Minimally-Supervised Morphological Segmentation using Adaptor Grammars

Kairit Sirts
Institute of Cybernetics
Tallinn University of Technology
sirts@phon.ioc.ee

Sharon Goldwater
School of Informatics
The University of Edinburgh
sgwater@inf.ed.ac.uk

Abstract

This paper explores the use of Adaptor Grammars, a nonparametric Bayesian modelling framework, for minimally supervised morphological segmentation. We compare three training methods: unsupervised training, semi-supervised training, and a novel model selection method. In the model selection method, we train unsupervised Adaptor Grammars using an over-articulated *metagrammar*, then use a small labelled data set to select which potential morph boundaries identified by the metagrammar should be returned in the final output. We evaluate on five languages and show that semi-supervised training provides a boost over unsupervised training, while the model selection method yields the best average results over all languages and is competitive with state-of-the-art semi-supervised systems. Moreover, this method provides the potential to tune performance according to different evaluation metrics or downstream tasks.

1 Introduction

Research into unsupervised learning of morphology has a long history, starting with the work of Harris (1951). While early research was mostly motivated by linguistic interests, more recent work in NLP often aims to reduce data sparsity in morphologically rich languages for tasks such as automatic speech recognition, statistical machine translation, or automatic text generation. For these applications, however, completely unsupervised systems may not be ideal if even a small amount of segmented training data is available. In this paper, we explore the use of Adaptor Grammars (Johnson et al., 2007) for minimally supervised morphological segmentation.

Adaptor Grammars (AGs) are a nonparametric Bayesian modelling framework that can learn latent tree structures over an input corpus of strings. For example, they can be used to define a morphological grammar where each word consists of zero or more prefixes, a stem, and zero or more suffixes; the actual forms of these morphs (and the segmentation of words into morphs) are learned from the data. In this general approach AGs are similar to many other unsupervised morphological segmentation systems, such as Linguistica (Goldsmith, 2001) and the Morfessor family (Creutz and Lagus, 2007). A major difference, however, is that the morphological grammar is specified as an input to the program, rather than hard-coded, which allows different grammars to be explored easily. For the task of segmenting utterances into words, for example, Johnson and colleagues have experimented with grammars encoding different kinds of sub-word and super-word structure (e.g., syllables and collocations), showing that the best grammars far outperform other systems on the same corpora (Johnson, 2008a; Johnson and Goldwater, 2009; Johnson and Demuth, 2010).

These word segmentation papers demonstrated both the power of the AG approach and the synergistic behavior that occurs when learning multiple levels of structure simultaneously. However, the best-performing grammars were selected using the same corpus that was used for final testing, and each paper dealt with only one language. The ideal unsupervised learner would use a single grammar tuned on

Transactions of the Association for Computational Linguistics, 1 (2013) 255–266. Action Editor: Kristina Toutanova.
Submitted 11/2012; Published 5/2013.

one or more development languages and still perform well on other languages where development data is unavailable. Indeed, this is the basic principle behind Linguistica and Morfessor. However, we know that different languages can have very different morphological properties, so using a single grammar for all languages may not be the best approach if there is a principled way to choose between grammars. Though AGs make it easy to try many different possible grammars, the process of proposing and testing plausible options can still be time-consuming.

In this paper, we propose a novel method for automatically selecting good morphological grammars for different languages (English, Finnish, Turkish, German, and Estonian) using a small amount of gold-segmented data (1000 word types). We use the AG framework to specify a very general binary-branching grammar of depth four with which we learn a parse tree of each word that contains several possible segmentation splits for the word. Then, we use the gold-segmented data to learn, for each language, which of the proposed splits from the original grammar should actually be used in order to best segment that language.

We evaluate our approach on both a small development set and the full Morpho Challenge test set for each language—up to three million word types. In doing so, we demonstrate that using the posterior grammar of an AG model to decode unseen data is a feasible way to scale these models to large data sets. We compare to several baselines which use the annotated data to different degrees: parameter tuning, grammar tuning, supervised training, or no use of annotated data. In addition to existing approaches—unsupervised and semi-supervised Morfessor, unsupervised Morsel (Lignos, 2010), and unsupervised AGs—we also show how to use the annotated data to train semi-supervised AGs (using the data to accumulate rule statistics rather than for grammar selection). The grammar selection method yields comparable results to the best of these other approaches.

To summarize, our contributions in this paper are: 1) scaling AGs to large data sets by using the posterior grammar to define an inductive model; 2) demonstrating how to train semi-supervised AG models, and showing that this improves morphological segmentation over unsupervised training; and 3) introducing a novel grammar selection method for AG models whose segmentation results are competitive with the best existing systems.

Before providing details of our methods and results, we first briefly review Adaptor Grammars. For a formal definition, see Johnson et al. (2007).

2 Adaptor Grammars

Adaptor Grammars are a framework for specifying nonparametric Bayesian models that can be used to learn latent tree structures from a corpus of strings. There are two components to an AG model: the *base distribution*, which is just a PCFG, and the *adaptor*, which "adapts" the probabilities assigned to individual subtrees under the PCFG model, such that the probability of a subtree under the complete model may be considerably higher than the product of the probabilities of the PCFG rules required to construct it. Although in principle the adaptor can be any function that maps one distribution onto another, Johnson et al. (2007) use a Pitman-Yor Process (PYP) (Pitman and Yor, 1997) as the adaptor because it acts as a caching model. Under a PYP AG model, the posterior probability of a particular subtree will be roughly proportional to the number of times that subtree occurs in the current analysis of the data (with the probability of unseen subtrees being computed under the base PCFG distribution).

An AG model can be defined by specifying the CFG rules (the support for the base distribution) and indicating which non-terminals are "adapted", i.e., can serve as the root of a cached subtree. Given this definition and an input corpus of strings, Markov chain Monte Carlo samplers can be used to infer the posterior distribution over trees (and all hyperparameters of the model, including PCFG probabilities in the base distribution and PYP hyperparameters). Any frequently recurring substring (e.g., a common prefix) will tend to be parsed consistently, as this permits the model to treat the subtree spanning that string as a cached subtree, assigning it higher probability than under the PCFG distribution.

