Learning from Unseen Data

Constantine Lignos
Department of Computer and Information Science
University of Pennsylvania
lignos@cis.upenn.edu

Abstract

The learner of Lignos et al. (2009) attained excellent performance in English in Morpho Challenge 2009, but its reliance on minimal word pairs in the input to learn which words a rule applies to led to poor performance in other languages. We demonstrate that this learner can perform well across a broader set of languages if it works to infer word forms unseen in the data. We evaluate approaches to compounding and base word inference to accomplish this goal, improving the learner's performance greatly in Turkish and Finnish.

1 Introduction

Data sparsity, as best quantified by Zipf's law, is a defining characteristic of language and its impact is felt in unsupervised morphology learning. The pervasive sparsity both between and within lemmas (Chan, 2008) suggests that learners that rely on identifying paradigms will face significant difficulties while learners that learn independent rules can more reliably succeed with less data.

The learner developed by Chan (2008) and extended by Lignos et al. (2009) embraces sparsity by learning independent morphological rules and using characteristics of the distribution of inflected forms to guide the learner's design. This learner was evaluated in Morpho Challenge 2009 (Kurimo et al., 2009) in English and German, but it was not able to model agglutinative languages as the learner reached the limitations of learning rules by minimal pairs of words present in the corpus. While it effectively avoided the difficulty of learning a paradigm representation, it struggled with the sparsity caused by languages in which words consist of many morphemes and there are rarely corresponding minimal pairs for each one.

We extend that learner in this paper by adding features that allow the learner to infer words not present in the corpus, allowing it to succeed without changes to the core minimal pair-based learning model.

2 The Learning Framework

We present a brief summary of the Base and Transforms model and the core operations of the learning algorithm. For further details, see Lignos et al. (2009) and Chan (2008).

2.1 The Base and Transforms Model

A morphologically derived word is modeled as a base word with an accompanying transform that changes the base to create a derived form. A transform is an orthographic modification made to a base to create a derived form. It is defined by two affixes $(s1, s2)$, where $s1$ is removed from the base before concatenating $s2$. A null suffix is represented as *$*. A transform also has a corresponding word set, which is the set of base-derived pairs that the transform accounts for.

Word Sets. Each word in the corpus belongs to one of three word sets at any point in execution: Base, Derived, or Unmodeled. The Base set contains the words that are used as bases of learned transforms but are not derived from any other form. The Derived set contains words that are derived forms of learned transforms; these words can also serve as bases for other derived forms. All words begin in the Unmodeled set and are moved into Base or Derived as transforms are learned.

2.2 The Learning Loop

Each iteration, the learner does the following:

1. Counts the affixes that appear in each word set, ignoring low frequency words.

2. Hypothesizes transforms between pairs of the most frequent affixes and scores each transform using the number of word pairs it models and the amount it changes the base word. For example, the transform *($, s)* can model the pair *paper/papers*.

3. Selects the highest scoring transform and moves the words modeled by that transform into the Base and Derived sets as appropriate.

The learning loop continues until none of the highest ranked transforms meet the criteria for an acceptable transform. After learning is complete, each word is analyzed using its base word and any transforms required to derive it from the base.

3 Additions to the Framework

While the innovations introduced by Lignos et al. (2009) addressed the largest gaps between the cognitive model proposed by Chan (2008) and the requirements of a morphological analyzer, as shown in the results of Morpho Challenge 2009 (Kurimo et al., 2009) the algorithm's success was primarily limited to English. Modest results were reported in German, but the algorithm was not submitted for other languages because it was unable to handle languages with many morphemes per word. We add the following features to allow the learner to maintain its core learning process while adding additional words to the lexicon as it models the corpus. We show how our features integrate with the algorithm of Lignos et al. (2009) in Figure 1.

3.1 Base Inference

As an example of the limitations of applying rules based on minimal pairs, consider three words appearing in the Brown corpus (Francis and Kucera, 1967): *adjoins, adjoined, adjoining*. Even though the transforms *($, s)*, *($, ed)*, and *($, ing)* are learned, they cannot be used to model these three words because the required base, *adjoin*, is not present in the corpus. This results in lower recall because these three words remain unmodeled despite the fact that the rules required to model them have been learned.

To address situations of this type, we introduce *base inference* to infer the existence of an unseen word when more than one learned transform suggests that it should exist. After each rule is learned, the learner iterates over every word with the transform's $s2$ affix that was not successfully modeled by the transform and notes the base that would have been required to model that word. If a later transform requires the same base to model another word, the learner infers the existence of that base and then attempts to use that base in all transforms learned to that point and those learned later.

