Unsupervised Learning of Morphological Forests

Jiaming Luo
CSAIL, MIT
j_luo@mit.edu

Karthik Narasimhan
CSAIL, MIT
karthikn@mit.edu

Regina Barzilay
CSAIL, MIT
regina@csail.mit.edu

Abstract

This paper focuses on unsupervised modeling of morphological families, collectively comprising a forest over the language vocabulary. This formulation enables us to capture edge-wise properties reflecting single-step morphological derivations, along with global distributional properties of the entire forest. These global properties constrain the size of the affix set and encourage formation of tight morphological families. The resulting objective is solved using Integer Linear Programming (ILP) paired with contrastive estimation. We train the model by alternating between optimizing the local log-linear model and the global ILP objective. We evaluate our system on three tasks: root detection, clustering of morphological families, and segmentation. Our experiments demonstrate that our model yields consistent gains in all three tasks compared with the best published results.[1]

1 Introduction

The morphological study of a language inherently draws upon the existence of families of related words. All words within a family can be derived from a common root via a series of transformations, whether inflectional or derivational. Figure 1 depicts one such family, originating from the word *faith*. This representation can benefit a range of applications, including segmentation, root detection and clustering of morphological families.

[1]Code is available at https://github.com/j-luo93/MorphForest.

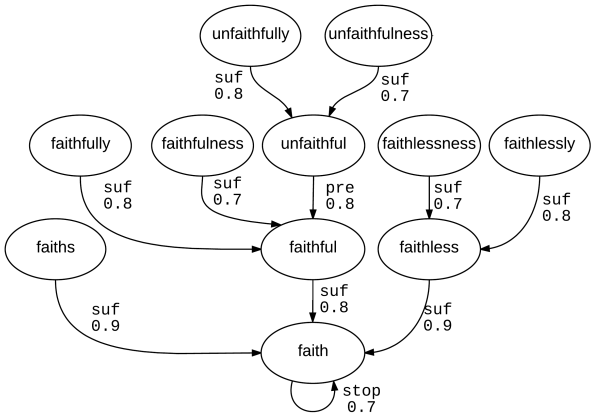

Figure 1: An illustration of a single tree in a morphological forest. *pre* and *suf* represent prefixation and suffixation. Each edge has an associated probability for the morphological change.

Using graph terminology, a full morphological assignment of the words in a language can be represented as a forest.[2] Valid forests of morphological families exhibit a number of well-known regularities. At the global level, the number of roots is limited, and only constitute a small fraction of the vocabulary. A similar constraint applies to the number of possible affixes, shared across families. At the local edge level, we prefer derivations that follow regular orthographic patterns and preserve semantic relatedness. We hypothesize that enforcing these constraints as part of the forest induction pro-

[2]The correct mathematical term for the structure in Figure 1 is a *directed 1-forest* or *functional graph*. For simplicity, we shall use the terms forest and tree to refer to a directed 1-forest or a directed 1-tree because of the cycle at the root.

Transactions of the Association for Computational Linguistics, vol. 5, pp. 353–364, 2017. Action Editor: Eric Fosler-Lussier.
Submission batch: 10/2016; Revision batch: 12/2016; Published 10/2017.

cess will allow us to accurately learn morphological structures in an unsupervised fashion.

To test this hypothesis, we define an objective over the entire forest representation. The proposed objective is designed to maximize the likelihood of local derivations, while constraining the overall number of affixes and encouraging tighter morphological families. We optimize this objective using Integer Linear Programming (ILP), which is commonly employed to handle global constraints. While in prior work, ILP has often been employed in supervised settings, we explore its effectiveness in unsupervised learning. We induce a forest by alternating between learning local edge probabilities using a log-linear model, and enforcing global constraints with the ILP-based decoder. With each iteration, the model progresses towards more consistent forests.

We evaluate our model on three tasks: root detection, clustering of morphologically related families, and segmentation. The last task has been extensively studied in recent literature, providing us with the opportunity to compare the model with multiple unsupervised techniques. On benchmark datasets representing four languages, our model outperforms the baselines, yielding new state-of-the-art results. For instance, we improve segmentation performance on Turkish by 4.4% and on English by 3.7%, relative to the best published results (Narasimhan et al., 2015). Similarly, our model exhibits superior performance on the other two tasks. We also provide analysis of the model behavior which reveals that most of the gain comes from enforcing global constraints on the number of unique affixes.

2 Related Work

Unsupervised morphological segmentation Most top performing algorithms for unsupervised segmentation today center around modeling single-step derivations (Poon et al., 2009; Naradowsky and Toutanova, 2011; Narasimhan et al., 2015). A commonly used log-linear formulation enables these models to consider a rich set of features ranging from orthographic patterns to semantic relatedness. However, these models generally bypass global constraints (Narasimhan et al., 2015) or require performing inference over very large spaces (Poon et al., 2009). As we show in our analysis (Section 5), this omission negatively affects model performance.

In contrast, earlier work focuses on modeling global morphological assignment, using generative probabilistic models (Creutz and Lagus, 2007; Snyder and Barzilay, 2008; Goldwater et al., 2009; Sirts and Goldwater, 2013). These models are inherently limited in their ability to incorporate diverse features that are effectively utilized by local discriminative models.

Our proposed approach attempts to combine the advantages of both approaches, by defining an objective that incorporates both levels of linguistic properties over the entire forest representation, and adopting an alternating training regime for optimization.