Adaptor Grammars have been applied to a wide variety of tasks, including segmenting utterances into words (Johnson, 2008a; Johnson and Goldwater, 2009; Johnson and Demuth, 2010), classifying documents according to perspective (Hardisty et al., 2010), machine transliteration of names (Huang et

al., 2011), native language identification (Wong et al., 2012), and named entity clustering (Elsner et al., 2009). There have also been AG experiments with morphological segmentation, but more as a proof of concept than an attempt to achieve state-of-the-art results (Johnson et al., 2007; Johnson, 2008b).

3 Using AGs for Learning Morphology

Originally, the AG framework was designed for unsupervised learning. This section first describes how AGs can be used for unsupervised morphological segmentation, and then introduces two ways to use a small labelled data set to improve performance: semi-supervised learning and grammar selection.

3.1 Unsupervised Adaptor Grammars

We define three AG models to use as unsupervised baselines in our segmentation experiments. The first of these is very simple:

$$\begin{aligned} \text{Word} &\rightarrow \underline{\text{Morph}}^{+} \\ \underline{\text{Morph}} &\rightarrow \text{Char}^{+} \end{aligned} \tag{1}$$

The underline notation indicates an adapted non-terminal, and $^{+}$ abbreviates a set of recursive rules, e.g., Word $\rightarrow \underline{\text{Morph}}^{+}$ is short for

$$\begin{aligned} \text{Word} &\rightarrow \text{Morphs} \\ \text{Morphs} &\rightarrow \underline{\text{Morph}}\ \text{Morphs} \\ \text{Morphs} &\rightarrow \underline{\text{Morph}} \end{aligned}$$

Grammar 1 (**MorphSeq**) is just a unigram model over morphs: the Morph symbol is adapted, so the probability of each Morph will be roughly proportional to its (inferred) frequency in the corpus. The grammar specifies no further structural relationships between morphs or inside of morphs (other than a geometric distribution on their length in characters).

Experiments with AGs for unsupervised word segmentation suggest that adding further latent structure can help with learning. Here, we add another layer of structure below the morphs,[1] calling the resulting grammar **SubMorphs**:

$$\begin{aligned} \text{Word} &\rightarrow \underline{\text{Morph}}^{+} \\ \underline{\text{Morph}} &\rightarrow \underline{\text{SubMorph}}^{+} \\ \underline{\text{SubMorph}} &\rightarrow \text{Char}^{+} \end{aligned} \tag{2}$$

For capturing the rules of morphotactics, a grammar with linguistically motivated non-terminals can be created. There are many plausible options and the best-performing grammar may be somewhat language-dependent. Rather than experimenting extensively, we designed our third grammar to replicate as closely as possible the grammar that is implicitly implemented in the Morfessor system. This **Compounding** grammar distinguishes between prefixes, stems and suffixes, allows compounding, defines the order in which the morphs can occur and also allows the morphs to have inner latent structure:

$$\begin{aligned} \text{Word} &\rightarrow \underline{\text{Compound}}^{+} \\ \underline{\text{Compound}} &\rightarrow \underline{\text{Prefix}}^{*}\ \underline{\text{Stem}}\ \underline{\text{Suffix}}^{*} \\ \underline{\text{Prefix}} &\rightarrow \underline{\text{SubMorph}}^{+} \\ \underline{\text{Stem}} &\rightarrow \underline{\text{SubMorph}}^{+} \\ \underline{\text{Suffix}} &\rightarrow \underline{\text{SubMorph}}^{+} \\ \underline{\text{SubMorph}} &\rightarrow \text{Char}^{+} \end{aligned} \tag{3}$$

3.2 Semi-Supervised Adaptor Grammars

The first new use of AGs we introduce is the *semi-supervised* AG, where we use the labelled data to extract counts of the different rules and subtrees used in the gold-standard analyses. We then run the MCMC sampler as usual over both the unlabelled and labelled data, treating the counts from the labelled data as fixed.

We assume that the labelled data provides a consistent bracketing (no two spans in the bracketing can partially overlap) and the labels of the spans must be compatible with the grammar. However, the bracketing may not specify all levels of structure in the grammar. In our case, we have morpheme bracketings but not, e.g., submorphs. Thus, using the SubMorphs grammar in semi-supervised mode will constrain the sampler so that Morph spans in the labelled data will remain fixed, while the SubMorphs inside those Morphs will be resampled.

[1] Because the nonterminal labels are arbitrary, this grammar can also be interpreted as adding another layer on top of morphs, allowing the model to learn morph *collocations* that encode dependencies between morphs (which themselves have no substructure). However preliminary experiments showed that the morph-submorph interpretation scored better than the collocation-morph interpretation, hence we chose the corresponding nonterminal names.

The main change made to the AG inference process[2] for implementing the semi-supervised AG was to prune out from the sampling distribution any non-terminals that are inconsistent with the spans/labels in the given labelling.

3.3 AG Select

Both the unsupervised and semi-supervised methods described above assume the definition of a grammar that adequately captures the phenomena being modelled. Although the AG framework makes it easy to experiment with different grammars, these experiments can be time-consuming and require some good guesses as to what a plausible grammar might be. These problems can be overcome by automating the grammar development process to systematically evaluate different grammars and find the best one.

