A Rule-Based Unsupervised Morphology Learning Framework*

Constantine Lignos†, Erwin Chan‡, Mitchell P. Marcus†, Charles Yang†
†University of Pennsylvania, ‡University of Arizona
lignos@cis.upenn.edu, echan3@email.arizona.edu,
mitch@cis.upenn.edu, charles.yang@ling.upenn.edu

Abstract

We use the Base and Transforms Model proposed by Chan [1] as the core of a morphological analyzer, extending its concept of base-derived relationships to allow multi-step derivations and adding a number of features required for robustness on larger corpora. The result is a rule-based morphological analyzer, attaining an F-score of 58.48% in English and 33.61% in German in the Morphochallenge 2009 Competition 1 evaluation.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.1 Content Analysis and Indexing; H.3.3 Information Search and Retrieval; I.2 [**Artificial Intelligence**]: I.2.6 Learning; I.2.7 Natural Language Processing

General Terms

Measurement, Performance, Experimentation

Keywords

Language acquisition, Text analysis

1 Introduction

We address the issue of learning a rule-based representation of morphology by modeling the words of an unannotated corpus using a greedy approach to learn the most salient rules of the language. We begin with the model of morphology and matching algorithm presented by Chan in his dissertation [1] and present a number of modifications to the algorithm to improve performance on larger corpora, support multi-step derivations, and output word analyses.

There are many approaches to morpheme induction, well summarized by Goldsmith [2] among others, and thus here we briefly discuss only the psychologically-motivated models most relevant to our learner. Many leading psychologically-motivated approaches to morphological acquisition [6, Rumelhart], [5, Pinker] assume that pairs of morphologically related items are provided to the learner, e.g. "walks/walked, bring/brought," which makes these techniques generally unusable in a fully unsupervised setting. Unsupervised approaches have often focused on transitional probabilities [3, Harris], [4, Keshava], which provide a good approach to discovering affixes and

*We thank Jana Beck for her assistance in analyzing the German results and for her insightful comments throughout the development process.

segmentation points, but their emphasis on pure concatenation causes difficulty in the simplest word pairs, such as *make* → *making*.

We believe the largest barrier to success in morphology learning is the sparsity of the data that underlies language at all levels, including the morphological level as shown by Chan. The learner presented here is designed not only to function in a Zipfian world but to exploit the Zipfian distribution of derived forms to identify the "base" of a morphologically related group of words. While the learner still suffers from the effects of sparsity when learning multistep derivations, we believe this designed-for-sparsity approach is an effective cognitive model, and our results show it to be an effective engineering solution to the problem of morphological segmentation.

2 Methodology

We use the Base and Transforms Model developed in chapter 5 of Erwin Chan's dissertation [1] and extend the given algorithm to create a morphological analyzer. We first describe the original model and then the extensions we made to it to adapt it for a morphological analysis task.

2.1 The Base and Transforms Model

2.1.1 Base and Derived Forms

A morphologically derived word is modeled as a base word and a morphological transform that modifies the base to create a derived form.[1] A base is not an abstract stem, but rather the most frequent form of a morphologically related set of words. As shown by Chan [1], in a wide[2] variety of languages the base is easily identified by being at the top of a Zipfian frequency distribution of morphologically related words and its form is consistent among members of a category. For example, in Greek among nouns it is consistently the accusative singular form. This property, called base homogeneity, allows us to select the most type-frequent transforms and in doing so automatically select the correct bases within each category.

2.1.2 Transforms

A transform is an orthographic modification made to a base to create a derived form. It is defined as two affixes[3] $(s1, s2)$, where $s1$ is removed from the base before concatenating $s2$. Thus to derive *making* from *make* we apply the transform (e, ing), removing $-e$ from the base and then concatenating $-ing$. We represent a null suffix as \$, thus a purely additive suffix is of the form $(\$, x)$ and a purely subtractive one is of the form $(x, \$)$, where x is not null. The concept of a base and transform is similar to traditional approaches involving an abstract stem and a set of rules that change it into derived forms. The key contribution of the Base and Transforms model is that it allows statistical identification of a base and the rules that it allows without relying on the induction of an often unobserved stem.