Graph-based representations in computational morphology Variants of a graph-based representation have been used to model various morphological phenomena (Dreyer and Eisner, 2009; Peng et al., 2015; Soricut and Och, 2015; Faruqui et al., 2016). The graph induction methods vary widely depending on the task and the available supervision. The distinctive feature of our work is the use of global constraints to guide the learning of local, edge-level derivations.

ILP for capturing global properties Integer Linear Programming has been successfully employed to capture global constraints across multiple applications such as information extraction (Roth and Yih, 2001), sentence compression (Clarke and Lapata, 2008), and textual entailment (Berant et al., 2011). In all of these applications, the ILP formulation is used with a supervised classifier. Our work demonstrates that this framework continues to be effective in an unsupervised setting, providing strong guidance for a local, unsupervised classifier.

3 Model

Our model considers a full morphological assignment for all the words in a language, representing it as a forest. Let $F = (V, E)$ be a directed graph where each word corresponds to a node $v \in V$. A directed edge $e = (v_c, v_p) \in E$ encodes a *single* morphological derivation from a *parent* word v_p to a *child* word v_c. Edges also reflect the type of the

underlying derivation (e.g., prefixation), and an associated probability $\Pr(e)$. Note that the root of a tree is always marked with a self-directed (i.e. v_c = v_p) edge associated with the label *stop*. Figure 1 illustrates a single tree in the forest.

3.1 Inducing morphological forests

We postulate that a valid assignment yields forests with the following properties:

1. **Increased edge weights** Edge weights reflect probabilities of single-step derivations based on the local features including orthographic patterns and semantic relatedness. This local information helps identify that the edge (*painter*, *paint*) should be preferred over (*painter*, *pain*), because $-er$ is a valid suffix and *paint* is semantically closer to *painter*.

2. **Minimized number of affixes** Prior research has shown that local models tend to greatly overestimate the number of suffixes. For instance, the model of Narasimhan et al. (2015) produces 617 unique affixes when segmenting 10000 English words. Thus, we explicitly encourage the model towards assignments with the least number of affixes.

3. **Minimized number of roots relatively to vocabulary size** Similarly, the number of roots, and consequently the number of morphological families is markedly smaller than the size of the vocabulary.

The first property is local in nature, while the last two are global and embody the principle of Minimum Description Length (MDL). Based on these properties, we formulate an objective function $\mathcal{S}(F)$ over a forest F:

$$\mathcal{S}(F) = -\frac{\sum_{e \in E} \log \Pr(e)}{|E|} + \alpha |\textit{Affix}| + \beta \frac{|F|}{|V|}, \quad (1)$$

where $|\cdot|$ denotes set cardinality, $\textit{Affix} = \{a_k\}_{k=1}^{K}$ is the set of all affixes, and $|F|$ is the number of trees in F. $|E|$ and $|V|$ are the size of the edge set and vocabulary, respectively. The hyperparameters α and β capture the relative importance of the three terms.

By minimizing this objective, we encourage assignments with high edge probabilities (first term), while limiting the number of affixes and morphological families (second and third terms, respectively). This objective can also be viewed as a simple log-likelihood objective regularized by the last two terms in Equation (1).

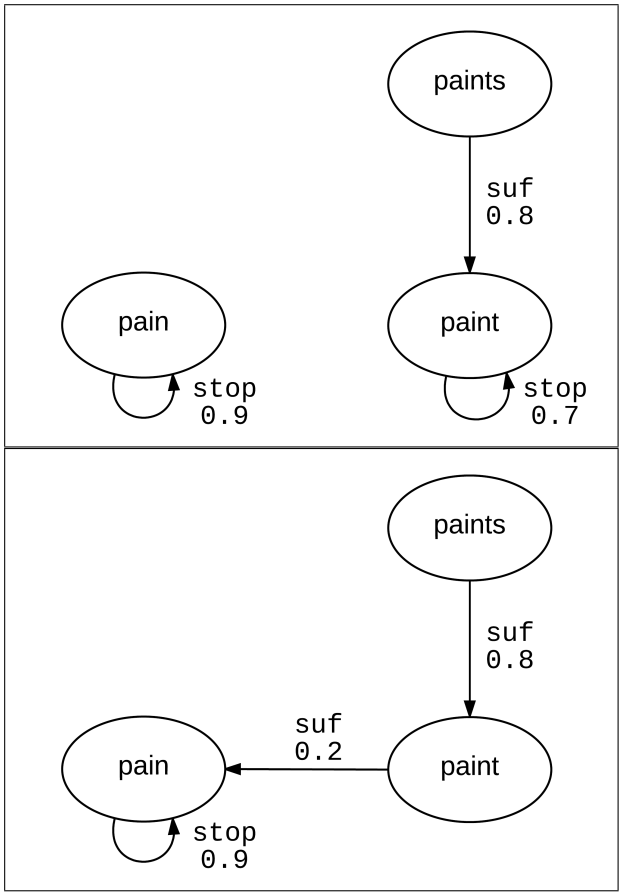

Figure 2: Illustration of two chosen forest representations. The top forest has only one affix *-s*, but two roots {*pain*, *paint*}. Shown in the bottom forest, choosing the edge (*paint*, *pain*) instead of (*paint*, *paint*) will introduce another affix *-t*, while reducing the set of roots to just {*pain*}.