We propose a minimally supervised model selection method AG Select that uses the AG framework to automatically identify the best grammar for different languages and data sets. We first define a very general binary-branching CFG grammar for AG training that we call the *metagrammar*. The metagrammar learns a parse tree for each word where each branch contains a different structure in the word. The granularity of these structures is determined by the depth of this tree. For example, Grammar 4 generates binary trees of depth two and can learn segmentations of up to four segments.

$$
\begin{array}{ll}
\text{Word} \rightarrow \underline{\text{M1}} & \\
\text{Word} \rightarrow \underline{\text{M1}}\ \underline{\text{M2}} & \underline{\text{M11}} \rightarrow \text{Chars}^{+} \\
\underline{\text{M1}} \rightarrow \underline{\text{M11}} & \underline{\text{M12}} \rightarrow \text{Chars}^{+} \\
\underline{\text{M1}} \rightarrow \underline{\text{M11}}\ \underline{\text{M12}} & \underline{\text{M21}} \rightarrow \text{Chars}^{+} \\
\underline{\text{M2}} \rightarrow \underline{\text{M21}} & \underline{\text{M22}} \rightarrow \text{Chars}^{+} \\
\underline{\text{M2}} \rightarrow \underline{\text{M21}}\ \underline{\text{M22}} &
\end{array}
\tag{4}
$$

Next we introduce the notion of a *morphological template*, which is an ordered sequence of non-terminals whose concatenated yields constitute the word and which are used to parse out a specific segmentation of the word. For example, using Grammar 4 the parse tree of the word *saltiness* is shown in Figure 1. There are four possible templates with four

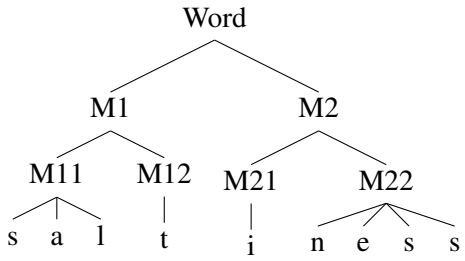

Figure 1: The parse tree generated by the metagrammar of depth 2 for the word *saltiness.*

different segmentations: **M1 M2** (*salt_iness*), **M11 M12 M2** (*sal_t_iness*), **M1 M21 M22** (*salt_i_ness*), and **M11 M12 M21 M22** (*sal_t_i_ness*).

The morphological template consisting only of non-terminals from the lowest cached level of the parse tree is expected to have high recall, whereas the template containing the non-terminals just below the Word is expected to have high precision. Our goal is to find the optimal template by using a small labelled data set. The grammar selection process iterates over the set of all templates. For each template, the segmentations of the words in the labelled data set are parsed out and the value of the desired evaluation metric is computed. The template that obtained the highest score is then chosen.

For each language we use a single template to segment all the words in that language. However, even using (say) a four-morph template such as **M11 M12 M21 M22**, some words may contain fewer morphs because the metagrammar permits either unary or binary branching rules, so some parses may not contain **M12** or **M2** (and thus **M21 M22**) spans. Thus, we can represent segmentations of different lengths (from 1 to 2^n, where n is the depth of the metagrammar) with a single template.[3]

For our experiments we use a metagrammar of depth four. This grammar allows words to consist of up to 16 segments, which we felt would be enough for any word in the training data. Also, iterating over all the templates of a grammar with bigger depth would not be feasible as the number of different templates increases very rapidly.[4]

[2]We started with Mark Johnson's PYAG implementation, http://web.science.mq.edu.au/~mjohnson/code/py-cfg.tgz, which we also used for our unsupervised and grammar selection experiments.

[3]We also experimented with selecting different templates for words of different length but observed no improvements over the single template approach.

[4]The number of templates of each depth can be expressed recursively as $N_i = (N_{i-1} + 1)^2$, where N_{i-1} is the number of

3.4 Inductive Learning

Previous work on AGs has used relatively small data sets and run the sampler on the entire input corpus (some or all of which is also used for evaluation)—a *transductive* learning scenario. However, our larger data sets contain millions of word types, where sampling over the whole set is not feasible. For example, 1000 training iterations on 50k word types took about a week on one 2.67 GHz CPU. To solve this problem, we need an *inductive* learner that can be trained on a small set of data and then used to segment a different larger set.

To create such a learner, we run the sampler on up to 50k word types, and then extract the *posterior grammar* as a PCFG.[5] This grammar contains all the initial CFG rules, plus rules to generate each of the cached subtrees inferred by the sampler. The sampler counts of all rules are normalized to obtain a PCFG, and we can then use a standard CKY parser to decode the remaining data using this PCFG.

4 Experiments

4.1 Data

We test on languages with a range of morphological complexity: English, Finnish, Turkish, German, and Estonian. For each language we use two small sets of gold-annotated data—a *labelled set* for semi-supervised training or model selection and a *dev set* for development results—and one larger gold-annotated dataset for final tests. We also have a large *unlabelled* training set for each language. Table 1 gives statistics.

The data sets for English, Finnish, Turkish and German are from the Morpho Challenge 2010 competition[6] (MC2010). We use the MC2010 training set of 1000 annotated word types as our labelled data, and for our dev sets we collate together the development data from all years of the MC competition. Final evaluation is done on the official MC2010 test sets, which are not public, so we rely on the MC organizers to perform the evaluation. The words in each test set are an unknown subset of the words in the unlabelled corpus, so to evaluate we segmented the entire unlabelled corpus and sent the results to the MC team, who then computed scores on the test words.

	Unlab.	Lab.	Dev	Test
English	0.9M	1000	1212	16K
Finnish	2.9M	1000	1494	225K
Turkish	0.6M	1000	1531	64K
German	2.3M	1000	785	62K
Estonian	2.1M	1000	1500	74K

Table 1: Number of word types in our data sets.

The Estonian wordlist is gathered from the newspaper texts of a mixed corpus of Estonian.[7] Gold standard segmentations of some of these words are available from the Estonian morphologically disambiguated corpus;[8] we used these for the test set, with small subsets selected randomly for the labelled and dev sets.

For semi-supervised tests of the AG Compounding grammar we annotated the morphemes in the English, Finnish and Estonian labelled sets as prefixes, stems or suffixes. We could not do so for Turkish because none of the authors knows Turkish.