For example, consider the learner's operation when learning the transforms *($, s)*, *($, ed)*, and *($, ing)* in order while *adjoins*, *adjoined*, and *adjoining* are present in the corpus but *adjoin* is not. After the transform *($, s)* is learned, the learner iterates over all words containing the suffix *-s* that were not modeled, including *adjoins*, and notes the required base, *adjoin*. In the next iteration, the learner selects the transform *($, ed)* and similarly notes that the required base for *adjoined* is *adjoin*. Because two different rules have indicated the possible existence of *adjoin*, the learner infers its existence. The learner adds it to the lexicon with the frequency of the word that caused it to be inferred and marks it for the transforms *($, s)* and *($, ed)*, thus modeling *adjoins* and *adjoined*. In the next iteration, since *adjoin* is now in the Base word set, when the transform *($, ing)* is learned, *adjoining* can be modeled as if *adjoin* were present in the input corpus.

3.2 Compounding

The model presented by Lignos et al. (2009) performs a simple n-gram based compounding as a post-processing step on the learner's output. But as with base inference, it would be beneficial for the algorithm to be able to use the components of a compound word during learning instead of only breaking compounds after analysis.

We adopt the compounding model of Koehn and Knight (2003) where a word is broken down into the set of component words present in the lexicon with the highest geometric mean of frequencies. Thus the splitter selects a split using the following equation for a split S comprised of $w_i \dots w_n$ component words:

$$\arg\max_{S}\left(\prod_{w_i \in S} count(w_i)\right)^{\frac{1}{n}}$$

The null hypothesis is also considered where S contains only the word being split, thus a word is only split if the geometric mean of its component

1. Add all words in the corpus to the Unmodeled set.
2. Until a stopping condition is met, perform the main learning loop:
   (a) Score suffixes and transforms and select the best transform.
   (b) Move the words used in the selected transform.
   (c) Optionally perform Base Inference, inferring new bases and adding them to learned transforms as appropriate.
   (d) Optionally perform compounding for the current iteration.
3. Optionally perform compounding after learning is complete.

Figure 1: An overview of the learning algorithm, integrating compounding and inference features

words' frequencies is greater than the word's frequency. Koehn and Knight also use "filler," character sequences that can be placed between component words of a compound. Rather than specifying the filler sequences by hand, we allow the filler to be the application of a previously learned transform to a word in the lexicon when including it in the compound. The learner only allows component words from the Base or Derived sets to have filler added to them; this helps in excluding morphologically unproductive words from having transforms applied to them for the purpose of compounding.

If a compound word is split using a transform applied to a word in the lexicon, the derived form is added to the lexicon and marked as derived from the word the transform was applied to. For example, consider an example from Koehn and Knight (2003) where we are splitting *Aktionsplan*. Assume that *Aktion* and *plan* are in the lexicon but *Aktions* is not and that the learner has learned the transform *($, s)*. Assuming the correct frequency requirements are met, the learner would break the compound as *Aktions* and *plan*, where *Aktions* was derived by applying *($, s)* to *Aktion*. The learner would add *Aktions* to the lexicon, noting the relationship to *Aktion*. This provides the companion to Base Inference for derived words; the learner infers the existence of a derived form by its presence in a compound. The learner is then able to learn words derived from *Aktions* if needed, a crucial ability in agglutinative languages which typically contain many compounds where the component words can take a large number of suffixes but may not be observed elsewhere in the corpus.

We apply the compounding approach in three variants:

Basic Compounding. Compounding is applied to words in the Base and Unmodeled sets after all learning is complete, and no transforms are supplied as fillers for the compounding system.

Iterative Compounding. Compounding is applied to words in the Base set after every iteration and to the Unmodeled set after all learning is complete. The transforms learned up to the current iteration are always supplied as fillers for the compounding system.

Aggressive Compounding. Compounding is applied to words in the Base and Unmodeled sets after every iteration. As in Iterative Compounding, the transforms learned up to the current iteration are always supplied as fillers for the compounding system, but the key difference is that they are applied to the words in Unmodeled every iteration, not just when learning is complete. This is more aggressive because it introduces many more words during the learning process than if unmodeled words are only split after learning is complete.