To illustrate the interaction between local and global constraints in this objective, consider an example in Figure 2. If the model selects a different edge – e.g. (*paint*, *pain*) instead, all the terms in Equation (1) will be affected.

3.2 Computing local probabilities

We now describe how to parameterize $\Pr(e)$, which captures the likelihood of a single-step morphological derivation between two words. Following prior

work (Narasimhan et al., 2015), we model this probability using a log-linear model:

$$\Pr(w, z) \propto \exp(\theta \cdot \phi(w, z)), \tag{2}$$

where θ is the set of parameters to be learned, and $\phi(w, z)$ is the feature vector extracted from w and z. Each candidate z is a tuple (*string*, *label*), where *label* refers to the label of the potential edge.

As a result, the marginal probability is

$$\begin{aligned}\Pr(w) &= \sum_{z \in C(w)} \Pr(w, z) \\ &= \frac{\sum_{z \in C(w)} \exp(\theta \cdot \phi(w, z))}{\sum_{w' \in \Sigma^*} \sum_{z' \in C(w')} \exp(\theta \cdot \phi(w', z'))},\end{aligned}$$

where Σ^* is the set of all possible strings. Computing the sum in the denominator is infeasible. Instead, we make use of *contrastive estimation* (Smith and Eisner, 2005), substituting Σ^* with a (limited) set of neighbor strings $N(w)$ that are orthographically close to w. This technique distributes the probability mass among neighboring words and forces the model to identify meaningful discriminative features. We obtain $N(w)$ by transposing characters in w, following the method described in Narasimhan et al. (2015).

Now for the forest over the set of nodes V, the log-likelihood loss function is defined as:

$$\begin{aligned}\mathcal{L}(V; \theta) &= -\sum_{v \in V} \log \Pr(v) \\ &= -\sum_{v \in V} \Big[\log \sum_{z \in C(v)} \exp(\theta \cdot \phi(v, z)) \\ &\quad - \log \sum_{v' \in N(v)} \sum_{z' \in C(v')} \exp(\theta \cdot \phi(v', z')) \Big],\end{aligned} \tag{3}$$

This objective can be minimized by gradient descent.

Space of Possible Candidates We only consider assignments where the parent word is strictly shorter than the child word to prevent cycles of length two or more. In addition to suffixation and prefixation, we also consider three types of transformations introduced in Goldwater and Johnson (2004): repetition, deletion, and modification. We also handle compounding, where two stems are combined to form a new word (e.g., *football*). One of these stems carries the main semantic meaning of the compound and is considered to be the parent of the word. Note that stems are not considered affixes, so this does not affect the affix list.

We allow parents to be words outside V, since many legitimate word forms might never appear in the corpus. For instance, if we have $V = \{painter, paints\}$, the optimal solution would add an unseen word $paint$ to the forest, and choose edges (*painter*, *paint*) and (*paints*, *paint*).

Features We use the same set of features shown to be effective in prior work (Narasimhan et al., 2015), including word vector similarity, beginning and ending character bigrams, word frequencies and affixes. Affix features are automatically extracted from the corpus based on string difference and are thresholded based on frequency. We also include an additional sibling feature that counts how many words are siblings of word w in its tree. Siblings are words that are derived from the same parent, e.g., *faithful* and *faithless*, both from the word *faith*.

3.3 ILP formulation

Minimizing the objective in Equation (1) is challenging because the second and third terms capture discrete global properties of the forest, which prevents us from performing gradient descent directly. Instead, we formulate this optimization problem as Integer Linear Programming (ILP), where these two terms can be cast as constraints.[3]

For each child word $v_i \in V$, we have a bounded set of its candidate outgoing edges $C(v_i) = \{z_i^j\}$, where z_i^j is the j-th candidate for v_i. $C(v_i)$ is the same set as defined in Section 3.2. Each edge is associated with p_{ij}, which is computed as $\log \Pr(z_i^j | v_i)$. Let x_{ij} be a binary variable that has value 1 if and only if z_i^j is chosen to be in the forest. Without loss of generality, we assume the first candidate edge is always the self-edge (or *stop* case), i.e., $z_i^1 = (v_i, \textit{stop})$. We also use a set of binary variables $\{y_k\}$ to indicate whether affix a_k is used at

[3] If we had prior knowledge of words belonging to the same family, we can frame the problem as growing a Minimum Spanning Tree (MST), and use Chu-Liu-Edmonds algorithm (Chu and Liu, 1965; Edmonds, 1967) to solve it. However, this information is not available to us.

least once in F (i.e. required to explain a morphological change).

Now let us consider how to derive our ILP formulation using the notations above. Note that $|F|$ is equal to the number of self-edges $\sum_i x_{i1}$, and also a valid forest will satisfy $|V| = |E|$. Combining these pieces, we can rewrite the objective in Equation (1) and arrive at the following ILP formulation:

$$\begin{aligned}
\underset{x_{ij}, y_k}{\text{minimize}} \quad & -\frac{1}{|V|}\sum_{ij} x_{ij} p_{ij} + \alpha \sum_k y_k + \frac{\beta}{|V|}\sum_i x_{i1} \\
\text{subject to} \quad & x_{ij}, y_k \in \{0, 1\}, \\
& \sum_j x_{ij} = 1, \forall i, \qquad (4) \\
& x_{ij} \leq y_k, \text{if } a_k \text{ is involved in } z_i^j. \qquad (5)
\end{aligned}$$

Constraint 4 states that exactly one of the candidate edges should be chosen for each word. The last constraint implies that we can only consider this candidate (and construct the corresponding edge) when the involved affix[4] is used at least once in the forest representation.

