Extending the Use of Adaptor Grammars for Unsupervised Morphological Segmentation of Unseen Languages

Ramy Eskander* **Owen Rambow*** **Tianchun Yang†**
* Center for Computational Learning Systems
†Department of Computer Science
Columbia University
New York, NY, USA
{rnd2110@,rambow@ccls.,ty2313}columbia.edu

Abstract

We investigate using Adaptor Grammars for unsupervised morphological segmentation. Using six development languages, we investigate in detail different grammars, the use of morphological knowledge from outside sources, and the use of a cascaded architecture. Using cross-validation on our development languages, we propose a system which is language-independent. We show that it outperforms two state-of-the-art systems on 5 out of 6 languages.

1 Introduction

Morphological segmentation, the splitting of words into smaller units (morphs), is an important sub-task in several natural language processing (NLP) applications. With the increasing interest in NLP for low-resource languages, unsupervised morphological segmentation becomes a crucial pre-processing step to reduce data sparseness: instead of working on a large vocabulary of plausible words, a smaller set of smaller word units is processed.

A well-known toolkit used for unsupervised segmentation is Morfessor (Creutz and Lagus, 2007), which is a generative probabilistic model. In competition, Adaptor Grammars (AGs) (Johnson et al., 2007) represent a framework for specifying compositional nonparametric Bayesian models and are applied in unsupervised segmentation with notable success (Johnson, 2008).

AGs generalize probabilistic context-free grammars by allowing some nonterminals to be "adapted, which allows for dependencies between applications of these rules. Sirts and Goldwater (2013) present an in-depth investigation of the use of AGs. They make two important contributions. First, they discuss the effect of the underlying grammar on the results of unsupervised morphological segmentation. Second, they investigate two ways of using a small amount of annotated data during training. They show that while for English, Morfessor remains the top performing system, on three other languages their approach can beat the high Morfessor baseline.

A typical application for unsupervised morphological segmentation involves situations in which we are confronted with a low-resource language for which no prior NLP work exists, and for which we cannot annotate even a small corpus (either for lack of time, or because no annotators are available). Therefore, in this paper, we are interested in remaining language-independent and entirely unsupervised. We make the following contributions:

- We follow the insight of Sirts and Goldwater (2013) that the underlying grammar in an unsupervised AG approach matters. We explore a much larger set of grammars. We show that for most languages we develop on, the grammars we propose in this paper allow for the best AG results to date. The grammars are discussed in Section 4.

- We explore the use of "scholar-seeded knowledge". Here, instead of annotating even a small set with the desired result of the machine learning process (a segmentation), we search the web for easily accessible information about affixes in our language of interest, and explicitly include these in the grammar used in the AG approach (before learning happens). This can be done in a few hours by a scholar who has never studied the language. We show that generally scholar-seeded knowledge increases the performance. We discuss scholar-seeded knowledge in Section 5.

Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pages 900–910, Osaka, Japan, December 11-17 2016.

- We introduce an AG-based approach to approximate the effect of scholar-seeded knowledge: we use a high-precision AG to derive a set of affixes in a first round, and then we insert these into the AGs for the second round. We call this approach "cascaded adaptor grammars". We show that again, our approach generally improves performance. We discuss our cascaded architecture in Section 6.

- The best performing AG-based configuration differs from language to language. However, we would like to have a language-independent system which we can apply to unseen languages. Sirts and Goldwater (2013) solve this problem by using a hand-annotated tuning set to choose the best underlying grammar. We want to remain entirely unsupervised. Therefore, we perform a cross-validation on the six languages, using the five development languages of one fold to determine which grammar to apply to the held-out unseen language. We find that we always obtain the same cascade of the same two grammars for all languages if we do not consider scholar seeding. For the scholar-seeded approach we get a tie between two grammars. We use these cross-validation results to define our LIMS system, which is a language-independent morphological segmentation system, and show that it always outperforms Morfessor and on five out of six languages outperforms the best single AG of (Sirts and Goldwater, 2013). LIMS is our main contribution, and its performance on unseen languages in the cross-validation is our main result.

2 Related Work

Early research on unsupervised morphological segmentation was performed by extensive manual rule engineering, which was very expensive. With the advance of machine learning, minimum description length (MDL) based unsupervised approaches were applied for morphological segmentation in several languages (Goldsmith, 2001). However, this required extensive manual work, which was then replaced by maximum likelihood optimization (Creutz and Lagus, 2002).

Morfessor (Creutz and Lagus, 2007) is a commonly used system for unsupervised morphological segmentation. It is based on a generative probabilistic model. Because of its broad use, we use Morfessor as a reference system in this paper. Another system was developed by Poon et al. (2009), who apply classic log-linear models that use contextual and global features.