A transform also has a corresponding word set, which is the set of base-derived pairs that the transform accounts for. The bases[4] of a transform are the only words that the transform can be applied to.

[1] In the context of [1], a derived word is the result of a single transform being applied to a base word. We extend this to allow a chain of multiple transforms between a base form and a derived form, provided that all intermediate forms are observed in the corpus.

[2] Greek, Swedish, Spanish, and English are most explicitly analyzed, but evidence is also given for Finnish, Hungarian, Slovene, and Catalan.

[3] In the original model presented by Chan [1], only suffixes are explicitly discussed but we trivially extend the model to also include prefixes.

[4] Note that we use the lowercases "bases" for the words that are the bases used by a particular transform, while later we use "Bases" to signify the set of words in the language serving as bases in learned transforms.

1. Place all words in the corpus in the Unmodeled set
2. For a specified number of iterations:
   (a) Count suffixes in words of the Base ∪ Unmodeled set and the Unmodeled set.
   (b) Hypothesize transforms from words in Base ∪ Unmodeled to words in Unmodeled.
   (c) Select the best transform.
   (d) Move the bases used in the transform to the Base set, and the derived forms modeled by the transform from the Unmodeled set to the Derived set.

Figure 1: The Original Learning Algorithm

2.1.3 An Algorithm for Discovering Transforms

In Chan [1], the algorithm in Figure 1 is given for discovering a set of transforms to model the words of a corpus. 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. Similarly, the derived set contains words that are derived forms of learned transforms. All words begin in the Unmodeled set and are moved into Base or Derived as transforms are learned. When hypothesizing transforms, the algorithm creates word pairs whose bases are in the Base or Unmodeled set and whose derived forms are in Unmodeled. Except where otherwise noted, the frequency of words in the corpus is ignored; all counts regarding affixes and transforms are based on the number of types, not tokens, they represent.

We now present the learning loop in detail:

Count Suffixes
Iterate over the words in two sets, the Base ∪ Unmodeled set and the Unmodeled set, and count all of the suffixes of length 0-4 contained in each word, maintaining a separate count for the suffixes in each set. For example, the word "hopeless" contains the suffixes (-$, -s, -ss, -ess, -less), and if it is only in the Base set those affixes would be counted toward the Base ∪ Unmodeled set's affix counts, not the Unmodeled set's. A suffix is only counted if removing it leaves a sufficiently long stem, in this case 3 characters long. This length limitation exists to prevent the modeling of extremely short words that are likely closed-class or morphologically irregular words. Affixes are ranked by the number of types they appear in.

Hypothesize Transforms
Hypothesize transforms of all combinations of the top 50 affixes as $s1$ and $s2$ of a transform. For example, from the common English suffixes $, s, and ing the transforms $(\$, s)$, $(s, \$)$, $(\$, ing)$, $(ing, \$)$, (ing, s), (s, ing) are hypothesized. For each hypothesized transform, check every word in the Base ∪ Unmodeled set and verify that the word that is the result of applying the transform is in the Unmodeled set. If it is, add the base-derived pair to the word set of this transform. Transforms are ranked by the number of word pairs they account for, without regard to the frequency of the words in those pairs.

Select a Transform
The highest ranked transform is selected, provided its overlap ratio is acceptable. A transform's overlap ratio is calculated as the ratio of the stem overlap to base overlap, where they are defined as follows by Chan [1]:

> The base overlap is the number of base forms in the proposed transform that are base forms in the current grammar. The stem overlap is the number of base forms' stems

(computed as the first four characters) in the proposed transform that are also stems of base forms in the current grammar. The stem overlap is a rough computational approximation of the lexical similarity between two sets of words.

A high overlap ratio implies that the bases used in the transform's word set are very similar to words in the Base set but not members of it. The likely cause is that the bases used in the transform are inflected forms of words in the Base set, and thus accepting the transform would cause the Base set to include multiple inflections of the same lexical category, which is undesirable.[5]

If the first and second ranked transforms account for the same number of types and are symmetric pairs, for example in English the transforms $(\$, s)$ and $(s, \$)$, a special tie-breaking procedure is invoked. For each of the transforms, count the number of base-derived pairs in which the base form is more frequent in the training corpus than the derived form. Choose the transform with the greater number of higher-frequency bases. This leads to the selection of a transform with more frequent bases, consistent with the definition of a base as the most frequent morphological form within a category. This tie-breaking procedure is typically only needed in the first iteration where the Base set is empty.