3.4 Alternating training

The objective function contains two sets of parameters: a continuous weight vector θ that parameterizes edge probabilities, and binary variables $\{x_{ij}\}$ and $\{y_k\}$ in ILP. Due to the discordance between continuous and discrete variables, we need to optimize the objective in an alternating manner. Algorithm 1 details the training procedure. After automatically extracting affixes from the corpus, we alternate between learning the local edge probabilities (line 3) and solving ILP (line 4).

The feedback from solving ILP with the global constraints can help us refine the learning of local probabilities by removing incorrect affixes (line 5). For instance, automatic extraction based on frequencies can include *-ers* as an English suffix. This is likely to be eliminated by ILP, since all occurrences of *-ers* can be explained away without adding a new affix by concatenating *-er* and *-s*, two very common suffixes. After refining the affix set, we remove all candidates that involve any affix discarded by ILP. This corresponds to reducing the size of $C(w)$ in Equation (3). We then train the log-linear model again using the newly-pruned candidate set. By doing so, we force the model to learn from better contrastive signals, and focus on affixes of higher quality, resulting in a new set of probabilities $\{p_{ij}\}$. This procedure is repeated until no more affixes are rejected.[5]

4 Experiments

We evaluate our model on three tasks: segmentation, morphological family clustering, and root detection. While the first task has been extensively studied in the prior literature, we consider two additional tasks to assess the flexibility of the derived representation.

4.1 Morphological segmentation

Data We choose four languages with distinct morphological properties: English, Turkish, Arabic, and German. Our training data consists of standard datasets used in prior work. Statistics for all datasets are summarized in Table 1. Note that for the Arabic test set, we filtered out duplicate words, and we reran the baselines to obtain comparable results.

Following Narasimhan et al. (2015), we reduce the noise by truncating the training word list to the top K frequent words. In addition, we train word vectors (Mikolov et al., 2013) to obtain cosine similarity features. Statistics for all datasets are summarized in Table 1.

Baselines We compare our approach against the state-of-the-art unsupervised method of Narasimhan et al. (2015) which outperforms a number of alternative approaches (Creutz and Lagus, 2005; Virpioja et al., 2013; Sirts and Goldwater, 2013; Lee et al., 2011; Stallard et al., 2012; Poon et al., 2009). For this baseline, we report the results of the publicly available implementation of the technique (*NBJ'15*), as well as our own improved reimplementation (*NBJ-Imp*). Specifically in *NBJ-Imp*, we expanded the original algorithm to handle compounding, along with sibling features as described in Section 3.2, making it essentially an ablation of our model without ILP and alternating training. We employ grid search to find the optimal hyperparameter setting.[6]

[4] For English and German, where non-concatenative transformations are possible such as deletion of ending *e* (*taking* → *take*), we also include them in *Affix*.

[5] Typically the model converges after 5 rounds

[6] $K \in \{2500, 5000, 10000\}$, number of automatically extracted affixes $\in \{100, 200, 300, 400, 500\}$

Algorithm 1 Morphological Forest Induction

Input: wordlist V

Output: Forest representation of V

Affix $\leftarrow$ *ExtractAffixes*(W) ▷ Extract common patterns as affixes from the wordlist
for $t \leftarrow 1$ to T **do** ▷ Alternating training for T iterations
  $p_{ij}^t \leftarrow$ *ContrastiveEstimation*$(W, Affix)$ ▷ Compute local probabilities, cf. Section 3.2
  $y^{*t}, F^t \leftarrow$ *ILP*(p_{ij}^t) ▷ Get indicators for affixes, and the forest, cf. Section 3.3
  PruneAffixSet$(Affix, y^{*t})$ ▷ Prune affix set using the output from ILP, cf. Section 3.4
return F^T

Language	Train #Words	Test #Words	WordVec #Words
English	MC-10 878K	MC-05:10 2212	Wikipedia 129M
Turkish	MC-10 617K	MC-05:10 2531	BOUN 361M
Arabic	Gigaword 3.83M	ATB 21085	Gigaword 1.22G
German	MC-10 2.34M	Dsolve 15522	Wikipedia 589M

Table 1: Data statistics: MC-10 = MorphoChallenge 2010 , MC:05-10 = aggregated from MorphoChallenge 2005-2010, BOUN = BOUN corpus (Sak et al., 2008), Gigaword = Arabic Gigaword corpus (Parker et al., 2011), ATB = Arabic Treebank (Maamouri et al., 2003). Duplicates in Arabic test set are filtered. Dsolve is the dataset released by Würzner and Jurish (2015), and for training German vectors, we use the pre-processed Wikipedia dump from (Al-Rfou et al., 2013).

We also include a supervised counterpart, which uses the same set of features as *NBJ-Imp* but has access to gold segmentation during training (we perform 5-fold cross-validation using the same data). We obtain the gold standard parent-child pairs required for training from the segmented words in a straightforward fashion.

Evaluation metric Following prior work (Virpioja et al., 2011), we evaluate all models using the standard *boundary precision and recall (BPR)*. This measure assesses the accuracy of individual segmentation points, producing IR-style *Precision, Recall* and *F1* scores.