Nonparametric Bayesian methods with probabilistic grammars, such as Dirichlet process mixture models (see Antoniak (1974), Pitman (2002)) are widely used in unsupervised learning for NLP. Johnson et al. (2007) introduce nonparametric Bayesian models on whole tree structures, namely; Adaptor Grammars (AGs). AGs provide a flexible distribution over parse trees and are successfully applied in unsupervised segmentation (Johnson, 2008).

Sirts and Goldwater (2013) explore the use of AGs for minimally supervised morphological segmentation. They also compare the performance of different grammar trees. In this paper, we explore a much larger set of grammars, and simulate the performance of doing supervised morphological segmentation without the use of any annotated data or scholar-seeded knowledge.

Snyder and Barzilay (2008) propose a discriminative model for unsupervised morphological segmentation by using morphological chains to model the word formation process. A main drawback in their system is the slow learning curve, which requires a vast amount of data to learn from.

Wang et al. (2016) propose novel neural network architectures that learn the structure of input sequences directly from raw input words and are subsequently able to predict morphological boundaries. The architectures rely on Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997).

3 Problem Definition, Data, and Experimental Setup

The specific problem we are tackling is a segmentation of words in a language into a sequence of morphs. We do not rewrite or normalize morphs, we do not identify the stem, and we do not identify morphological features. For example, the English word *repayments* should be returned as *re pay ment s*.

We perform experiments on six languages, namely English, German, Finnish, Turkish, Zulu and Estonian. The data for English, German, Finnish, Turkish, and Estonian is the data from MorphoChallenge.[1]

[1] http://research.ics.aalto.fi/events/morphochallenge/

For Zulu, we used the Ukwabelana corpus (Spiegler et al., 2010). The sizes of our corpora are summarized in Table 1. Note that we reduced the German dev corpus to obtain only instances with true segmentation; therefore, our results are not comparable to other published results on this corpus. However, all comparisons we present in this paper are always based on the same train and dev corpora.

We use the Adaptor Grammar (AG) package available from Mark Johnson.[2] We run the experiments using nine grammars of different characteristics, which we present in detail in Section 4. All experiments are run using transductive learning, i.e., we include the evaluation data in the unsupervised learning along with the training data. The AG parameters are the same as the ones used by (Sirts and Goldwater, 2013) with 500 sampling iterations instead of 1,000 iterations; early experiments showed that the results of 500 iterations are nearly as good as 1,000 iterations, while fewer iterations decreased performance. We adapt all nonterminals except nontermials with recursive rules. For the AG results, we report the average of five different runs. We also run Morfessor2[3] as a baseline (Virpioja et al., 2013).

Language	Training	Dev
English	50,851	1,171
German	50,537	540
Finnish	51,305	1,031
Turkish	51,096	1,531
Zulu	50,597	1,000
Estonian	50,374	1,497

Table 1: Size of corpora (in types) of our development languages

We evaluate the segmentation against the DEV data from MorphoChallenge. Our evaluation metric is EMMA (Spiegler and Monson, 2010), which is based on morph recognition. EMMA has the advantage that it can return a meaningful result on unsegmented words (also see (Virpioja et al., 2011)).

4 Underlying Grammars

There are three fundamental dimensions in designing the grammars. The first dimension is how the grammar generates prefix, stem, and suffix. The first option is that a grammar does not explicitly model the division into prefix, stem, and suffix at all and only has morphs ("morph-only"); this is illustrated by Morph+SM in Figure 1. If we assume that we do want an explicit modeling of prefixes, stems, and suffixes, we have a "tripartite" grammar. The "tripartite" grammar is illustrated by PrStSu (tripartite, Figure 2). As an example, we give a schematic tree for the word *repayments*; we omit many details such as the handling of the beginning and end of word markers, the details of the recursion within nonterminals with plural names such as `PrefixMorphs` and `SuffixMorphs`, and the details of the generation of morphs through a recursive generation of characters. The plus signs in the trees are just for orientation, they are not actually generated.

A second dimension of modeling is the levels which are represented in the nonterminals. All grammars represent morphs. The tripartite grammars also explicitly model prefixes and suffixes. In addition, we follow Sirts and Goldwater (2013) in allowing morphs to be composed of submorphs; even when we distinguish prefix morphs from suffix morphs in a grammar, the submorphs are shared (as is the case in (Sirts and Goldwater, 2013)). A second option we take from Sirts and Goldwater (2013) is the possibility of having compounding, which is implemented as an iterated nonterminal immediately below the word level. It allows for the generation of compounds in which each compound element has its own prefix, stem and suffix (for example, German noun compounds such as *Ver+läng+er+ung+s+ge+such+e* 'requests for extension').