2.2 Limitations of the Original Model/Algorithm

While the original base and transforms model provides a simple, learnable, and psychologically reasonable representation for the morphology of a language, it lacks important features of an unsupervised morphological analyzer:

1. **Limited decomposition**- Once a word is modeled with a single transform it cannot be modeled further. Thus a word is either unmodeled, a base, or a base with a single transform applied.

2. **No support for hyphens and compounding**- Hyphenated and compound words are treated as simple words.

3. **Manual stopping control**- The algorithm runs for a hand-specified number of iterations, and has no way of detecting a desirable stopping point.

4. **No concept of derivational versus inflectional morphology**- The algorithm cannot identify the function of a morphological transform, which along with part of speech information[6] can further constrain the application of rules and suggest their function.

5. **No model of morphophonemics**- Because the application of a transforms is a simple deletion/concatenation operation, there is no accommodation of morphophonemics such as $-s$ and $-es$ both representing the noun plural in English, or orthographic phenomena, such as the deletion of a silent final e in English when morphemes are attached.

Additionally, it presents a number of problems when trained on large corpora:

1. **Poor robustness to noisy data**- All types are considered equally important, regardless of their frequency in the corpus. While the intuition that a morphological transform's salience should be based on the number of types it applies to is correct, the large number of foreign and misspelled words in large corpora can cause spurious transforms to be learned that are surprisingly type frequent, for example $(\$, o)$ in English corpora from pairs like *poll*/*pollo*, *carmel*/*carmelo*, *carl*/*carlo*, etc.

2. **Limited transform metrics**- While a base/stem overlap ratio is computed to determine whether a transform should be rejected for specifying a primarily derived-derived relationship, there are no other ways of determining how likely a transform is a morphological phenomenon and not merely an orthographic regularity.

[5] For a more thorough explanation of the importance of base homogeneity, see [1], chapter 4.

[6] In later chapters of [1] a part-of-speech clustering system is introduced that can learn the part of speech of a base form, but not a derived form.

1. Pre-process words and populate the Unmodeled set.
2. Until a stopping condition is met, perform the main learning loop:
   (a) Count suffixes in words of the Base ∪ Unmodeled set and the Unmodeled set.
   (b) Hypothesize transforms from words in Base ∪ Unmodeled to words in Unmodeled.
   (c) Select the best transform.
   (d) Reevaluate the words that the selected transform applies to, using the Base, Derived and Unmodeled sets
   (e) Move the words used in the transform accordingly.
3. Break compound words in the Base and Unmodeled sets.

Figure 2: The Modified Learning Algorithm

Many of these limitations are briefly discussed in the original presentation of the algorithm, but no concrete solutions are given.

2.3 Enhancements to the Core Algorithm

We added a number of enhancements, detailed in Figure 2, meant to address the limitations of the original algorithm. In many cases we have used a crude approximation for a complex linguistic problem, for example the simple trick used of "doubling" and "undoubling" used to approximate morphophonemic and orthographic phenomena. The success of these enhancements should not be taken as an endorsement of these simple methods but rather as evidence that a more principled approach to these problems could yield significant benefit.

2.4 Pre-processing

We perform a minimal amount of pre-processing to support learning on hyphenated words. Any word with a hyphen is placed into a set of words excluded from the learning process, but each segment in the hyphenated word is included in learning. For example, *punk-rock-worshipping* would not be included in learning, but *punk*, *rock*, and *worshipping* would. The analysis of any hyphenated word is the concatenation of the analysis of its segments, in this case *PUNK ROCK WORSHIP +(ing)*.

2.5 The Learning Loop

2.5.1 Affix Ranking

We count the affixes contained in each word in the base and unmodeled sets as in the original algorithm, but extend the length of affixes allowed and support prefixes in addition to suffixes. We allow the null prefix and prefixes between length 2 and 5, and allow suffixes between length 0 and 5.[7] Affixes are ranked by the number of types they appear in, weighted by the length of the affix, with a null affix considered length one. Thus a length one (or null) affix appearing in 50 types would get a score of 50, while a length two affix appearing in 50 types would get a score of 100. This weighting compensates for the fact that shorter affixes are always more type-frequent, as they occur at least as often as any longer affixes they are a subsets of.

