Morphological Segmentation: An Unsupervised Method and Application to Keyword Spotting

by

Karthik Rajagopal Narasimhan

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2014

Signature redacted

Author .....
Department of Electrical Engineering and Computer Science
May 21, 2014

Signature redacted

Certified by .....
Regina Barzilay
Professor
Thesis Supervisor

Signature redacted

Accepted by .....
Leslie A. Kolodziejski
Chairman, Department Committee on Graduate Students

Morphological Segmentation: An Unsupervised Method and Application to Keyword Spotting

by

Karthik Rajagopal Narasimhan

Submitted to the Department of Electrical Engineering and Computer Science
on May 21, 2014, in partial fulfillment of the
requirements for the degree of
Master of Science in Computer Science and Engineering

Abstract

The contributions of this thesis are twofold. First, we present a new unsupervised algorithm for morphological segmentation that utilizes pseudo-semantic information, in addition to orthographic cues. We make use of the semantic signals from continuous word vectors, trained on huge corpora of raw text data. We formulate a log-linear model that is simple and can be used to perform fast, efficient inference on new words. We evaluate our model on a standard morphological segmentation dataset, and obtain large performance gains of up to 18.4% over an existing state-of-the-art system, Morfessor.

Second, we explore the impact of morphological segmentation on the speech recognition task of Keyword Spotting (KWS). Despite potential benefits, state-of-the-art KWS systems do not use morphological information. In this thesis, we augment a KWS system with sub-word units derived by multiple segmentation algorithms including supervised and unsupervised morphological segmentations, along with phonetic and syllabic segmentations. Our experiments demonstrate that morphemes improve overall performance of KWS systems. Syllabic units, however, rival the performance of morphological units when used in KWS. By combining morphological and syllabic segmentations, we demonstrate substantial performance gains.

Thesis Supervisor: Regina Barzilay
Title: Professor

Acknowledgments

First and foremost, I would like to thank my advisor, Regina Barzilay, for being a constant source of ideas, feedback and support. Regina has an intuitive sense of what research questions are important and how they tie up into the big scheme of things in NLP and Machine Learning. I have learnt a lot from my interactions with her, ranging from how to perform day-to-day research in a more structured manner, to air plants, that can grow without any soil! I also thank my collaborators from BBN Technologies Ltd. - Damianos Karakos, Rich Schwartz and Stavros Tsakalidis - for their help in running the Keyword Spotting experiments. Without them, the second portion of this thesis would not be possible. I express my gratitude to the Qatar Foundation and the National Science Foundation (NSF Award No.: IIS-0835652) for funding my stay at MIT.

These two years at MIT have been an amazing experience for me. I have had several enriching discussions on varied topics and fun times with my group mates - Nate, Tahira, Tao, Yonatan, Yoong-Keok, Yuan, and Zach. Special thanks must go to Branavan, for being a great mentor to me in my first year here and helping me understand several nuances of the research process. The amazing social community of CSAIL and EECS have made sure I never had trouble keeping up with friends, and taking breaks from research. At home, my roommates Ardavan, Diego and Julian are an endless supply of support, fun and entertainment through sailing, board games, and house parties, to name a few. I am grateful to my many friends at MIT who have ensured I have enough play to balance the workload here.

This thesis would not be possible without the constant love and support from my family - my mother, my father, my brother, and my grandmothers, aunts, uncles and cousins. They have been a great source of inspiration and have always encouraged me to aim higher and not rest on past achievements. To them, I dedicate this thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Morphological analysis, in linguistics, refers to the study and description of the internal orthographic structure of words. The words in a language are often the smallest units of syntax. However, there do exist regularities that are shared between many of them, through common rules that are part of the language's grammar. These are especially evident in the form of common affixes (prefixes, suffixes or infixes), that are used to identify plurality, tense, noun forms, and many other nuances of the grammar. For example, in English, the suffix *-s*, when added to a word, almost always creates a plural form, while the prefix *re-* usually represents repetition of an action verb.

Most languages have three kinds of morphological rules. The first kind, known as *inflectional* rules, generate variant forms of the same word. An example of an inflection is the morphing of the verb *create* to its past tense *created*. The other set of rules, called *derivational* rules, form new words from the existing words. These rules change a word's meaning considerably, such as the verb *create* to the noun *creator*. The final set of paradigms is that of *compounding*, where multiple words are fused together to create a compound word. For example, the words *dog* and *catcher* form the compound word *dogcatcher*.