Language	#Words	#Clusters	#Words per Cluster
English	75,416	20,249	3.72
German	367,967	28,198	13.05

Table 2: Data statistics for the family clustering task (CELEX). We only evaluate on English and German, since these are the languages MorphoChallenge has segmentations for.

Training For unsupervised training, we use the gradient descent method ADAM (Kingma and Ba, 2014) and optimize over the whole batch of training words. We use a Gurobi[7] solver for the ILP.

4.2 Morphological family clustering

Morphological family clustering is the task of clustering morphologically related word forms. For instance, we want to group *paint*, *paints* and *pain* into two clusters: {*paint, paints*} and {*pain*}. To derive clusters from the forest representation, we assume that all the words in the same tree form a cluster.

Data To obtain gold information about morphological clusters, we use CELEX (Baayen et al., 1993). Data statistics are summarized in Table 2. We remove words without stems from CELEX.[8]

Baseline We compare our model against *NBJ-Imp* described above. We select the best variant of our model and the base model based on their respective performance on the segmentation task.

Evaluation We use the metrics proposed by Schone and Jurafsky (2000). Specifically, let X_w

[7] http://www.gurobi.com/

[8] An example is *aerodrome*, where both *aero-* and *drome* are affixes.

Language	#Words	#Words (Test only)
English	1675	687
Turkish	1759	763
German	1747	749

Table 3: Data statistics for root detection task. Duplicate words are removed.

and Y_w be the clusters for word w in our predictions and gold standard, respectively. We compute the number of correct ($\mathcal{C}$), inserted ($\mathcal{I}$) and deleted ($\mathcal{D}$) words for the clusters as follows:

$$\mathcal{C} = \sum_{w \in W} \frac{|X_w \cap Y_w|}{|Y_w|}$$

$$\mathcal{I} = \sum_{w \in W} \frac{|X_w \setminus Y_w|}{|Y_w|}$$

$$\mathcal{D} = \sum_{w \in W} \frac{|Y_w \setminus X_w|}{|Y_w|}$$

Then we compute $precision = \frac{\mathcal{C}}{\mathcal{C}+\mathcal{I}}$, $recall = \frac{\mathcal{C}}{\mathcal{C}+\mathcal{D}}$, $F1 = 2\frac{precision \cdot recall}{precision+recall}$.

4.3 Root detection

In addition, we evaluate how accurately our model can predict the root of any given word.

Data We report the results on the Chipmunk dataset (Cotterell et al., 2015) which has been used for evaluating supervised models for root detection. Since our model is unsupervised, we report the performance both on the test set only, and on the entire dataset, combining the train/test split. Statistics for the dataset are shown in Table 3.

5 Results

In the following subsections, we report model performance on each one of the three evaluation tasks.

5.1 Segmentation

[9]We used cosine similarity features in all experiments. But the root forms of German verbs are rarely used, except in imperative sentences. Consequently they have barely trained word vectors, contributing to the low recall value. We suspect better treatment with word vectors can further improve the results.

[10]http://www.mathcracker.com/sign-test.php

Language	Method	BPR		
		P	R	F
English	*Supervised*	0.905	0.813	0.856
	NBJ'15	0.807	0.722	0.762
	NBJ-Imp	0.820	0.726	0.770
	Our model	0.838	0.729	0.780
	+ *Sibl*	0.796	0.739	0.767
	+ *Comp*	0.840	0.761	**0.799***
	+ *Comp, Sibl*	0.815	0.774	0.794
Turkish	*Supervised*	0.826	0.803	0.815
	NBJ'15	0.743	0.520	0.612
	NBJ-Imp	0.697	0.583	0.635
	Our model	0.717	0.577	0.639
	+ *Sibl*	0.698	0.619	**0.656***
	+ *Comp*	0.716	0.581	0.642
	+ *Comp, Sibl*	0.692	0.621	0.655
Arabic	*Supervised*	0.904	0.921	0.912
	NBJ'15	0.840	0.724	0.778
	NBJ-Imp	0.866	0.725	0.789
	Our model	0.848	0.769	0.806
	+ *Sibl*	0.829	0.787	**0.807***
	+ *Comp*	0.851	0.765	0.806
	+ *Comp, Sibl*	0.881	0.745	**0.807***
German[9]	*Supervised*	0.823	0.810	0.816
	NBJ'15	0.716	0.275	0.397
	NBJ-Imp	0.790	0.480	0.597
	Our model	0.774	0.540	0.636
	+ *Sibl*	0.711	0.514	0.596
	+ *Comp*	0.777	0.595	**0.674***
	+ *Comp, Sibl*	0.701	0.616	0.656

Table 4: Segmentation results for the supervised model and three unsupervised models: the state-of-the-art system *NBJ'15* (Narasimhan et al., 2015), our improved implementation of their system *NBJ-Imp* and our model. For our model, we also report results with different feature combinations. + *Sibl* and + *Comp* refer to addition of sibling and compounding features, respectively. Best hyperparameter values for unsupervised baselines (*NBJ'15*, *NBJ-Imp*) are chosen via grid search, while for our model, we use 10K words and top 500 affixes throughout. * implies statistical significance with $p < 0.05$ against the *NBJ-Imp* model using the sign test [10].

From Table 4, we observe that our model consistently outperforms the baselines on all four lan-

guages. Compared to *NBJ'15*, our model has a higher F1 score by $3.7\%, 4.4\%, 2.9\%$ and 27.7% on English, Turkish, Arabic and German, respectively. While the improved implementation *NBJ-Imp* benefits from the addition of compounding and sibling features, our model still delivers an absolute increase in F1 score, ranging from 1.8% to 7.7% over *NBJ-Imp*. Note that our model achieves higher scores even without tuning the threshold K or the number of affixes, whereas the baselines have optimal hyperparameter settings via grid search.