[7]Prefixes of length 1 are excluded because in development on the Brown corpus a large number of spurious single-character transforms were learned. Ideally, with better metrics to gauge transform quality, any limitation on the lengths of affixes would only be necessary to limit the amount of computation required.

To prevent rare words and foreign words from affecting the affix and transform ranking process, while we do include all possible words in the word set of affixes and transforms, they only count toward an affix or transform's score if they are relatively frequent in the corpus. For a word to be considered common, it must appear more than once in the corpus and have a frequency greater than 0.000001, one in one million. In the English Morphochallenge 2009 wordlist, this results in using words that occur more than 62 times in the corpus, which means that 28,439 or about 7.4% of the 384,903 types in the English word list count toward transform/affix ranking. While this seems like a rather low threshold, some common transforms in English are rarely observed in this set- only six pairs for the prefix $re-$ are observed, and only two for the prefix $un-$. This frequency cutoff was set by examining the list of words in the corpus above the cutoff frequency to find a point where less common morphological productions are still included but most typos and foreign words are excluded.

2.5.2 Transform Ranking

As in the original algorithm, we hypothesize transforms of all combinations of the top affixes and count the number of base-derived pairs in each transform. A base-derived pair only counts toward the transform's score if both the base and derived words are in the subset of common words in the corpus. The score of a transform is the number of common pairs multiplied by the net number of characters that the transform adds or removes to a base. For example, if the transform (e, ing), which removes one letter from the base and adds three, has 50 common base-derived pairs, its score would be $50 * |3 - 1| = 100$.

To approximate orthographic gemination and the merging of repeated characters when a morpheme is attached, we relax the conditions of testing whether a base-derived pair is acceptable. For each potential base word for a transform, we compute two derived forms- a standard derived form that is the results of applying the transform precisely to the base, and a "doubled" derived form where $s1$ is removed from the base, the last character of the remaining base is repeated, and then $s2$ is attached. For example, when checking the transform (\$, ing) applied to run, we generate the standard derived form $runing$ and the doubled form $running$. Additionally, in cases where the final character of the base after $s1$ has been removed is the same as the first character of $s2$, we also create an "undoubled" derived form where the first character of $s2$ is removed such that applying the transform does not result in a repeated character. For example, when applying (\$, ed) to $bake$, the standard form would be $bakeed$, but the undoubled form would be $baked$. All derived forms that are observed in the Unmodeled set are added, so if the standard, doubled, and undoubled forms are all observed, three base-derived pairs would be added to the transform.

2.5.3 Transform Selection

Segmentation precision is calculated for each transform. Segmentation precision represents the probability that, given an Unmodeled word containing $s2$, undoing the transform in question will result in a word. It is calculated as the number of common base-derived pairs in the transform over the number of common words in Unmodeled that have the suffix $s2$. A transform would have a segmentation precision of 1.0 if every common word in the Unmodeled set word with the suffix $s2$ was included in the transform's common word pairs, and thus removing $s2$ and concatenating $s1$ always results in a word. Segmentation precision must exceed a set threshold for the learner to accept a hypothesized transform.

In practice, the segmentation precision of a desirable transform varies greatly. The highest observed segmentation precision on the English Morphochallenge 2009 data was for the suffix transform (\$, $'s$), the English possessive, with a value of 97.3%, and the second highest was the suffix transform (y, ies), a form of the English plural noun or third person singular verb for words ending in $-y$, with a value of 75.98%. Many desirable transforms, especially derivational ones, have extremely low precision; the prefix (\$, re) has a precision of 0.59%. By observing the precision of transforms during development against the Brown corpus, we set a threshold of 1% as the threshold of an acceptable transform.