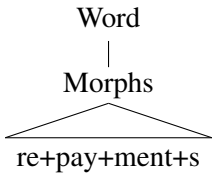

Figure 1: An analysis of *repayments* in Spine2+SM, a morph-only grammar. Submorphs are not shown.

A third dimension is the choice of the division into morphs for the output. If the grammar contains several levels of nonterminals (for example, compounds and morphs, or morphs and submorphs), then morph boundaries can be chosen at different levels (compound, morph, submorph). In the descriptions of our grammars below, we always list the level at which we define the morpheme boundary.

[2] `http://web.science.mq.edu.au/~mjohnson/Software.htm`
[3] `http://www.cis.hut.fi/projects/morpho/`

In developing our set of grammars, we started out with 44 grammars, which included the grammars used in (Sirts and Goldwater, 2013). Using EMMA F-measure as our criterion, we eliminated grammars which did not perform well across our languages, and we eliminated grammars which seemed to perform very similarly to other grammars. We ended up with nine grammars which we use for segmentation; one of these (PrStSu2b+Co+SM) was retained not because of its performance on F-measure, but on precision, which we only use for the first iteration in cascaded AG (which we will present in Section 6).[4] We now present our nine grammars.

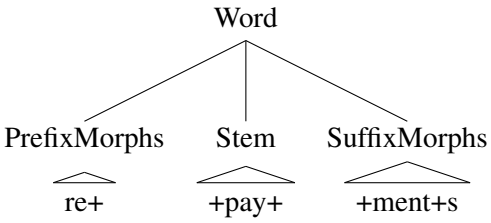

Figure 2: An analysis of *repayments* in PrStSu, a tripartite grammar. Submorphs are not shown.

We first have a morph-only grammar.

• Morph+SM: a word is recursively modeled as a sequence of morphs that consists of a lower level of submorphs, and the segmentation is based on the morph level. This grammar is grammar (2) (AG SubMorphs) from (Sirts and Goldwater, 2013), and we list it among our baselines.

We continue with tripartite grammars.

• Simple: a word is modeled as a sequence of an optional prefix, a stem, and an optional suffix, with no modeling of morphs beyond these three segments. The segmentation is based on the upper prefix, stem and suffix level.

• Simple+SM: the same as Simple with the introduction of a lower submorph level. The segmentation is still based on the prefix, stem and suffix level.

• PrStSu: a word is modeled as a prefix, stem and suffix sequence, where the prefix and suffix are sequences of zero or more morphs. The segmentation is based on the prefix, stem and suffix level. A sample analysis is shown in Figure 2. The segmentation is based on the prefix, stem and suffix level.

• PrStSu+SM: the same as PrStSu with the introduction of a lower submorph level shared by prefix and suffix morphs. The segmentation is based on the prefix morph, stem and suffix morph level.

• PrStSu+Co+SM: the same as PrStSu+SM with the introduction of an upper compound level. The segmentation is based on the prefix, stem and suffix level.

• PrStSu2a+SM: in contrast with PrStSu, where the prefix or suffix can simply be empty (but always exists in the derivation), we now allow the derivation to not have a prefix and/or a suffix. Specifically, a word is modeled as a stem-suffix sequence or a prefix and stem-suffix sequence, where the stem-suffix is a stem or a stem and a suffix. The prefix, stem and suffix are sequences of one or more morphs, with the introduction of a lower submorph level. The segmentation is based on the prefix, stem and suffix level. This grammar is an implementation of grammar (3) (AG Compounding) from (Sirts and Goldwater, 2013) without the compounding (since the compounding did not perform well for our languages), and we list it among our baselines in the result table.

• PrStSu2b+SM: this grammar is similar to PrStSu2a+SM but instead of modeling the the word as a prefix and stem-suffix sequence it is modeled reversely as a prefix-stem and suffix sequence.

• PrStSu2b+Co+SM: the same as PrStSu2b+SM but the upper compound level is added.

The results of the adaptor grammar experiments for our six development languages, English, German, Finnish, Turkish, Zulu and Estonian are in Table 2. Some observations:

- There is vast variation among languages in how grammars perform and which grammar is best.
- One of the grammars always beats the Morfessor baseline, and we always beat the AG baselines of Sirts and Goldwater (2013) except for Finnish. Some of the margins are small and probably not statistically significant.
- Grammar PrStSu2b+Co+SM, which has three levels of morphological representation (compounds, morphs, and submorphs) has very low recall across all languages, perhaps because the resulting morphs are too small (i.e., longer morphs are not predicted).

[4]The grammars are available at `http://www.cs.columbia.edu/~rambow/ag/ag.html` .