4.2 Evaluation

We evaluate our results with two measures: segment border F1-score (SBF1) and EMMA (Spiegler and Monson, 2010). SBF1 is one of the simplest and most popular evaluation metrics for morphological segmentations. It computes F1-score from the precision and recall of ambiguous segment boundaries—i.e., word edges are not counted. It is easy and quick to compute but has the drawback that it gives no credit for one-morpheme words that have been segmented correctly (i.e., are assigned no segment borders). Also it can only be used on systems and gold standards where the output is just a segmentation of the surface string (e.g., *availabil+ity*) rather than a morpheme analysis (e.g., *available+ity*). For this reason we cannot report SBF1 on our German data, which annotations contain only analyses.

EMMA is a newer measure that addresses both

templates in the grammar of depth one less and $N_0 = 0$.

[5] This can be seen as a form of Structure Compilation (Liang et al., 2008), where the solution found by a more costly model is used to define a less costly model. However in Liang et al.'s case both models were already inductive.

[6] http://research.ics.aalto.fi/events/morphochallenge2010/datasets.shtml

[7] http://www.cl.ut.ee/korpused/segakorpus/epl

[8] http://www.cl.ut.ee/korpused/morfkorpus/

of these issues—correctly segmented one-morpheme words are reflected in the score, and it can evaluate both concatenative and non-concatenative morphology. EMMA works by finding the best one-to-one mapping between the hypothesized and true segments. The induced segments are then replaced with their mappings and based on that, F1-score on matching segments is calculated. Using EMMA we can evaluate the induced segmentations of German words against gold standard analyses. EMMA has a freely available implementation,[9] but is slow to compute because it uses Integer Linear Programming.

For our dev results, we computed both scores using the entire dev set, but for the large test sets, the evaluation is done on batches of 1000 word types selected randomly from the test set. This procedure is repeated 10 times and the average is reported, just as in the MC2010 competition (Kohonen et al., 2010a).

4.3 Baseline Models

We compare our AG models to several other morphology learning systems. We were able to obtain implementations of two of the best unsupervised systems from MC2010, Morfessor (Creutz and Lagus, 2007) and Morsel (Lignos, 2010), and we use these for comparisons on both the dev and test sets. We also report test results from MC2010 for the only semi-supervised system in the competition, semi-supervised Morfessor (Kohonen et al., 2010a; Kohonen et al., 2010b). No dev results are reported on this system since we were unable to obtain an implementation. This section briefly reviews the systems.

4.3.1 Morfessor Categories-MAP

Morfessor Categories-MAP (Morfessor) is a state-of-the-art unsupervised morphology learning system. Its implementation is freely available[10] so it is widely used both as a preprocessing step in tasks requiring morphological segmentations, and as a baseline for evaluating morphology learning systems.

Morfessor uses the Minimum Description Length (MDL) principle to choose the optimal segment lexicon and the corpus segmentation. Each morph in the segment lexicon is labelled as a stem, prefix, suffix or non-morph. The morphotactic rules are encoded as an HMM, which specifies the allowed morph sequences with respect to the labels (e.g., a suffix cannot directly follow a prefix).

The morphs in the segment lexicon can have a hierachical structure, containing submorphs which themselves can consist of submorphs etc. We hypothesize that this hierarchical structure is one of the key reasons why Morfessor has been so successful, as the experiments also in this paper with different grammars show that the ability to learn latent structures is crucial for learning good segmentations.

One essential difference between Morfessor and the proposed AG Select is that while we use the labelled data to choose which levels of the hierarchy are to be used as morphs, Morfessor makes this decision based on the labels of the segments, choosing the most fine-grained morph sequence that does not contain the non-morph label.

Morfessor includes a free parameter, perplexity threshold, which we found can affect the SBF1 score considerably (7 points or more). The best value for this parameter depends on the size of the training set, characteristics of the language being learned, and also the evaluation metric being used, as in some cases the best SBF1 and EMMA scores are obtained with completely different values.

Thus, we tuned the value of the perplexity threshold on the labelled set for each language and evaluation metric for different unlabelled training set sizes.

4.3.2 Semi-Supervised Morfessor

Recently, the Morfessor system has been adapted to allow semi-supervised training. Four versions of the system were evaluated in MC2010, using different degrees of supervision. Results reported here are from the Morfessor S+W system, which performed best of those that use the same kind of labelled data as we do.[11] This system uses the Morfessor Baseline model (not Cat-MAP), which incorporates a lexicon prior and data likelihood term. The semi-supervised version maintains separate likelihoods for the labelled and unlabelled data, and uses the development set to tune two parameters that weight these terms with respect to each other and the prior.

[9] http://www.cs.bris.ac.uk/Research/MachineLearning/Morphology/resources.jsp#eval

[10] http://www.cis.hut.fi/projects/morpho/morfessorcatmapdownloadform.shtml

[11] Morfessor S+W+L performs better, but uses training data with morpheme analyses rather than surface segmentations.

4.3.3 Morsel

Morsel (Lignos, 2010) is an unsupervised morphology learning system introduced in MC2010; we obtained the implementation from the author. Morsel learns morphological analyses rather than segmentations, so it can be evaluated only using EMMA. There are two options provided for running Morsel: aggressive and conservative. We used the development set to choose the best in each experimental case.

The MC data sets contain gold standard morphological analyses (as well as segmentations) so we could compute Morsel's EMMA scores using the analyses. However, we found that Morsel obtains higher EMMA scores when evaluated against gold standard segmentations and thus we used this option in all the experiments. (EMMA scores for other systems were also computed using the segmentations.)

4.4 Method

The experiments were conducted in two parts. First, we evaluated different aspects of the AG models and compared to all baseline models using the dev set data. Then we evaluated the most competitive models on the final test data.

For the development experiments, we compiled unlabelled training sets with sizes ranging from 10k to 50k word types (using the most frequent word types in each case). For the AG results, we report the average of five different runs made on the same training set. We let the sampler run for 1000 iterations. No annealing was used as it did not seem to help. The table label resampling option was turned on and the hyperparameter values were inferred.

We trained all AG and baseline models on each of these training sets. For AG Select, the words from the labelled data set were added to the training set to allow for template selection.[12] To compute results in transductive mode, the words from the dev set were also added to the training data. In inductive mode, the dev set was instead parsed with the CKY parser.