As in the original algorithm, the overlap ratio is calculated for all hypothesized transforms. As with segmentation precision, we observed the output of the learner on the Brown corpus to select the value of 5 for the overlap ratio. Any transform with a higher overlap ratio is rejected. In the Brown corpus these are usually clearly derived-derived transforms such as the suffix transforms (ing, ed). These transforms are hypothesized after appropriate transforms for $-ing$ and $-ed$ have been learned (($, ing), (e, ing) and ($, ed)) because of missing bases in the corpus. For example, late in learning against the Brown corpus "ionizing" and "ionized" remain unmodeled because "ionize" was never observed in the corpus and thus the transforms (e, ing) and ($, ed) cannot be used to model its derived words. The learner hypothesizes (ing, ed) as a way to model these remaining words, but its overlap ratio is 15.0, and thus it is rejected. In the case of the Morphochallenge 2009 data, however, in our runs against English and German the overlap ratio is never used to reject a transform.[8]

If more than 20 transforms are rejected in an iteration because of unacceptable segmentation precision or overlap ratio, the learning loop stops as it is unlikely that there are good transforms left to model. As with other parameters, this limit was selected during development against the Brown corpus and applied to the Morphochallenge 2009 data for English and German.

2.5.4 Transform Word Set Selection

The original algorithm only allows a word to be created by a single transform, so multistep derivations, such as hope → hopeless → hopelessness were not learnable. This stems from the way that transforms are selected; a hypothesized transform changes a word in the Base or Unmodeled set into a word in the Unmodeled set. Once a transform is selected, the bases used in it are moved into the Base set and the derived forms moved into Derived. Thus any word that has is a derived form cannot be the base of another transform, and any word that is a base form cannot be the derived form of another transform.

While we still use the original algorithm's process for selecting the best transform, once we have selected the best transform we try to apply it as broadly as possible by relaxing the allowable sources of base/derived pairs. In the original algorithm, a base/derived pair could consist of words from the following sets: Base → Unmodeled, Unmodeled → Unmodeled. We relax the constraints to allow the following combinations in addition: Base → Base, Derived → Unmodeled. This allows bases to be further broken down, for example to model derivational affixes on an inflectional base, and derived forms to serve as bases for unmodeled words.

This expansion of the permissible types of base/derived pairs requires similar changes to how words are moved between sets once a transform has been selected. We use the following logic:

1. No word in Base may be the derived form of another word. If a word pair of the form Base→ Base is used in the selected transform, the derived word of that pair is moved to Derived, so after movement the relationship is of the form Base → Derived.

2. A word in Derived may be the base of another word in Derived. If a word pair of the form Derived → Unmodeled is used in the selected transform, the derived word of that pair is moved to Derived, and the base word remains in Derived. After movement the relationship is of the form Derived → Derived.

2.6 Post-processing

Once the learning loop has stopped, we try to break the compound words that remain in the Base and Unmodeled sets. We train a 4-gram character-level model on the words in Base, which after

[8] Our explanation for this is that the overlap ratio is only critical in our version of the algorithm if the corpus is relatively small. On a smaller corpus, the set of common words we are using for scoring affixes and transforms is a large proportion of the entire corpus (in a million-token or smaller corpus, only hapaxes are excluded). Because the threshold for a common word is so low, it is likely that we will see derived forms in the common words whose bases do not appear in the corpus at all. In a larger corpus, since the standard for a common word is much higher, it is less likely that a common word's base will not appear in the corpus, and thus less probable we fill fail to model a common word because of a missing base.

successful learning should be largely free of affixes. To segment a word, we use the 4-gram model to find the largest drop in forward transitional probabilities between characters of a word, and if the substrings to either side of that split point are words observed in the corpus, we split the word there and then recursively segment the substrings. The model is unsmoothed and has no backoff—unseen transitions are given zero probability—and we do not check other split points if the optimal one is not a proper segmentation. While there would likely be benefit to a more sophisticated model, we opted to use the simplest model possible and consider this a simple engineering solution to a problem difficult to solve without syllabic or stress information.

2.7 Analysis

When the learning loop has completed, the output is a set of Base, Derived, and Unmodeled words with transforms connecting morphologically related words. In the case of multistep derivations, a derivation chain is formed from a word in Base to multiple words in Derived. For example, in English we may end with the chain $bake \rightarrow baker \rightarrow bakers$, where the transforms ($\$$, er) and ($\$$, s) form the links in the derivation chain. We refer to the leftmost word in the derivation chain as the root of all words to the right of it in the chain. The analysis of a word is the concatenation of the analysis of its root and the analysis of all transforms between the root and the word.