	English			German			Finnish		
Grammar	**Prec.**	**Rec.**	**F1**	**Prec.**	**Rec.**	**F1**	**Prec.**	**Rec.**	**F1**
Morfessor	79.7	81.4	80.5	79.0	69.6	74.0	74.5	61.8	67.5
Morph+SM	84.3	77.6	80.8	80.5	66.2	72.6	76.8	59.7	67.2
PrStSu2a+SM	75.9	80.3	78.0	80.1	72.6	76.2	74.2	**68.8**	**71.4**
Simple	64.5	72.1	68.1	71.7	67.5	69.6	69.6	61.6	65.3
Simple+SM	78.6	73.1	75.7	81.8	69.0	74.9	77.8	57.1	65.9
PrStSu	71.0	77.5	74.1	72.7	68.4	70.5	69.8	51.4	59.2
PrStSu+SM	81.2	**83.1**	82.1	81.3	**76.8**	**79.0**	66.6	59.8	63.0
PrStSu+Co+SM	89.7	76.5	**82.6**	**82.4**	61.6	70.5	81.5	58.3	68.0
PrStSu2b+SM	60.9	75.6	67.5	75.8	74.5	75.2	64.0	60.6	62.2
PrStSu2b+Co+SM	**92.4**	50.2	65.0	78.9	39.4	52.5	**93.0**	42.5	58.3
	Turkish			Zulu			Estonian		
Grammar	**Prec.**	**Rec.**	**F1**	**Prec.**	**Rec.**	**F1**	**Prec.**	**Rec.**	**F1**
Morfessor	67.4	46.6	55.1	53.4	33.8	41.4	75.0	81.0	77.9
Morph+SM	69.6	43.0	53.2	59.3	36.2	45.0	81.5	83.8	82.6
PrStSu2a+SM	75.8	55.1	63.8	52.2	42.1	46.6	66.8	82.7	73.9
Simple	64.8	45.7	53.6	59.4	49.5	54.0	65.5	78.9	71.6
Simple+SM	69.4	41.4	51.9	58.6	37.2	45.5	79.9	83.6	81.7
PrStSu	68.5	45.7	54.8	68.8	48.0	56.5	72.0	80.7	76.1
PrStSu+SM	77.8	**55.4**	**64.7**	57.4	43.5	49.5	65.3	81.1	72.3
PrStSu+Co+SM	71.1	40.6	51.7	59.9	34.7	43.9	84.8	**83.9**	**84.3**
PrStSu2b+SM	55.3	45.5	49.9	**72.0**	**51.5**	**60.1**	56.9	77.4	65.6
PrStSu2b+Co+SM	**82.3**	25.3	38.8	63.9	19.9	30.3	**93.9**	59.0	72.5

Table 2: Adaptor-grammar results (percent Emma) for English, German, and Finnish (above) and Turkish, Zulu, and Estonian (below). The best result per column is highlighted in boldface. The first three rows are baselines, and the next seven rows are tripartite grammars.

5 Scholar-Seeded Knowledge

The intuition behind the use of scholar-seeded knowledge is that for many languages, we have more or less extensive descriptions of their morphology. In fact, traditional descriptive grammars often concentrate on morphology. Today, a lot of information is available online. These resources provide lists of affixes, often in the form of paradigms or tables. Typically, only a very small number of lexemes are used to illustrate the morphology, or the affixes are simply listed without stems. Thus, the data is not a representative (type or token) sample of actual words in the language. We investigate the question of whether this data can be used in unsupervised segmentation. We note that this data is not "data" in the normal sense of machine learning: it is not in the same format as the desired output (i.e., segmented words). Therefore, this is not a case of semi-supervised machine learning, as Sirts and Goldwater (2013) explore.

Adaptor grammars is a framework that is particularly well suited for applying scholar-seeded knowledge as AG takes as input a hand-crafted grammar. Into this grammar, we can explicitly insert the affixes we have gleaned from the literature and from online sources. In Section 4, we investigated nine different grammars. We can insert the same affixes into all of these grammars in the position where morphs are generated. Of course, we continue to allow the grammars to generate new morphs, as we do not expect the sources to contain complete lists.

For these experiments, we consulted only online resources. We spent about two hours per language, and assembled between 30 and 120 affixes.