As speakers of a language, we intuitively sense the morphological patterns in the words we use. We learn the function and purpose of various affixes and word forms, and can even predict the meaning of a previously unseen word using this knowledge. Computational morphological analysis aims at enabling computers to perform the same kind of assessment on words. There are various levels of morphological tasks, ranging from segmentation to

paradigm learning. In this thesis, we tackle the problem of morphological segmentation, which deals with breaking down words into minimal meaningful units called morphemes.

In terms of computational models, we usually have three kinds of approaches. Rule-based systems can be constructed, which require a great amount of morphological rules to be input to the system. This approach is both time-consuming and expensive, and requires linguistic experts on the language. Supervised approaches utilize gold segmentations to learn models using machine learning techniques. Though this approach is better than rule-based systems in terms of time and cost requirements, gold annotations are hard to come by for more than a handful of the world's languages. However, many languages nowadays do boast the availability of a significant amount of raw text data on the World Wide Web. Unsupervised approaches, that use "raw natural language text data to output a description of the morphological structure of the language of the input text with as little supervision (parameters, thresholds, human intervention, model selection during development) as possible"[10], are therefore a very attractive choice, since they are easily extensible to many languages.

1.1 Unsupervised Morphology

The task of unsupervised morphological analysis has received considerable attention in the NLP community. This is especially due to the fact that we have so many languages in the world, and morphological analysis plays an increasingly important role in many language processing applications. Using word morphology allows us to alleviate the problem of out-of-vocabulary (OOV) words in language models. In addition to solving the OOV problem, it provides a structured representation for words that can be utilized in several NLP tasks. Further, morphological analysis gives us a natural categorization of words into clusters, which can be leveraged in other NLP tasks like part-of-speech tagging, syntactic parsing, and machine translation.

We present a new method for unsupervised morphological segmentation that utilizes pseudo-semantic information in addition to orthographic cues. We make use of the semantic signals from continuous word vectors, trained on huge corpora of raw text data. We

formulate a log-linear model that is simple and can be used to perform fast, efficient inference on new words. We evaluate our model on a standard morphological segmentation dataset, and obtain large performance gains of up to 18.4% over an existing state-of-the-art system, Morfessor.

1.2 Morphological Segmentation in Keyword Spotting

Recent research has demonstrated that adding information about word structure increases the quality of translation systems and alleviates sparsity in language modeling [5, 9, 12, 26].

In this thesis, we study the impact of morphological analysis on the keyword spotting (KWS) task. The aim of KWS is to find instances of a given keyword in a corpus of speech data. The task is particularly challenging for morphologically rich languages as many target keywords are unseen in the training data. For instance, in the Turkish dataset [1] we use, from the March 2013 IARPA Babel evaluations, 36.06% of the test words are unseen in the training data. However, 81.44% of these unseen words have a morphological variant in the training data. Similar patterns are observed in other languages used in the Babel evaluations. This observation strongly supports the use of morphological analysis to handle out-of-vocabulary (OOV) words in KWS systems.

Despite this potential promise, state-of-the-art KWS systems do not commonly use morphological information. This surprising development can be due to multiple reasons, ranging from the accuracy of existing morphological analyzers to the challenge of integrating morphological information into existing KWS architectures. While using morphemes is likely to increase coverage, it makes recognition harder due to the inherent ambiguity in the recognition of smaller units. Moreover, it is not clear a priori that morphemes, which are based on the semantics of written language, are the appropriate segmentation units for a speech-based application.

We investigate the above hypotheses in the context of a standard KWS architecture [2]. We augment word lattices with smaller units obtained via segmentation of words, and use these modified lattices for keyword spotting. We consider multiple segmentation algorithms, ranging from near-perfect supervised segmentations to random segmentations,

along with unsupervised segmentations and purely phonetic and syllabic segmentations. Our experiments reveal the following findings:

- Using sub-word units improves overall performance of KWS systems.
- Syllabic units rival the performance of morphological units when used in KWS systems.
- Improving the accuracy of morphological segmentation beyond a certain level does not necessarily translate into improved KWS performance.
- Adding phonetic information improves the quality of morphological segmentation.
- Combining morphological, phonemic and syllabic segmentations gives better KWS results than either in isolation.