To understand the importance of global constraints (the last two terms of Equation (1)), we analyze our model's performance with different values of α and β (see Figure 3). The first constraint, which controls the size of the affix set, plays a more dominant role than the second. By setting $\alpha = 0.0$, the model scores at best 75.7% on English and 63.2% on Turkish, lower than the baseline. While the value of β also affects the F1 score, its role is secondary in achieving optimal performance.

The results also demonstrate that language properties can greatly affect the feature set choice. For fusional languages such as English, computing of sibling features is unreliable. For example, two descendants of the same parent *spot* – *spotless* and *spotty* – may not be necessarily identified as such by a simple sibling computation algorithm, since they undergo different changes. In contrast, Turkish is highly agglutinative, with minimal (if any) transformations, but each word can have up to hundreds of related forms. Consequently, sibling features have different effects on English and Turkish, leading to changes of -0.3% and $+2.1\%$ in F1 score respectively.

Understanding model behavior We find that much of the gain in model performance comes from the first two rounds of training. As Figure 4 shows, the improvement mainly stems from solving ILP in the first round, followed by training the log-linear model in the second round after removing affixes and pruning candidate sets. This is exactly what we expect from the ILP formulation – to globally adjust the forest by reducing the number of unique affixes. We find this to be quite effective – in English, out of 500 prefixes, only 6 remain: *de*, *dis*, *im*, *in*, *re*, and *un*. Similarly, only 72 out of 500 suffixes survive after this reduction.

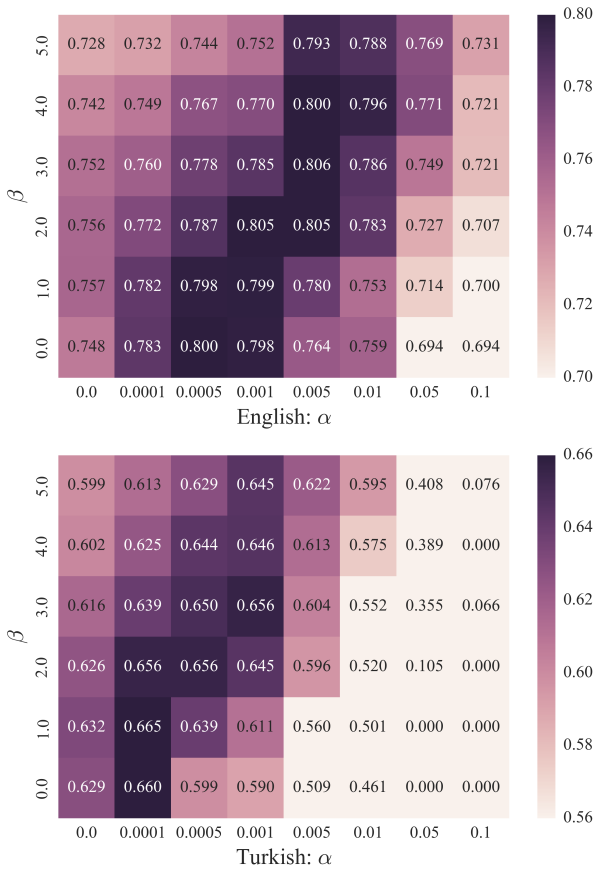

Figure 3: Heat maps of α and β for English and Turkish. Darker cells mean higher scores. Models used are $+Comp$ for English and $+Sibl$ for Turkish.

Robustness We also investigate how robust our model is to the choice of hyperparameters. Figure 3 illustrates that we can obtain a sizable boost over the baseline by choosing α and β within a fairly wide region. Note that α takes on a much smaller value than β, to maintain the two constraints ($|\textit{Affix}|$ and $\frac{|F|}{|V|}$) at comparable magnitudes.

Narasimhan et al. (2015) observe that after including more than $K = 10,000$ words, the performance of the unsupervised model drops noticeably. In contrast, our model handles training noise more robustly, resulting in a steady boost or not too big drop in performance with increasing training size (Figure 5). In fact, it scores 83.0% with $K = 40,000$ on English, a **6.0%** increase in absolute value over the baseline.

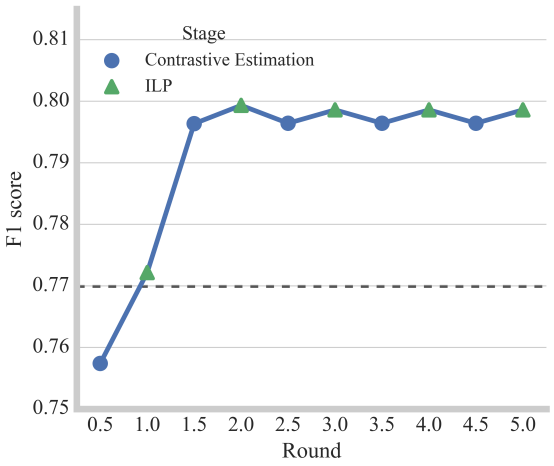

Figure 4: F1 score vs round of training, for + *Comp* on English. Training log-linear models and solving ILP are marked by circles and triangles respectively. Best result for *NBJ-Imp* is represented as a dashed horizontal line.