The results are shown in Table 3. We only show F-measure, and repeat the result for each grammar

Grammar	English			German			Finnish		
	Std.	**Sch.**	**Casc.**	**Std.**	**Sch.**	**Casc.**	**Std.**	**Sch.**	**Casc.**
Morfessor	80.5			74.0			67.5		
Morph+SM	**80.8**	75.2	75.3	72.6	**74.0**	73.1	**67.2**	63.9	63.3
PrStSu2a+SM	78.0	77.8	**79.6**	76.2	**77.3**	76.8	71.4	72.4	**73.3**
Simple	**68.1**	65.1	64.6	**69.6**	61.0	61.4	**65.3**	59.6	58.9
Simple+SM	**75.7**	72.2	72.0	**74.9**	68.8	68.9	**65.9**	60.8	60.8
PrStSu	**74.1**	64.2	64.5	**70.5**	67.3	67.6	59.2	60.3	**62.3**
PrStSu+SM	**82.1**	81.8	80.9	79.0	**79.3**	77.7	63.0	**72.9**	72.7
PrStSu+Co+SM	**82.6**	80.8	78.2	**70.5**	67.9	66.9	**68.0**	64.5	63.8
PrStSu2b+SM	67.5	69.1	**70.0**	75.2	76.6	**76.7**	62.2	64.9	**65.2**
PrStSu2b+Co+SM	65.0	**65.1**	65.0	52.5	52.6	**53.0**	58.3	58.4	**58.9**

Grammar	Turkish			Zulu			Estonian		
	Std.	**Sch.**	**Casc.**	**Std.**	**Sch.**	**Casc.**	**Std.**	**Sch.**	**Casc.**
Morfessor	55.1			41.4			77.9		
Morph+SM	53.2	54.5	**55.9**	45.0	47.6	**49.0**	**82.6**	71.2	71.1
PrStSu2a+SM	**63.8**	63.4	57.2	46.6	**56.5**	47.2	73.9	82.3	**82.8**
Simple	**53.6**	50.7	44.7	**54.0**	41.5	41.6	**71.6**	69.9	69.9
Simple+SM	**51.9**	51.0	49.3	**45.5**	44.1	44.1	**81.7**	77.3	77.3
PrStSu	**54.8**	**54.8**	51.4	**56.5**	46.4	46.1	**76.1**	70.2	69.5
PrStSu+SM	**64.7**	51.2	59.1	49.5	**65.7**	61.1	72.3	80.4	**80.5**
PrStSu+Co+SM	51.7	**52.3**	49.1	43.9	44.0	**44.1**	84.3	**84.4**	77.3
PrStSu2b+SM	49.9	**51.3**	51.0	**60.1**	58.2	47.7	65.6	**66.3**	66.2
PrStSu2b+Co+SM	38.8	39.5	**40.1**	30.3	**33.9**	30.3	72.5	**72.7**	**72.7**

Table 3: Adaptor-grammar results (percent Emma F-measure) for English, German, and Finnish (above) and Turkish, Zulu, and Estonian (below) for standard (Std; repeated from Table 2), scholar-seeded (Sch), and cascaded approaches (Casc). Boldface indicates best result by language for that grammar. The first three rows are baselines, and the next seven rows are tripartite grammars.

from Table 2 in the first column of each language. The second column shows the scholar-seeded result. (We discuss the third column in Section 6.) As we can see, the seeding of the grammars with some initial morphs does not, in general, improve our results. We provide a more detailed discussion at the end of the next section.

6 Cascaded Adaptor Grammars

In this approach, we investigate whether we can find the list of affixes which we use in scholar-seeded knowledge automatically, using AGs themselves. The basic approach is as follows:

1. We choose a grammar that has a high precision according to Emma across our development languages. The reason to choose a high precision (rather than a high F-measure) is that we want to be certain of having true affixes in the grammar, rather than having as many affixes as possible (even if some are not correct). We choose grammar PrStSu2b+Co+SM, which achieves the highest precision of all our grammars for English, Finnish, Turkish and Estonian, and close to highest for German. Only for Zulu is the precision mediocre. However, we want to choose a single grammar independently of the language, as we do not want to tune our approach to the language (we have no annotated tuning set).

2. We use this grammar to run AGs in a first iteration, and identify a list of prefixes and suffixes. We order the affixes by frequency.

English: +s, +ed, +ing, +e, +'s, re+, +es, +er, +y, a+, de+, +a, in+, co+, +ers, **ma+**, **ca+**, **se+**, **u+**, **e+**, **n+**, con+, **pa+**, +ly, **o+**, **ra+**, **+o**, **la+**, **ro+**, **ha+**, **ba+**, **mo+**, **ho+**, di+, +s', +ion, pro+, **sa+**, be+, **po+**
German: +en, +e, +er, +t, +s, ver+, be+, ge+, +ung, +es, +te, er+, +ten, +n, an+, ein+, aus+, ab+, +ungen, un+, vor+, zu+, +a, re+, ueber+, ent+, **s+**, +ischen, auf+, +et, +ern, +lich, +in, **+-**, unter+, +ische, **ma+**, +o, +ende, +enden

Figure 3: English and German affixes produced by our cascaded AG, shown in order of frequency in the corpus; incorrect affixes are in **boldface**.