Preliminary experiments showed that the performance of unsupervised AG and AG Select improved with larger training sets, though the effect is small (see Figure 2 for results of AG Select in transductive mode; the trend in inductive mode is similar). Based on these and similar results with other baseline systems, all results reported later for unsupervised models (AG and baseline) and AG Select were obtained using training sets of 50k words.

In contrast to the above models, the semi-supervised AG does not always improve with more unlabelled data (see Figure 2) and in the limit, it will match the performance of the same grammar in the unsupervised setting. Other semi-supervised approaches often solve this problem by weighting the labelled data more heavily than the unlabelled data when estimating model parameters—effectively, assuming that each labelled item has actually been observed more than once. However, duplicating the labelled data does not make sense in the AG framework, because duplicate items will in most cases just be cached at the root (Word) node, providing no additional counts of Morphs (which are where the useful information is). It might be possible to come up with a different way to weight the labelled data more heavily when larger unlabelled sets are used, however for this paper we instead kept the labelled data the same and tuned the amount of unlabelled data. We used the dev set to choose the amount of unlabelled data (in the range from 10k to 50k types); results for semi-supervised AG are reported using the optimal amount of unlabelled data for each experiment.

For test set experiments with semi-supervised AG, we evaluated each language using whichever grammar performed best on that language's dev set. For test set experiments with AG Select, we chose the templates with a two-pass procedure. First, we trained 5 samplers on the 50k training set with labelled set added, and used the labelled data to choose the best template for each inferred grammar. Then, we decoded the dev set using each of the 5 grammar/template pairs and based on these results, chose the best of these pairs to decode the test set.

4.5 Results

We present the dev set results in Table 2(a) for transductive and in Table 2(b) for inductive learning. In each table, unsupervised models are shown in the upper section and the semi-supervised models and AG Select below. Morsel appears only in Table 2(a) since it only works transductively. Semi-supervised grammars cannot be trained on German, since we

[12] We also experimented with smaller sets of labelled data. In most cases, the template selected based on only 300 word types was the same than the one selected with 1000 word types.

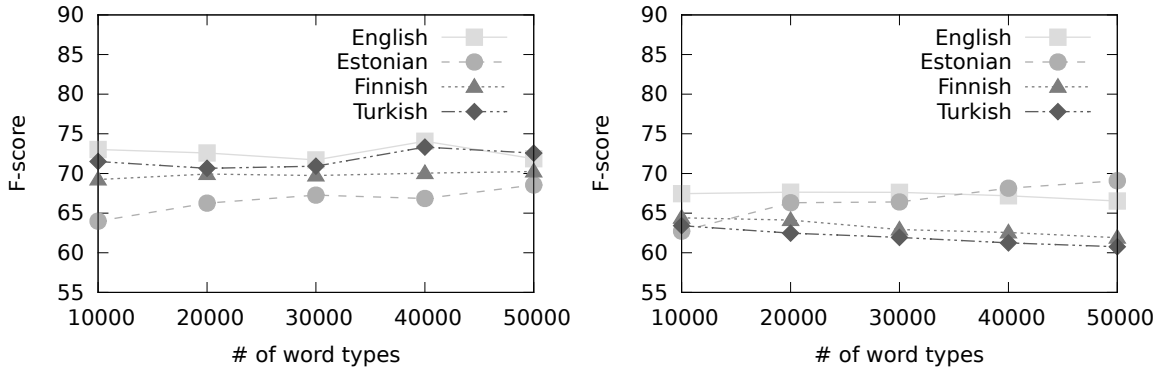

Figure 2: Effect of training data size on dev set SBF1 for AG Select (left) and semi-supervised SubMorphs grammar (right) in transductive mode.

only have gold standard analyses, not segmentations.

The SubMorphs grammar performs the best of the unsupervised AG models, with the Compounding grammar being only slightly worse. We also tried the Compounding grammar without the sub-morph structures but the results were even worse than those of MorphSeq. This shows that the latent structures are important for learning good segmentations.

In all cases, the semi-supervised AGs perform better (ofen much better) than the corresponding unsupervised grammars. Even though their average scores are not as high as AG Select's, they give the best dev set results in many cases. This shows that although for semi-supervised AG the grammar must be chosen manually, with a suitable choice of the grammar and only a small set of labelled data it can improve considerably over unsupervised AG.

In transductive mode, the AG Select performs the best in several cases. In both transductive and inductive mode, the results of AG Select are close to the best results obtained and are consistently good across all languages—it achieves the best average scores of all models, suggesting that the model selection method is robust to different types of morphology and annotation schemes.

Table 3 presents the test set results. We include scores for unsupervised Morfessor in both transductive and inductive mode, where transductive mode trains on the entire unlabelled corpus and inductive mode trains on the 50k subset. The semi-supervised Morfessor scores are taken from the MC results page[13] after verifying that the evaluation methodology and labelled data used is the same as ours.[14]

There is a good deal of variation between development and test results, indicating that the dev sets may not be a representative sample. The most notable differences are in Turkish, where all models perform far worse on the test than dev set. However, AG Select performs slightly better on the test set for the other languages. Thus its average SBF1 score actually improves on the test set and is not much worse than semi-supervised Morfessor. While its average performance drops somewhat on test set EMMA, it is still as good as any other model on that measure. Again, these results support the idea that AG Select is robust to variations in language and data set.

We also note the surprisingly good performance of Morfessor in transductive mode on Estonian; this could possibly be due to the larger amount of training data used for the test set results, but it is not clear why this would improve performance so much on Estonian and not on the other languages.

5 Discussion

To give a sense of what the AG Select model is learning, we provide some examples of both correctly and incorrectly induced segmentations in Table 4. These examples suggest that for example in English, M1 is used to model the stem, M21 is for the suffix or the second stem in the compound word, and the rest of the elements in the template are for the remaining suffixes (if any).