1.3 Background

1.3.1 Unsupervised Morphological Analysis

A survey by Hammarstrom and Borin [10] provides a good overview of various unsupervised approaches to morphological analysis, ranging from border and frequency-based methods, to grouping of morphologically related words, and using features and classes. Most prior work has focussed on orthographic properties of words to determine the morphological rules of the language. In the case of unsupervised segmentation, one particularly successful algorithm has been Morfessor [7]. Morfessor uses the Minimum Description Length (MDL) principle to discover morphemes in the language in an iterative fashion. However, they do not employ any semantic cues in their algorithm.

Work by Schone et al. [24] uses some notion of semantics to filter out incorrect morphological pairings. Their usage of semantics is limited to performing LSA on the words and utilizing a similarity metric between the learnt vectors. They then collect pairs of potential morphological variants (PPMVs), and then determine which of these are legitimate. This work uses the vectors learnt using LSA as a filtering step part of a pipeline. In contrast,

we utilize pseudo-semantic information jointly with orthographic information in a single model.

Log-linear models have been considered in unsupervised morphological segmentation previously by Poon et al. [21]. They jointly model a word and its segmentation using morphemes and their contexts as features, taking cues from the MDL principle. However, their learning and inference is quite involved, using contrastive estimation [25] and sampling techniques. Our method is much simpler in both the learning and inference steps.

1.3.2 Word Vectors

Word vectors are simply representations of words as vectors over $\mathbb{R}$. The interesting property of these vectors is that their values are continuous instead of discrete. This provides several opportunities to utilize these vectors to directly compare and contrast words. It has been shown that these vectors capture linguistic regularities, in the form of semantic similarities, and analogies[19]. Recent advances have made it possible to train these vectors on large datasets[18], which has resulted in better quality vectors, that exhibit these regularities quite strongly.

These word vectors have already found applications in several NLP tasks, including syntactic parsing, part-of-speech tagging and machine translation. However, to our knowledge, they haven't been utilized directly to induce morphological segmentations. The closest work related to ours is that of Luong et al.[16], which utilizes morphological analysis to obtain better word representations, by tying together words that are morphologically related to each other.

1.3.3 Applications of Morphological Analyzers

Prior research on applications of morphological analyzers has focused on machine translation, language modeling and speech recognition [9, 4, 12]. Morphological analysis enables us to link together multiple inflections of the same root, thereby alleviating word sparsity common in morphologically rich languages. This results in improved language model perplexity, better word alignments and higher BLEU scores. Recent work has demonstrated

that even morphological analyzers that use little or no supervision can help improve performance in language modeling and machine translation [5, 26].

Similar to existing work, we leverage morphological segmentation to reduce OOV rates in KWS. However, prior work has not explored the impact of the type and quality of segmentations on the performance of the target application, instead assuming that perfect morphemes are the optimal sub-word units. In contrast, we investigate segmentations produced by a range of models, including acoustic sub-word units. We demonstrate the value of using alternative segmentations instead of or in combination with morphemes. In addition to improving the performance of KWS systems, this finding may also benefit other applications that currently use morphological segmentation for OOV reduction.

Chapter 2

Unsupervised Morphological Segmentation

This chapter describes a new method to perform morphological segmentation using very weak to no supervision. We first start out with some definitions and follow it with a description of the model. We then provide details on experiments on the task of morphological segmentation.

The motivation for this work stems from the fact that morphologically similar words also share semantic similarity. This is a property shared by most languages in the world. For example, the word *player* in English, which is a morphological variant of *play* is also very similar in meaning to *play*. Figure 2.1 illustrates how the cosine similarity is a strong signal for predicting the correct parent for a word. We utilize this semantic signal in order to obtain better morphological segmentations.

Segment	Cosine Distance
p	0.095
pl	-0.037
pla	-0.041
play	**0.580**
playe	0.000
player	1.000

Table 2.1: Cosine distances between word vectors of various segments of the word *player* and the vector of *player*.

2.1 Model

The formation of a word can be thought of in the form of a morphological chain. We define a morphological chain as a sequence of words that start from a base word, and lead up to the morphological variant, via intermediate variants. At each step of the chain, an affix or a word is added to the previously created word in the chain to obtain a new word. This process continues until we generate the final word. For example, a morphological chain for the word *internationally* could be *nation* $\rightarrow$ *national* $\rightarrow$ *international* $\rightarrow$ *internationally*.

Our goal is to predict the morphological chain for a word, which can then be used to produce a morphological segmentation. We focus on modeling one step of the morphological chain, in which a word undergoes morphological morphing into a variant. We shall refer to the word, which undergoes a morphological change to create a new word, as the parent of the newly formed word. After learning the parameters, this model can be used to infer the correct parent given a child.