3. We then include the top n affixes from our list in the grammars, in the same way we do for scholar-seeded knowledge. We run AG again, in a second iteration, using these modified grammars. We perform experiments on all our languages with $n = 10, 20, 30, 40, 50, 100$, which we refer to as Cascade$n$.

For example, grammar PrStSu2b+Co+SM finds the prefixes and suffixes for English and German shown in Figure 3; we list the 40 most common affixes (not including the empty prefix and the empty suffix, which are also generated). We show incorrect affixes in boldface. As we can see, the 15 most frequent affixes found for English are indeed correct affixes of English, but among the subsequent 25 most frequent affixes, only seven are correct.[5] In contrast, for German, a language with much richer inflectional and derivational morphology, all but three affixes are correct among the top 40 (and the top 25 are all correct).

The results for Cascade40 (i.e., using the top 40 affixes from the first iteration in the second iteration) are shown in the third column for each language in Table 3. We chose $n = 40$ since it has the best performance across all languages.

We now jointly discuss the results for the scholar-seeded approach (column 2 in Table 3, Section 5) and the cascaded approach (column 3, this section).

- If we simply look for the language-specific best performance, we see that the best score is achieved by a Standard AG for English and Turkish; by a scholar-seeded AG for German, Zulu, and Estonian; and by a cascaded AG for Finnish.

- The cascaded approach in general achieves results that are comparable to the scholar-seeded approach, i.e., we have shown that we can use AGs to provide information that is equivalent for the AG to what we can obtain from scholarly sources and the Internet.

- The scholar-seeded and cascaded approaches outperform the basic AG approach for some languages and some of our grammars. However, many grammars do not profit from scholar seeding or cascading. For example, for grammars PrStSu+SM and PrStSu+Co+SM, for all six languages the best configuration is the simple AG.

- Only for German and Zulu is the best performing configuration exactly the same; each of the other languages has a different configuration as its best performing. We thus do not see a cross-linguistic generalization emerge readily from Table 3.

If we were interested in optimizing performance for each language separately (i.e., by using the development set on which we are reporting results as a tuning set), then we would be done now. However, if we want to optimize our result across languages (as we do in this paper), we need to look more closely at the results, which we do in the next section.

[5]We consulted the Wiktionary pages for English affixes (`https://en.wiktionary.org/wiki/Category:English_prefixes` and `https://en.wiktionary.org/wiki/Category:English_suffixes`), but discarded some affixes which we did not feel were relevant (such and $n+$ as a typographical variant of μ). For German, we used our native speaker knowledge, as the corresponding Wiktionary pages do not contain all inflected affixes.

7 Finding the Optimal Language-Independent System

So far, we have discussed different approaches and presented results only on the development sets. In this paper, we are not interested in maximizing a single language-specific system; instead, we are interested in finding a single system which will perform well on an entirely unseen language (for which we have no annotation at all). To do this, we perform leave-one-out cross validation *on languages*. In each of the six folds of the cross validation, we choose one language in turn as the test language. We average the results for the other five languages (which become our development languages in this fold) for all grammars, for Standard, Cascaded, and Scholar-Seeded. We are interested in two configurations: no use of scholar-seeded knowledge (Standard or Cascaded), and inclusion of scholar-seeded knowledge (Scholar-Seeded). For each of these two configurations, we determine which grammar performs best on average across the five development languages of the fold. We then apply this grammar to the held-out test language. This means that for the held-out language, the choice of grammar was not in any way influenced by any observation from that language.[6] We first describe which grammars we choose in this manner.

- For the configuration without scholar-seeded knowledge, the cross-validation results are shown in table 4. We find that the best performing system for all six averages across five development languages (with the sixth language being held out) is a cascade, starting with PrStSu2b+Co+SM, and then inserting the obtained affixes into PrStSu+SM and running this modified PrStSu+SM. We will call this cascade of AGs LIMS, for Language-Independent Morphological Segmenter.

- For the configuration with scholar-seeded knowledge, the cross-validation results are shown in table 5. We find that the best performing system is split. For three held-out languages (Finnish, Turkish, and Estonian), the best performing grammar for the average of the other five languages is PrStSu+SM (augmented with scholar-seeded affixes), while for the other three held-out languages (English, German, and Zulu), it is PrStSu2a+SM, as shown in table 5. Since we need to choose a single configuration, we (arbitrarily) choose the grammar which we have already chosen for the configuration without scholar-seeded knowledge, namely PrStSu+SM, and we call this system LIMS-Scholar.

