</td>
<td>100.0</td>
<td>\textbf{100.0}</td>
<td>89.8</td>
</tr>
<tr>
<td>ceb</td>
<td>85.2</td>
<td>86.2</td>
<td>86.5</td>
<td>84.7</td>
</tr>
<tr>
<td>crh</td>
<td>97.5</td>
<td>97.0</td>
<td>96.4</td>
<td>97.7</td>
</tr>
<tr>
<td>czn</td>
<td>79.0</td>
<td>76.0</td>
<td>72.5</td>
<td>76.1</td>
</tr>
<tr>
<td>dje</td>
<td>100.0</td>
<td>100.0</td>
<td>\textbf{100.0}</td>
<td>\textbf{100.0}</td>
</tr>
<tr>
<td>gaa</td>
<td>100.0</td>
<td>100.0</td>
<td>\textbf{100.0}</td>
<td>\textbf{100.0}</td>
</tr>
<tr>
<td>izh</td>
<td>93.4</td>
<td>91.1</td>
<td>\textbf{92.9}</td>
<td>77.2</td>
</tr>
<tr>
<td>kon</td>
<td>100.0</td>
<td>100.0</td>
<td>98.7</td>
<td>97.4</td>
</tr>
<tr>
<td>lin</td>
<td>100.0</td>
<td>100.0</td>
<td>\textbf{100.0}</td>
<td>\textbf{100.0}</td>
</tr>
<tr>
<td>mao</td>
<td>85.5</td>
<td>85.7</td>
<td>66.7</td>
<td>57.1</td>
</tr>
<tr>
<td>mlg</td>
<td>100.0</td>
<td>100.0</td>
<td>\textbf{100.0}</td>
<td>-</td>
</tr>
<tr>
<td>nya</td>
<td>100.0</td>
<td>100.0</td>
<td>\textbf{100.0}</td>
<td>\textbf{100.0}</td>
</tr>
<tr>
<td>ood</td>
<td>81.0</td>
<td>87.5</td>
<td>71.0</td>
<td>62.4</td>
</tr>
<tr>
<td>orm</td>
<td>99.6</td>
<td>100.0</td>
<td>\textbf{99.0}</td>
<td>93.6</td>
</tr>
<tr>
<td>ote</td>
<td>91.2</td>
<td>93.5</td>
<td>90.9</td>
<td>91.3</td>
</tr>
<tr>
<td>san</td>
<td>88.5</td>
<td>89.7</td>
<td>89.0</td>
<td>88.3</td>
</tr>
<tr>
<td>sna</td>
<td>100.0</td>
<td>100.0</td>
<td>\textbf{100.0}</td>
<td>99.3</td>
</tr>
<tr>
<td>sot</td>
<td>100.0</td>
<td>100.0</td>
<td>\textbf{100.0}</td>
<td>99.0</td>
</tr>
<tr>
<td>swa</td>
<td>100.0</td>
<td>100.0</td>
<td>\textbf{100.0}</td>
<td>\textbf{100.0}</td>
</tr>
<tr>
<td>syc</td>
<td>89.3</td>
<td>87.3</td>
<td>88.3</td>
<td>89.1</td>
</tr>
<tr>
<td>tgl</td>
<td>100.0</td>
<td>100.0</td>
<td>\textbf{93.8}</td>
<td>\textbf{93.8}</td>
</tr>
<tr>
<td>xty</td>
<td>77.9</td>
<td>75.0</td>
<td>\textbf{77.8}</td>
<td>-</td>
</tr>
<tr>
<td>zpv</td>
<td>81.1</td>
<td>80.0</td>
<td>81.7</td>
<td>70.3</td>
</tr>
<tr>
<td>zul</td>
<td>84.3</td>
<td>77.9</td>
<td>78.9</td>
<td>81.1</td>
</tr>
</tbody>
</table>

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 handwritten 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 underlingly /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
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 /-a/—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:

\[
\text{bati/batian/pamatian ginpamantian V;PROG;PST}
\]

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)

Zenonpepec 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, there were 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