Table 5 presents examples of some of the most frequently used metagrammar rules and cached rules

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

[14] Sami Virpioja, personal communication.

(a) Transductive mode	Border F1-score					EMMA					
	Eng	**Est**	**Fin**	**Tur**	**Avg**	**Eng**	**Est**	**Fin**	**Tur**	**Ger**	**Avg**
AG MorphSeq	61.5	54.0	56.9	59.5	58.0	74.7	74.1	63.7	53.5	59.4	65.1
AG SubMorphs	66.2	66.9	60.5	59.5	63.3	79.1	83.4	66.8	53.4	57.4	68.0
AG Compounding	63.0	64.8	60.9	60.9	62.4	75.4	81.6	65.5	53.7	62.2	67.7
Morfessor	69.5	55.7	65.0	69.3	64.9	**81.3**	75.3	67.8	62.2	**62.7**	69.9
Morsel	-	-	-	-	-	76.8	74.4	66.1	50.1	55.9	64.7
AG ssv MorphSeq	64.4	57.3	63.0	68.9	63.4	74.4	75.9	65.6	59.6	-	-
AG ssv SubMorphs	67.6	**69.1**	64.4	63.4	66.1	79.5	**84.4**	69.2	56.1	-	-
AG ssv Compounding	70.0	67.5	**71.8**	-	-	79.5	82.8	**74.0**	-	-	-
AG Select	**71.9**	68.5	70.2	**72.6**	**70.8**	77.5	81.8	73.2	**63.0**	62.4	**71.6**

(b) Inductive mode	Border F1-score					EMMA					
	Eng	**Est**	**Fin**	**Tur**	**Avg**	**Eng**	**Est**	**Fin**	**Tur**	**Ger**	**Avg**
AG MorphSeq	57.6	54.0	55.4	59.8	56.7	72.0	73.8	62.6	53.7	58.9	64.2
AG SubMorphs	66.1	67.5	61.6	59.8	63.7	78.6	**83.7**	67.4	53.4	56.0	67.8
AG Compounding	62.0	64.8	57.4	61.1	61.3	73.5	81.1	61.9	53.2	61.0	66.2
Morfessor	68.9	51.1	63.5	68.2	62.9	**80.9**	72.0	68.1	60.6	60.8	68.5
AG ssv MorphSeq	64.6	56.9	63.1	**70.3**	63.8	72.7	73.3	65.9	**61.2**	-	-
AG ssv SubMorphs	70.1	**69.7**	66.3	67.9	68.4	80.4	**83.7**	70.5	59.0	-	-
AG ssv Compounding	**70.5**	67.2	**70.0**	-	-	77.3	81.9	70.5	-	-	-
AG Select	69.8	68.8	67.5	70.1	**69.1**	77.3	81.9	**71.1**	59.7	**62.6**	**70.5**

Table 2: Dev set results for all models in (a) transductive and (b) inductive mode. Unsupervised AG models and baselines are shown in the top part of each table; semi-supervised AG models and grammar selection method are below.

	Border F1-score						EMMA						
	Eng	**Est**	**Fin**	**Tur**	**Avg**	**-Est**	**Eng**	**Est**	**Fin**	**Tur**	**Ger**	**Avg**	**-Est/Ger**
Morf. trans	67.3	**73.9**	61.2	57.1	64.9	61.9	78.4	78.8	61.8	49.8	**65.2**	66.8	63.3
Morf. ind	65.7	57.7	60.8	60.1	61.1	62.2	76.5	70.5	59.6	47.0	64.1	63.5	61.0
Morsel	-	-	-	-	-	-	**81.9**	77.2	63.3	47.8	59.0	65.8	**64.3**
Morf. ssv	**77.8**	-	**71.7**	**68.9**	-	**72.8**	80.6	-	62.1	**49.9**	-	-	64.2
AG ssv best	$70.3^{\star}$	$68.6^{\dagger}$	$64.9^{\star}$	58.2^{*}	65.5	64.5	$75.9^{\star}$	$80.3^{\dagger}$	$61.3^{\star}$	46.1^{*}	-	-	61.1
AG Select	74.4	71.7	70.0	61.4	**69.4**	68.6	81.3	**81.0**	**64.0**	47.5	63.8	**67.5**	**64.3**

Table 3: Test set results for unsupervised baselines Morfessor CatMAP (in transductive and inductive mode) and Morsel; semi-supervised Morfessor; and AG semi-supervised ($\star$ marks the Compounding grammar, † denotes SubMorphs grammar, and $*$ is the MorphSeq grammar) and grammar selection methods. Results are shown for each language, averaged over all languages (when possible: Avg), and averaged over just the languages where scores are available for all systems (-Est, -Est/Ger).

for English, together with their relative frequencies. It shows that at the Word level the binary rule is selected over three times more frequently than the unary rule. Also, most of the more frequently used grammar rules expand the first branch (rooted in M1) into more finegrained structures. The second branch (M2) is mostly modelled with the unary rule.

Among the frequently cached rules we see the common English prefixes and suffixes. One of the most frequent cached rule stores the single letter *e* at the end of a word, which often causes oversegmentation of words ending in *e* (as seen in the incorrect examples in Table 4). This problem is common in unsupervised morphological segmentation of English (Goldwater et al., 2006; Goldsmith, 2001).