In order to obtain the entire chain for a word, we recursively predict parent words and stop when we predict a stop case. Note that at every stage, the parent word may have been morphed and hence, there are more candidates than just the ones obtained by splitting at every point in the child word. We can handle this case by using features in our log-linear model that represent such morph processes which alter the word. Examples in a language like English include repetition, deletion or modification of the last few characters in the parent word.

We use a log-linear framework for our model. A log-linear model consists of a set of features that form a feature vector $\phi : \mathcal{W} \times \mathcal{T} \rightarrow \mathbb{R}^d$ and a corresponding weight vector $\theta \in \mathbb{R}^d$. Here, $\mathcal{W}$ is the set of all words in the vocabulary and $\mathcal{T}$ is the set of all parent words/candidates. We define the probability of a particular word-parent pair $(w \in \mathcal{W}, t \in \mathcal{T})$ as $P(w,t) \propto e^{\theta \cdot \phi}$. Therefore, we have the conditional probability of a parent t given a word w:

$$P(t|w) = \frac{e^{\theta \cdot \phi}}{\sum_{t' \in T_w} e^{\theta \cdot \phi}}$$

A log-linear model is especially suited to our direction of attacking the problem since it allows an easy, efficient way of incorporating several views of the data. For training,

we use heuristics to select the subspace of parents for each word towards which we move probability mass through optimization techniques. The heuristics include combinations of orthographic and pseudo-semantic cues that help in choosing good parent candidates. These are described in more detail in Section 2.1.2.

2.1.1 Learning

We now describe the log-linear framework used for learning and inference. We formulate the learning problem as an optimization one. Given our heuristically selected set of word-parent pairs, we aim to find the model weights that maximize the log-likelihood of the pairs. This reduces to maximizing the following objective X (with a regularization term):

$$\max_{\theta} \sum_{w \in W} \left[log \sum_{t \in T_H} e^{\theta \cdot \phi} - log \sum_{t' \in T_w} e^{\theta \cdot \phi} \right] - \lambda_1 ||\theta||^2 \tag{2.1}$$

where T_H refers to heuristically selected parents for word w, T_w refers to the set of all possible parents for w, and W is the set of words. The corresponding gradient can be derived as:

$$\frac{\partial X}{\partial \theta_j} = \sum_{w \in W} \left[\frac{\sum_{t \in T_H} \phi_j \cdot e^{\theta \cdot \phi}}{\sum_{t \in T_H} e^{\theta \cdot \phi}} - \frac{\sum_{t' \in T_w} \phi_j \cdot e^{\theta \cdot \phi}}{\sum_{t' \in T_w} e^{\theta \cdot \phi}} \right] - 2\lambda_1 \theta_j \tag{2.2}$$

We learn the model parameters by optimizing the expected log probability of the entire lexicon using LBFGS-B[3], a gradient-based optimization technique. LBFGS-B iteratively climbs the gradient in an intelligent manner, and converges to the function optimum relatively quickly. We compute the above function and gradient at every iteration of the algorithm.

2.1.2 Heuristics

The heuristics we use are combinations of orthographic and pseudo-semantic cues to prune out highly impossible parents given a word. In all cases, we restrict the parent candidates to the cases where the parent word is strictly smaller than the word itself. For words where no parent passes the heuristic, we consider that as a stop case. We use the following component heuristics to approve a parent candidate:

- Does the parent appear frequently in the unannotated wordlist?
- Is the cosine of the angle between the word vectors of the word and its parent above a certain threshold?
- Is the length of the parent at least half the length of the word?

We use several combinations of these component heuristics in our experiments and show the effect each one has on the results. Note that just considering all substrings of a word is not enough to get a good set of parent candidates. This is due to repetition, modification and deletion effects that can occur during a morphological transformation. For example, we drop the 'e' at the end in deriving *creating* from *create*. We take these effects into account while generating the parent candidates by including a separate generation process for each such transformation type. So, considering our previous case, for the word *creating*, we would generate all strings by adding one character to the fragment *creat* and then add to the parent candidate list if the generated string exists in our wordlist.