Typically, the analysis of a root is just the word itself, which we output in uppercase. If we have marked a word as a compound (or hyphenated as earlier described) the word's analysis is the concatenation of the segments that make up the compound.

The analysis of a transform takes the form of a prefix/suffix marker, which is simply a + character to the left or right accordingly, and a display form of the suffixes themselves. If $s2$ is not null, only $s2$ is given in the analysis - we assume $s1$ serves as an orthographic accommodation and is not morphologically meaningful. This results in the analysis of the suffixes ($\$$, ing) and (e, ing) both being $+(ing)$, improving recall since these are considered the same morpheme in the analysis.

3 Results

3.1 Performance

The running time of the learner is constrained by the number of affixes used to hypothesize transforms, the number of types in the corpus, and the number of iterations executed. For each affix type (prefixes and suffixes), each iteration the learner keeps the top 50 affixes, hypothesizes transforms of all 200 non-repeating pairs of those affixes, and then to score each iteration it iterates over all words in Base and Unmodeled. Thus we can express the running time as $O(n(atb^2))$, where n is the number of iterations, a is the number of affix types, t is the number of types in the corpus, and b is the number of affixes used to hypothesize transforms each iteration. The running time characteristics, especially the dependence on the number of types in the language and the large number of transforms scored each iteration are unattractive given the goal of a psychologically plausible model. An online or batch learning model rather than examining all data every iteration would yield a more acceptable running time.

Learning on the Morphochallenge 2009 word lists completes in 92 minutes in English and 375 minutes in German when run on a 3GHz Intel Xeon CPU under 64-bit Python using the Psyco module. The large difference in running time between languages is caused by the number of types: 384,903 in English, 1,266,159 in German. The learner completes 27 iterations in English and 26 iterations in German before stopping. The resulting analyses achieve an F-measure of 58.48% in English and 33.61% in German in the official Morphochallenge 2009 competition 1 evaluation.

Figure 3 plots the performance of the learner after each iteration on the sample words pairs released for development for Morphochallenge 2009. The final iteration in each chart is the post-processing step. When run on English data, the precision of the first transforms learned is very high, and as learning progresses recall improves and precision drops as we form more morphological

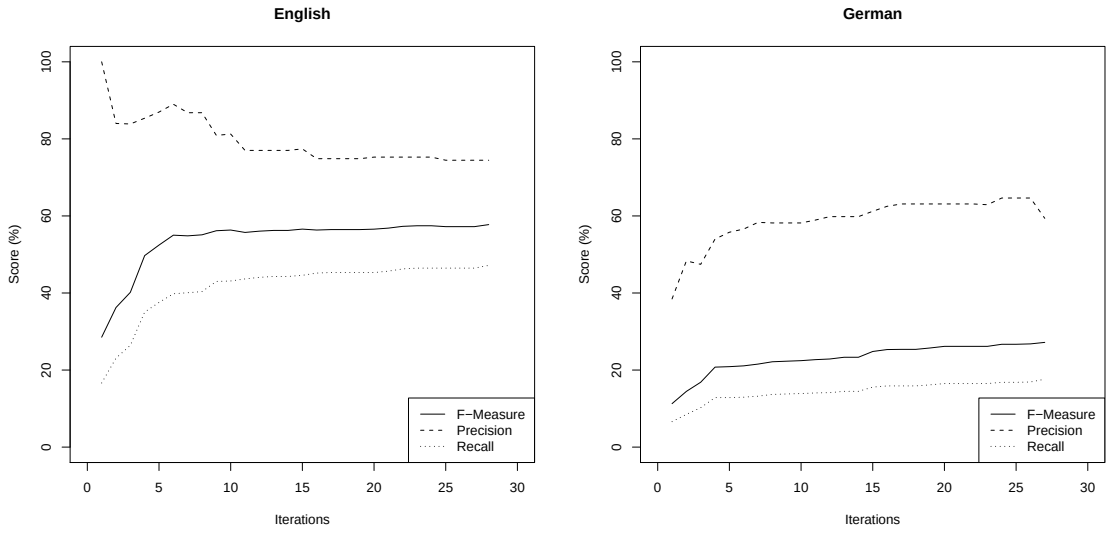

Figure 3: Learner performance in English and German, as evaluated using the development sample word pairs