We also took a look at the most frequent cached rules learned by the semi-supervised AG with the SubMorphs grammar, and observed that Morphs

Correct Segmentations		Incorrect Segmentations	
Word	**Segmentation**	**Induced**	**Correct**
treatable	[tr.ea.t]$_{M1}$ [a.b.le]$_{M21}$	disagree_s	dis_agree_s
disciplined	[dis.cip.l.i.n]$_{M1}$ [e.d]$_{M21}$	reduc_e	reduce
monogamous	[mon.o.g.a.m]$_{M1}$ [o.u.s]$_{M21}$	revalu_e	re_value
streakers	[st.r.e.a.k]$_{M1}$ [e.r]$_{M21}$ [s]$_{M2211}$	derid_e	deride
tollgate	[t.o.l.l.]$_{M1}$ [g.a.t.e]$_{M21}$ [s]$_{M2211}$	accompani_ed	ac_compani_ed
foxhunting	[f.o.x]$_{M1}$ [h.u.n.t]$_{M21}$ [ing]$_{M2211}$	war_y	wary
muscovites	[m.u.sc.o.v]$_{M1}$ [i.t.e]$_{M21}$ [s]$_{M2211}$	indescrib_able	in_describ_able
standardizes	[st.a.n.d.a.rd]$_{M1}$ [i.z.e]$_{M21}$ [s]$_{M2211}$	orat_es	orate_s
slavers'	[sl.a.v]$_{M1}$ [e.r]$_{M21}$ [s]$_{M2211}$ [']$_{M2212}$	alger_ian_s	algeri_an_s
earthiness'	[e.ar.th]$_{M1}$ [i]$_{M2111}$ [ness]$_{M2211}$ [']$_{M2212}$	disput_e_s	dispute_s
instinkt	[in.st.in.kt]$_{M1}$	meister_likkust	meisterlikkus_t
rebis	[re.b.i]$_{M1}$ [s]$_{M2}$	min_a	mina
toitsid	[to.it]$_{M1}$ [s.id]$_{M2}$	teiste	teis_te
armuavaldus	[a.rm.u]$_{M11}$ [ava.ld.u.s]$_{M12}$	kuritegu_de_sse	kuri_tegu_desse
määgivale	[mää.g.i]$_{M11}$ [v.a]$_{M12}$ [l.e]$_{M2}$	liharoa_ga	liha_roa_ga
keskuskoulussa	[kesk.us]$_{M11}$ [koul.u]$_{M12}$ [s.sa]$_{M2}$	polte_tti_in	polte_tt_i_in
peruslähteille	[per.u.s]$_{M11}$ [l.ä.ht.e]$_{M12}$ [i]$_{M211}$ [ll.e]$_{M212}$	kulttuuri_se_lt_a_kin	kulttuurise_lta_kin
perunakaupoista	[per.u.n.a]$_{M11}$ [k.au.p.o]$_{M12}$ [i]$_{M211}$ [st.a]$_{M212}$	tuote_palki_ntoja	tuote_palkinto_j_a
yöpaikkaani	[yö]$_{M11}$ [p.ai.kk.a]$_{M12}$ [a]$_{M21}$ [n.i]$_{M22}$	veli_puo_lt_a	veli_puol_ta
nimettäköön	[ni.m.e]$_{M11}$ [tt.ä]$_{M12}$ [k.ö]$_{M21}$ [ö.n]$_{M22}$	ota_ttava	otatta_va

Table 4: Examples of segmented words in English (top), Estonian (middle) and Finnish (bottom). Correctly segmented words are in the left part of the table. The identified segments are in brackets indexed by the respective template nonterminal; dots separate the metagrammar generated parse tree leaves. Examples of incorrectly segmented words together with the correct segmentation are on the right.

Freq (%)	Rule	Freq (%)	Cached Rule
9.9	Word → M1 M2	1.2	(M2 (M21 (M211 (M2111 s)))))
5.7	M1 → M11 M12	0.9	(M2 (M21 (M211 (M2111 e)) (M212 (M2121 d))))
3.1	Word → M1	0.7	(M2 (M21 (M211 (M2111 i)) (M212 (M2121 n g))))
2.5	M11 → M111	0.6	(M2 (M21 (M211 (M2111 e)))
1.8	M2 → M21	0.4	(M2 (M21 (M211 (M2111 '))) (M22 (M221 (M2211 s))))
1.4	M12→ M121 M122	0.3	(M1112 a)
1.4	M111 → M1111 M1112	0.3	(M2 (M21 (M211 (M2111 y))))
0.9	M12 → M121	0.3	(M2 (M21 (M211 (M2111 e))) (M212 (M2121 r)))
0.9	M11 → M111 M112	0.2	(M2 (M21 (M211 (M2111 a))))

Table 5: Examples from English most frequently used metagrammar rules and cached rules together with their relative occurrence frequencies (in percentages).

tended to contain only a single SubMorph. This helps to explain why the SubMorphs grammar in semi-supervised AG improved less over the unsupervised AG as compared to the MorphSeq grammar—the rules with only a single SubMorph under the Morph are essentially the same as they would be in the MorphSeq grammar.

Finally, we examined the consistency of the templates chosen for each of the 5 samplers during model selection for the test set (Section 4.4). We found that there was some variability in the templates, but in most experiments the same template was chosen for the majority of the samplers (see Table 6). While this majority template is not always the optimal one on the dev set, we observed that it does produce consistently good results. It is possible that using the majority template, rather than the optimal template for the dev set, would actually produce better results

	Majority template
English	M1 M21 M2211 M2212 M222
Finnish	M11 M12 M211 M212 M22
Turkish	M11 M12 M211 M212 M22
German	M11 M121 M122 M21 M221 M222
Estonian	M11 M12 M2

Table 6: Majority templates for each language. Note that the Estonian gold standard contains less fine-grained segmentations than some of the other languages.

on the test set, especially if (as appears to be the case here, and may often be the case in real applications) the dev and test sets are from somewhat different distributions.

It must be noted that both AG Select and semi-supervised AG are computationally more demanding than the comparison systems. Since we do inference over tree structures, the complexity is cubic in the input word length, while most segmentation systems are quadratic or linear. Even compared to the unsupervised AG, AG Select is more expensive, because of the larger grammar and number of cached symbols. Nevertheless, our systems can feasibly be run on the large Morpho Challenge datasets.

Other recent unsupervised systems have reported state-of-the art results by incorporating additional information from surrounding words (Lee et al., 2011), multilingual alignments (Snyder and Barzilay, 2008), or overlapping context features in a log-linear model (Poon et al., 2009), but they have only been run on Semitic languages and English (and in the latter case, a very small corpus). Since they explicitly enumerate and sample from all possible segmentations of each word (often with some heuristic constraints), they could have trouble with the much longer words of the agglutinative languages tested here. In any case the results are not directly comparable to ours.