Held-out Language (HL)	**Best Grammar for {All - HL} (=G)**	**Ave. F-Score of G on {All - HL}**	**F-Score of G on HL**	**Oracle**
English	PrStSu+SM	70.2%	80.9%	82.6%
German	PrStSu+SM	70.8%	77.7%	79.0%
Finnish	PrStSu+SM	71.9%	72.7%	73.2%
Turkish	PrStSu+SM	74.6%	59.1%	64.7%
Zulu	PrStSu+SM	74.2%	61.1%	61.1%
Estonian	PrStSu+SM	70.3%	80.5%	84.3%

Table 4: Leave-one-out cross-validation results with no use of scholar-seeded knowledge. The Oracle result is the best performing configuration not using scholar-seeded knowledge on the held-out language, as shown in Table 3.

We now have two segmentation systems. LIMS is a black box system which can be applied to any language. LIMS-Scholar is a system which requires input in the form of a list of affixes. Clearly, its performance depends on the list of affixes provided. We present experimental results for these two systems in Table 6. The top two rows are baselines: the Morfessor system used out of the box (Morfessor2[7]), and grammar Morph+SM, which is the same as the AG SubMorphs grammar of Sirts and Goldwater (2013),

[6] We acknowledge that early elimination of additional grammars, not discussed in this paper, was in fact done by considering all languages.

[7] http://www.cis.hut.fi/projects/morpho/

Held-out Language (HL)	Best Grammar for {All - HL} (=G)	Ave. F-Score of G on {All - HL}	F-Score of G on HL	Oracle
English	PrStSu2a+SM	70.4%	77.8%	81.8%
German	PrStSu+SM	70.5%	77.3%	79.3%
Finnish	PrStSu2a+SM	71.7%	72.9%	72.9%
Turkish	PrStSu+SM	76.0%	51.2%	63.4%
Zulu	PrStSu2a+SM	74.6%	56.5%	65.7%
Estonian	PrStSu+SM	70.2%	84.4%	84.4%

Table 5: Leave-one-out cross-validation results with scholar-seeded knowledge. The Oracle result is the best performing configuration using scholar-seeded knowledge on the held-out language, as shown in Table 3.

which obtains the best average across their five development languages (our six development languages but not Zulu). We use our reimplementation of the grammar, and the results we obtain with this grammar are somewhat better than the results published in (Sirts and Goldwater, 2013), probably because of slightly different parameter settings. (Recall that our German data is different from the German data used in (Sirts and Goldwater, 2013).) The following two rows are our two systems. Since LIMS is always the same system, this row is the same as the penultimate column in Table!4. And the last two rows are the best results we obtained for that language across all of our grammars, without and with scholar seeded knowledge. We note that these last two rows are oracle experiments in the sense that we observe all of our results and choose the best configuration. However, if (for some odd reason) we wanted to perform unsupervised segmentation of words in one of our development languages, we would use these systems in the bottom two rows, though we have not demonstrated their performance on unseen test data in those languages (because that is not the goal of this paper).

System	English	German	Finnish	Turkish	Zulu	Estonian	Avg.
Morfessor	80.5%	74.0%	67.5%	55.1%	41.4%	77.9%	66.1%
Morph+SM	80.8%	72.6%	67.2%	53.2%	45.0%	**82.6%**	66.9%
LIMS	80.9%	77.7%	72.7%	**59.1%**	61.1%	80.5%	**72.0%**
LIMS-Scholar	**81.8%**	**79.3%**	**72.9%**	51.2%	**65.7%**	80.4%	71.9%
Best Standard/Cascaded	82.6%	79.0%	73.3%	64.7%	61.1%	84.7%	74.2%
Best Scholar-Seeded	82.1%	79.3%	72.9%	63.4%	65.7%	84.4%	74.6%

Table 6: A comparison between our systems and two baselines: Morfessor and Morph+SM (= AG SubMorphs of (Sirts and Goldwater, 2013)). The two best performing grammars from our experiments are also shown as an oracle result. The best result among the baselines and our systems is boldfaced.

Table 6 shows that for all of our held-out development languages except Turkish and Estonian, LIMS and LIMS-Scholar both outperforms both Morfessor and AG SubMorphs. For Turkish, LIMS outperforms both baselines, but LIMS-scholar outperforms neither. For Estonian, both LIMS and LIMS-scholar outperform Morfessor, while AG SubMorphs performs better than either of our systems. Furthermore, we see that in four of the six held-out languages, LIMS-Scholar outperforms LIMS, but on average LIMS slightly outperforms LIMS-Scholar because of the poor performance on Turkish. We note again that the performance of LIMS-Scholar depends on the quality of the scholar-seeded knowledge, so that one should be cautious with drawing conclusions. However, it appears overall that the cascaded approach of LIMS is a perfectly adequate alternative to using the scholar-seeded knowledge required for LIMS-Scholar.