relationships between words of the corpus. In German, however, while recall steadily improves as in English, precision starts low and steadily increases. There is no clear explanation for why the precision is so different than in English, but it appears that the first transforms learned in German, $(\$, en)$, $(\$, er)$, $(\$, es)$, create a large number of spurious base-derived pairs, resulting in low precision from the start. As later, less frequent rules are learned, it appears there are fewer spurious relationships formed, causing precision to slowly recover. While the post-processing step improved F-measure in both languages, in English it does not affect precision at all while significantly lowering it in German. The compounding model is trained on the presumably uninflected words in Base, so in languages like German where words used in compounds are often inflected we would expect our compounding approach to be less effective.

3.2 Errors

While it is difficult to assign precise, mutually exclusive categories to the learner's errors, they can be loosely grouped into these categories:

1. **Rare affixes**– Many productive affixes in the gold standard are rarer than would be expected in the training corpus, for example the English suffixes $-ness$ and $-able$, and thus the learner fails to distinguish them from orthographic regularities or simply noise in the data.

2. **Unproductive affixes**– Some affixes in the gold standard are no longer productive in the language being learned. For example, the gold standard suggests that *embark* be analyzed as $em + bark$, but the Germanic prefix $em-$ is not productive in modern English, and thus appears in few pairs.

3. **Multistep Derivations**- The learner fails to learn multistep derivations, for example *acidified* as *ACID +ify +ed* if any intermediate derivations (*acidify*) are not present in the corpus.

4 Future Work

While the core concept of the base and transforms model stems from the sparse distribution of morphological variants of each word, the algorithm for learning the words to which a rule can be applied suffers at the hands of the same sparsity. For any multi-step derivation to be learned, all intermediate steps must be available to the learner. For example, if "hope" and "hopelessness" are observed in the corpus but "hopeless" is not, even though the transforms $(\$, less)$ and $(\$, ness)$ may be learned from other words in the corpus, they will not be applied to "hopelessness" because "hopeless" was not observed and the learner does not create unseen derived forms. While this limitation does not severely impact results in English where there are few morphemes per word, this significantly affects recall in other languages.

The definition of a transform given by Chan[1] and used unchanged here is expressive enough for English, German, and Romance languages where affixes are generally joined to words through simple deletion/concatenation processes but lacks the power to model morphological rules in languages that exhibit phenomena such as vowel harmony or morphology that is not suffix-based. It also lacks the ability to express other effects the attachment of the morpheme may have on the stem, such as vowel shortening. Improvements to the transform representation might include a set of operations that can be applied to the stem or suffixes as the transform is applied and some learned set of functions to account for morphophonemics.

A transform could also be defined as specifically inflectional or derivational. Extending the part of speech induction approach suggested by Chan in later chapters of [1] to cover the parts of speech of derived forms as well as bases would allow for scoping of rules to appropriate parts of speech for base and derived forms and thus an understanding of whether a rule is inflectional or derivational. With this information, it may be possible to learn constraints for combining rules, for example that the number of derivational rules used to derive a word is not strictly bounded, but the number of inflectional rules is.

References

[1] E. Chan. *Structures and distributions in morphology learning.* PhD thesis, University of Pennsylvania, 2008.

[2] J. Goldsmith. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27(2):153–198, 2001.

[3] Z.S. Harris. From phoneme to morpheme. *Language*, pages 190–222, 1955.

[4] S. Keshava and E. Pitler. A simpler, intuitive approach to morpheme induction. In *Proceedings of 2nd Pascal Challenges Workshop*, pages 31–35, 2006.

[5] S. Pinker. *Words and rules: The ingredients of language.* Basic Books, 1999.

[6] D.E. Rumelhart and J.L. McClelland. *Parallel distributed processing: explorations in the microstructure of cognition, vol. 2: psychological and biological models.* MIT Press Cambridge, MA, USA, 1986.