Qualitative analysis Table 5 shows examples of English words that our model segments correctly, while *NBJ'15* fails on them. We present them in three categories (top to bottom) based on the component of our model that contributes to the successful segmentation. The first category benefits from a refinement of the affix set, by removing noisy affixes, such as *-nce, -ch,* and *k-*. This leads to correct stopping as in the case of *knuckle* or induction of the right suffix, as in *divergence*. Further, a smaller affix set also leads to more concentrated weights for the remaining affixes. For example, the feature weight for *-ive* jumps from 0.06 to 0.25, so that the derivation *negative* $\rightarrow$ *negate* is favored, as shown in the second category. Finally, the last category lists some compound words that our model successfully segments.

5.2 Morphological family clustering

We show the results for morphological family clustering in Table 6. For both languages, our model increases $precision$ by a wide margin, with a modest boost for $recall$ as well. This corroborates our findings in the segmentation task, where our model can effectively remove incorrect affixes while still encouraging words to form tight, cohesive families.

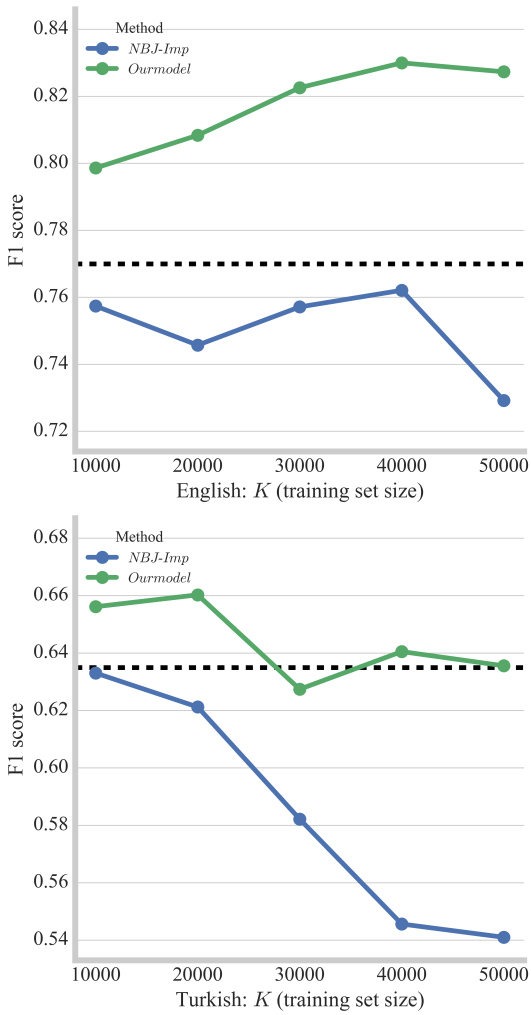

Figure 5: Performance using bigger training sets. $+Comp$ for English and $+Sibl$ for Turkish. Dashed lines represent the best results for *NBJ-Imp* (with smaller training sets).

5.3 Root detection

Table 7 summarizes the results for the root detection task. Our model shows consistent improvements over the baseline on all three languages. We also include the results on the test set of two *supervised* systems: *Morfette* (Chrupala et al., 2008) and *Chipmunk* (Cotterell et al., 2015). *Morfette* is a string transducer while *Chipmunk* is a segmenter. Both systems have access to morphologically annotated corpora.

Our model is quite competitive against *Morfette*. In fact, it achieves higher accuracy for English and Turkish. Compared with *Chipmunk*, our model

NBJ-Imp	Our model
diverge-nce	*diverg-ence*
lur-ch	*lurch*
k-nuckle	*knuckle*
negative	*negat-ive*
junks	*junk-s*
unreserved	*un-reserv-ed*
gaslight-s	*gas-light-s*
watercourse-s	*water-course-s*
expressway	*express-way*

Table 5: Some English words that our model segments correctly which the unsupervised base model (*NBJ'15*) fails at.

Language	Method	P	R	F
English	*NBJ-Imp*	0.328	0.680	0.442
	Our model	0.895	0.715	**0.795**
German	*NBJ-Imp*	0.207	0.421	0.278
	Our model	0.471	0.484	**0.477**

Table 6: Results for morphological family clustering. P = precision, R = recall.

scores 0.65 versus 0.70 on English, bridging the gap significantly. However, the high accuracy for morphologically complex languages such as Turkish and German suggests that unsupervised root detection remains a hard task.

6 Conclusions

In this work, we focus on unsupervised modeling of morphological families, collectively defining a forest over the language vocabulary. This formulation enables us to incorporate both local and global properties of morphological assignment. The resulting objective is solved using Integer Linear Programming (ILP) paired with contrastive estimation. Our experiments demonstrate that our model yields consistent gains in three morphological tasks compared with the best published results.

Acknowledgement

We thank Tao Lei, Yuan Zhang and the members of the MIT NLP group for helpful discussions and feedback. We are also grateful to anonymous reviewers for their insightful comments.

Language	Method	Accuracy	Accuracy (Test only)
English	*NBJ-Imp*	0.590	0.595
	Our model	**0.636**	**0.649**
	Morfette	-	0.628
	Chipmunk	-	0.703
Turkish	*NBJ-Imp*	0.446	0.442
	Our model	**0.463**	**0.467**
	Morfette	-	0.268
	Chipmunk	-	0.756
German	*NBJ-Imp*	0.347	0.331
	Our model	**0.383**	**0.364**
	Morfette	-	0.438
	Chipmunk	-	0.674

Table 7: Results for root detection. Numbers for *Morfette* and *Chipmunk* are reported by Cotterell et al. (2015).