8 Conclusion and Future Work

We have investigated the issue of unsupervised segmentation using Adaptor Grammars. Unlike recent work, we remain entirely unsupervised, i.e., we do not assume that we have some data which has been annotated by hand. We have experimented with different forms of grammars, the notion of seeding the grammar with knowledge obtained from scholarly publications and the web, and an architecture in which we obtain an equivalent amount of information using an AG, resulting in a cascaded architecture.

In this paper, we are not interested in maximizing performance on our development languages individually, and therefore we have not presented results on held-out test sets for the development languages. Instead, we have performed a cross-validation experiment on languages. We determine the best configuration to apply to the unseen language using only the other development languages, not the unseen language itself. We obtain two systems, which we call LIMS and LIMS-Scholar, with the latter using scholar-seeded knowledge. We show that LIMS outperforms Morfessor on all languages (LIMS-scholar on all but one language), and LIMS outperforms the previous best Adaptor Grammar results on all but one language (LIMS-scholar on all but two languages).

In future work, we intend to perform an extrinsic evaluation, in which an outside task, which requires morphological segmentation in its input, will be used to compare different settings. We will also investigate whether we can estimate the number of affixes to use in the cascaded approach; Figure 3 shows that choosing the top 40 affixes for all languages is not a good choice. Furthermore, we would like to determine in an entirely unsupervised manner the best underlying grammar for a language. This is an appealing goal as the best performance for a language is often superior to that obtained by LIMS or LIMS-Scholar.

Acknowledgments

We thank Kairit Sirts for her insights and for sharing her grammar trees, which we used as a baseline, and some of the data. We also thank three anonymous reviewers for their helpful comments. This work is supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense U.S. Army Research Laboratory (DoD/ARL) contract number W911NF-12-C-0012. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government.

References

Charles E. Antoniak. 1974. Mixtures of dirichlet processes with applications to bayesian nonparametric problems. *The Annals of Statistics*, 2(6):152–1174.

Mathias Creutz and Krista Lagus. 2002. Unsupervised discovery of morphemes. In *Proceedings of the ACL-02 workshop on Morphological and phonological learning-Volume 6*, pages 21–30. Association for Computational Linguistics.

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Trans. Speech Lang. Process.*, 4(1):3:1–3:34, February.

John Goldsmith. 2001. Unsupervised learning of the morphology of a natural language. *Computational linguistics*, 27(2):153–198.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. 2007. Adaptor grammars: a framework for specifying compositional nonparametric bayesian models. In B. Sch"olkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 641–648, Cambridge, MA. MIT Press.

Mark Johnson. 2008. Unsupervised word segmentation for Sesotho using adaptor grammars. In *Proceedings of the Tenth Meeting of ACL Special Interest Group on Computational Morphology and Phonology*, pages 20–27, Columbus, Ohio, June. Association for Computational Linguistics.

Jim Pitman. 2002. Combinatorial stochastic processes. *Lecture Notes for St. Flour Summer School*.

Hoifung Poon, Colin Cherry, and Kristina Toutanova. 2009. Unsupervised morphological segmentation with log-linear models. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 209–217, Boulder, Colorado, June. Association for Computational Linguistics.

Kairit Sirts and Sharon Goldwater. 2013. Minimally-supervised morphological segmentation using adaptor grammars. *Transactions of the Association for Computational Linguistics*, 1(May):231–242.

Benjamin Snyder and Regina Barzilay. 2008. Unsupervised multilingual learning for morphological segmentation. In *Proceedings of ACL-08: HLT*, pages 737–745, Columbus, Ohio, June. Association for Computational Linguistics.

Sebastian Spiegler and Christian Monson. 2010. Emma: A novel evaluation metric for morphological analysis. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1029–1037, Beijing, China, August. Coling 2010 Organizing Committee.

Sebastian Spiegler, Andrew van der Spuy, and Peter A. Flach. 2010. Ukwabelana - an open-source morphological zulu corpus. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1020–1028, Beijing, China, August. Coling 2010 Organizing Committee.

Sami Virpioja, Ville T. Turunen, Sebastian Spiegler, Oskar Kohonen, and Mikko Kurimo. 2011. Empirical comparison of evaluation methods for unsupervised learning of morphology. *Traitement Automatique des Langues*, 52(2):45–90.

Sami Virpioja, Peter Smit, Stig-Arne Grnroos, and Mikko Kurimo. 2013. Morfessor 2.0: Python implementation and extensions for morfessor baseline. Technical report, Aalto University.

Linlin Wang, Zhu Cao, Yu Xia, and Gerard de Melo. 2016. Morphological segmentation with window LSTM neural networks. In *Thirtieth AAAI Conference on Artificial Intelligence*.