4 Results

The learner's performance on the development set of Morpho Challenge 2010 is given in Table 1. The *Base* condition gives the performance of the learner without any compounding or word inference features active. The *Basic Compounding* condition gives the performance of the learner with Basic Compounding in use but without Base Inference. The *Base Inference* condition builds on the Basic Compounding condition, adding the Base Inference feature. The *Iterative Compounding* and *Aggressive Compounding* conditions build on the Base Inference condition, with their forms of compounding superseding Basic Compounding.

The results show that while the features introduced lead to mixed results on the F-score for En-

	Precision	Recall	F-score
English			
Base	**60.56**	50.47	55.05
+Basic Compounding	59.26	52.82	55.85
+Base Inference	59.26	**54.21**	**56.62**
+Iter. Compounding	57.80	52.07	54.79
+Aggr. Compounding	46.57	51.81	49.05
Finnish			
Base	69.19	09.89	17.30
+Basic Compounding	65.55	26.32	37.55
+Base Inference	**76.40**	26.84	39.72
+Iter. Compounding	72.85	29.32	41.81
+Aggr. Compounding	54.36	**44.49**	**48.93**
German			
Base	**57.72**	28.03	37.73
+Basic Compounding	42.17	34.08	37.70
+Base Inference	44.74	34.22	**38.78**
+Iter. Compounding	46.69	32.78	38.52
+Aggr. Compounding	38.52	**35.02**	36.69
Turkish			
Base	**70.00**	09.73	17.08
+Basic Compounding	61.68	12.78	21.17
+Base Inference	50.33	13.74	21.59
+Iter. Compounding	49.45	19.56	28.03
+Aggr. Compounding	35.19	**31.63**	**33.31**

Table 1: Learner performance on the Morpho Challenge 2010 development sets

glish and German, they improve F-score greatly in Turkish and Finnish. The Basic Compounding feature results in little change in English and German but moderate improvement in Turkish and a dramatic improvement in Finnish. Both Turkish and Finnish benefit especially from Aggressive Compounding, but while German sees a very small performance drop as compared to Iterative Compounding, the drop in English precision and F-score is large.

While in all languages the Base Inference feature led to a small gain in F-score, the impact of the feature was less than was expected. The likely cause is that the conditions that motivate the base inference feature are rare; the algorithm's design leads to the base words of a transform being more frequent on average than the derived words, so Base Inference only handles a small number of exceptions that are unlikely to be evaluated by the small development set. We expected Base Inference to provide a gain in recall with little impact to precision, but the Finnish and German results puzzlingly show precision improvements with almost no recall improvements.

Based on these results, we are submitting the analyses of the Base Inference, Iterative Compounding, and Aggressive Compounding conditions to be evaluated in Morpho Challenge 2010.

5 Discussion

The features present here succeed in transforming the learner into one better suited for agglutinative languages. The strongest improvement in agglutinative languages, however, comes at the expense of precision, most notably when Aggressive Compounding is used. Ideally, a single technique would result in the greatest performance across all languages, but as evaluated with the Morpho Challenge 2010 development set Aggressive Compounding results in the best results for Turkish and Finnish, near-best results for German, and the worst results for English.

As the name implies, Aggressive Compounding applies any transform learned to a word in the Base or Derived sets without any further criteria or any penalty for applying the transform. Given the significant drops in precision caused by using Aggressive Compounding, it appears that a restriction or regularization of some form is required. This points out a fundamental shortcoming of this learner: the learner does not understand the condition for applying a transform beyond suffixes on the base word. Learning part of speech information would likely help the learner decide whether a word can take a transform or not as a word's part of speech can determine what morphemes can be used with it.

The improvements to the learner for Morpho Challenge 2010 further support our position that a non-statistical approach to morphology learning can succeed in a variety of languages.

References

E. Chan. 2008. *Structures and distributions in morphology learning*. Ph.D. thesis, University of Pennsylvania.

S. Francis and H. Kucera. 1967. Computing analysis of present-day American English.

P. Koehn and K. Knight. 2003. Empirical methods for compound splitting. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 187–193. Association for Computational Linguistics.

Mikko Kurimo, Sami Virpioja, Ville T. Turunen, Graeme W. Blackwood, and William Byrneg. 2009. Overview and results of Morpho Challenge 2009. In *Working Notes of the 10th Workshop of the Cross-Language Evaluation Forum*, Corfu, Greece, September 30–October 2. CLEF2009.

Constantine Lignos, Erwin Chan, Mitchell P. Marcus, and Charles Yang. 2009. A Rule-Based Unsupervised Morphology Learning Framework. In *Working Notes of the 10th Workshop of the Cross-Language Evaluation Forum*, Corfu, Greece, September 30–October 2. CLEF2009.