References

Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2013. Polyglot: Distributed word representations for multilingual NLP. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 183–192, Sofia, Bulgaria, August. Association for Computational Linguistics.

R Harald Baayen, Richard Piepenbrock, and Rijn van H. 1993. The CELEX lexical data base on CD-ROM.

Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2011. Global learning of typed entailment rules. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 610–619. Association for Computational Linguistics.

Grzegorz Chrupala, Georgiana Dinu, and Josef van Genabith. 2008. Learning morphology with Morfette. In *LREC*.

Yoeng-Jin Chu and Tseng-Hong Liu. 1965. On shortest arborescence of a directed graph. *Scientia Sinica*, 14(10):1396.

James Clarke and Mirella Lapata. 2008. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, pages 399–429.

Ryan Cotterell, Thomas Müller, Alexander Fraser, and Hinrich Schütze. 2015. Labeled morphological segmentation with semi-Markov models. *CoNLL 2015*, page 164.

Mathias Creutz and Krista Lagus. 2005. Inducing the morphological lexicon of a natural language from unannotated text. In *Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning (AKRR05)*, volume 1, pages 51–59.

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing (TSLP)*, 4(1):3.

Markus Dreyer and Jason Eisner. 2009. Graphical models over multiple strings. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1*, pages 101–110. Association for Computational Linguistics.

Jack Edmonds. 1967. Optimum branchings. *Journal of Research of the National Bureau of Standards B*, 71(4):233–240.

Manaal Faruqui, Ryan McDonald, and Radu Soricut. 2016. Morpho-syntactic lexicon generation using graph-based semi-supervised learning. *Transactions of the Association for Computational Linguistics*, 4:1–16.

Sharon Goldwater and Mark Johnson. 2004. Priors in Bayesian learning of phonological rules. In *Proceedings of the 7th Meeting of the ACL Special Interest Group in Computational Phonology: Current Themes in Computational Phonology and Morphology*, pages 35–42. Association for Computational Linguistics.

Sharon Goldwater, Thomas L Griffiths, and Mark Johnson. 2009. A Bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112(1):21–54.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Yoong Keok Lee, Aria Haghighi, and Regina Barzilay. 2011. Modeling syntactic context improves morphological segmentation. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, pages 1–9. Association for Computational Linguistics.

Mohamed Maamouri, Ann Bies, Hubert Jin, and Tim Buckwalter. 2003. Arabic treebank: Part 1 v 2.0. *Distributed by the Linguistic Data Consortium. LDC Catalog No.: LDC2003T06*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Jason Naradowsky and Kristina Toutanova. 2011. Unsupervised bilingual morpheme segmentation and alignment with context-rich hidden semi-Markov models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 895–904. Association for Computational Linguistics.

Karthik Narasimhan, Regina Barzilay, and Tommi Jaakkola. 2015. An unsupervised method for uncovering morphological chains. *Transactions of the Association for Computational Linguistics*, 3:157–167.

Robert Parker, David Graff, Ke Chen, Junbo Kong, and Kazuaki Maeda. 2011. Arabic gigaword fifth edition ldc2011t11. *Philadelphia: Linguistic Data Consortium*.

Nanyun Peng, Ryan Cotterell, and Jason Eisner. 2015. Dual decomposition inference for graphical models over strings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 917–927, Lisbon, Portugal, September. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Dan Roth and Wen-tau Yih. 2001. Relational learning via propositional algorithms: An information extraction case study. In *International Joint Conference on Artificial Intelligence*, pages 1257–1263.

Haşim Sak, Tunga Güngör, and Murat Saraçlar. 2008. Turkish language resources: Morphological parser, morphological disambiguator and web corpus. In *Advances in natural language processing*, pages 417–427. Springer.

Patrick Schone and Daniel Jurafsky. 2000. Knowledge-free induction of morphology using latent semantic analysis. In *Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning-Volume 7*, pages 67–72. Association for Computational Linguistics.

Kairit Sirts and Sharon Goldwater. 2013. Minimally-supervised morphological segmentation using adaptor grammars. *Transactions of the Association for Computational Linguistics*, 1:255–266.

Noah A Smith and Jason Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 354–362. Association for Computational Linguistics.

Benjamin Snyder and Regina Barzilay. 2008. Unsupervised multilingual learning for morphological segmentation. In *Proceedings of the 46th Annual Meeting on Association for Computational Linguistics*, pages 737–745. Association for Computational Linguistics.

Radu Soricut and Franz Och. 2015. Unsupervised morphology induction using word embeddings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1627–1637. Association for Computational Linguistics.

David Stallard, Jacob Devlin, Michael Kayser, Yoong Keok Lee, and Regina Barzilay. 2012. Unsupervised morphology rivals supervised morphology for Arabic MT. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 322–327. Association for Computational Linguistics.

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 Grönroos, Mikko Kurimo, et al. 2013. Morfessor 2.0: Python implementation and extensions for Morfessor baseline.

Kay-Michael Würzner and Bryan Jurish. 2015. Dsolve-morphological segmentation for German using conditional random fields. In *Systems and Frameworks for Computational Morphology*, pages 94–103. Springer.