6 Conclusion

In this paper we have introduced three new methods for Adaptor Grammars and demonstrated their usefulness for minimally supervised morphological segmentation. First, we showed that AG models can be scaled to large data sets by using the posterior grammar for defining an inductive model, that on average results in the same accuracy as compared to full transductive training.

Second, we implemented semi-supervised AG inference, which uses labelled data to constrain the sampler, and showed that in all cases it performs much better than the unsupervised AG on the same grammar. Semi-supervised AG could benefit from labelled data reweighting techniques frequently used in semi-supervised learning, and studying the proper ways of doing so within the AG framework would be a potential topic for future research.

Our final contribution is the AG Select method, where the initial model is trained using a very general grammar that oversegments the data, and the labelled data is used to select which granularity of segments to use. Unlike other morphological segmentation models, this method can adapt its grammar to languages with different structures, rather than having to use the same grammar for every language. Indeed, we found that AG Select performs well across a range of languages and also seems to be less sensitive to differences between data sets (here, dev vs. test). In addition, it can be trained on either morphological analyses or segmentations. Although we tuned all results to optimize the SBF1 metric, in principle the same method could be used to optimize other measures, including extrinsic measures on downstream applications such as machine translation or information retrieval. In future we hope to show that this method can be used to improve performance on such applications, and also to explore its use for related segmentation tasks such as stemming or syllabification. Also, the method itself could potentially be improved by designing a classifier to determinine the best template for each word based on a set of features, rather than using a single template for all words in the language.

Acknowledgments

This work was supported by the Tiger University program of Estonian Information Technology Foundation for the first author. We thank Constantine Lignos for releasing his Morsel code to us, Sami Virpioja for evaluating test set results, and Federico Sangati for providing useful scripts.

References

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology

learning. *ACM Transactions of Speech and Language Processing*, 4(1):1–34, February.

Micha Elsner, Eugene Charniak, and Mark Johnson. 2009. Structured generative models for unsupervised named-entity clustering. In *Proceedings of NAACL*, pages 164–172. Association for Computational Linguistics.

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

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2006. Interpolating between types and tokens by estimating power-law generators. In *Advances in Neural Information Processing Systems 18*, pages 459–466, Cambridge, MA. MIT Press.

Eric A. Hardisty, Jordan Boyd-Graber, and Philip Resnik. 2010. Modeling perspective using adaptor grammars. In *Proceedings of EMNLP*, pages 284–292. Association for Computational Linguistics.

Zellig Harris. 1951. *Structural Linguistics*. University of Chicago Press.

Yun Huang, Min Zhang, and Chew Lim Tan. 2011. Nonparametric bayesian machine transliteration with synchronous adaptor grammars. In *Proceedings of ACL: short papers - Volume 2*, pages 534–539. Association for Computational Linguistics.

Mark Johnson and Katherine Demuth. 2010. Unsupervised phonemic chinese word segmentation using adaptor grammars. In *Proceedings of COLING*, pages 528–536. Association for Computational Linguistics.

Mark Johnson and Sharon Goldwater. 2009. Improving nonparameteric bayesian inference: experiments on unsupervised word segmentation with adaptor grammars. In *Proceedings of NAACL*, pages 317–325. Association for Computational Linguistics.

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

Mark Johnson. 2008a. Unsupervised word segmentation for sesotho using adaptor grammars. In *Proceedings of ACL Special Interest Group on Computational Morphology and Phonology*, pages 20–27. Association for Computational Linguistics.

Mark Johnson. 2008b. Using adaptor grammars to identify synergies in the unsupervised acquisition of linguistic structure. In *Proceedings of ACL*, pages 398–406. Association for Computational Linguistics.

Oskar Kohonen, Sami Virpioja, and Krista Lagus. 2010a. Semi-supervised learning of concatenative morphology. In *Proceedings of ACL Special Interest Group on Computational Morphology and Phonology*, pages 78–86. Association for Computational Linguistics.

Oskar Kohonen, Sami Virpioja, Laura Leppänen, and Krista Lagus. 2010b. Semi-supervised extensions to morfessor baseline. In *Proceedings of the Morpho Challenge 2010 Workshop*, pages 30–34. Aalto University School of Science and Technology.

Yoong Keok Lee, Aria Haghighi, and Regina Barzilay. 2011. Modeling syntactic context improves morphological segmentation. In *Proceedings of CoNLL*, pages 1–9. Association for Computational Linguistics.

Percy Liang, Hal Daumé, III, and Dan Klein. 2008. Structure compilation: trading structure for features. In *Proceedings of ICML*, pages 592–599. Association for Computing Machinery.

Constantine Lignos. 2010. Learning from Unseen Data. In Mikko Kurimo, Sami Virpioja, and Ville T. Turunen, editors, *Proceedings of the Morpho Challenge 2010 Workshop*, pages 35–38. Aalto University School of Science and Technology.

Jim Pitman and Marc Yor. 1997. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *Annals of Probability*, 25(2):855–900.

Hoifung Poon, Colin Cherry, and Kristina Toutanova. 2009. Unsupervised morphological segmentation with log-linear models. In *Proceedings of NAACL*, pages 209–217. Association for Computational Linguistics.

Benjamin Snyder and Regina Barzilay. 2008. Unsupervised multilingual learning for morphological segmentation. In *Proceedings of ACL*, pages 737–745. Association for Computational Linguistics.

Sebastian Spiegler and Christian Monson. 2010. Emma: A novel evaluation metric for morphological analysis. In *Proceedings of COLING*, pages 1029–1037. Association for Computational Linguistics.

Sze-Meng Jojo Wong, Mark Dras, and Mark Johnson. 2012. Exploring adaptor grammars for native language identification. In *Proceedings of EMNLP*, pages 699–709. Association for Computational Linguistics.