At a higher level, we are using human inductive bias as a form of weak supervision. The intuition behind the approach can be viewed as learning the model parameters using a set of 'silver' (noisy) annotations obtained from the heuristic as opposed to a single gold annotation per training example in a supervised case. The analogy of the supervised case can be easily observed by considering the following objective it uses:

$$\max_{\theta} \sum_{w \in W} \left[log \frac{e^{\theta \cdot \phi}}{\sum_{t' \in T} e^{\theta \cdot \phi}} \right] = \max_{\theta} \sum_{w \in W} \left[log\ e^{\theta \cdot \phi} - log \sum_{t' \in T} e^{\theta \cdot \phi} \right] \tag{2.3}$$

where t_w refers to the gold parent for the word w. Thus, in the supervised case, the parameters are adjusted to maximize the probability of the observed word-parent pairs, while in our case, we maximize the probability of a set of parent candidates for each word, obtained heuristically. The model we formulate is capable of learning good feature weights from this 'noisy' data.

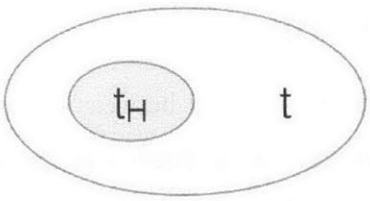

Figure 2-1: The model aims at concentrating the probability mass in the heuristic set t_H of parents for a word. t is the entire space of plausible parent words.

2.2 Features

The features we use in our model are derived from orthographic cues such as prefixes and suffixes, and pseudo-semantic indicators like the cosine similarity between word vectors. We now provide a description of each type of feature used. For the purposes of notation, these features (except for the stop case) are for a given word-parent pair (w, t). We refer to their corresponding word vectors as $\vec{w}$ and $\vec{t}$.

2.2.1 Affixes

First, we use indicator features for the affix involved in the generation of the word w from the parent t. We compute the affix as the characters in w that are extraneous to t. We have separate features for prefixes and suffixes, depending on whether t overlaps with the end or the beginning of w, in the respective cases. The intuition behind these features is to learn the suffixes and prefixes regularly used in the language, through their multiple occurrences in the data. This can help us identify potential suffixes given a new word.

In addition to indicator affix features, we also employ features that are the cartesian product (cross) of the affix with the context unigrams and bigrams (i.e. the characters appearing in the word to the left of a suffix, or to the right of a prefix in the word). We preprocess the unannotated wordlist and determine the most frequently occurring prefixes and suffixes using a threshold. Note that we have no knowledge of the prefixes or suffixes in the words. Instead, we consider any sequence that can be derived from a (w, t) pair such that w and t are in the wordlist, as a valid prefix candidate. If an affix does not occur in this list, we replace it with an unknown (UNK) tag during the feature computation.

2.2.2 Word Vector Similarity

Secondly, we consider the cosine similarity between the word vectors of the word and the parent,$\vec{w}$ and $\vec{t}$. We utilize the cosine directly as the feature value, banking on the observation that pairs of semantically similar words have higher cosine values between their vectors. Thus, this measure is often a very good signal for selecting a good parent. It is worth noting that this is the only non-binary feature in our model.

2.2.3 Presence in Wordlist

Another set of binary features we use serve to check if the parent t exists in the unannotated wordlist. We take the cartesian product of this information with a binned cosine similarity value between $\vec{w}$ and $\vec{t}$. This feature serves to capture the trade-off between choosing a parent that occurs in our training wordlist versus choosing one whose vector has a high cosine similarity with $\vec{w}$.

In addition, we also include features that represent the binned values of the frequency of the parent in the training wordlist. If the parent does not appear in the wordlist, we use an OOV feature for that case. This helps capture the relative importance of a parent using its frequency in the language. A word occurring more frequently is more likely to be a parent.

2.2.4 Modification processes

To capture steps in the chain that modify the parent word while adding affixes, we use a set of dedicated features. Modifications we handle are repetition of the last character in the parent (*planning*), deletion of the last character in the parent (*deciding*), and modification of the last character of the parent (*carried*). We model the repetition process using an indicator feature that catches the repeating character. For the modification case, we use another indicator feature that tracks the original and the new character in the parent. And, in the deletion case, the indicator feature tracks the character that was deleted. In all cases, we also cross the indicator features with unigram and bigram character contexts.

Feature type	Word Pair	Feature	Value
Affix	(painter, paint)	*suffix*=er	1
Cosine	(painter, paint)	$\vec{w} \cdot \vec{t}$	0.58
Wordlist	(painter, paint)	*InVocab*=1 $\times$ $\vec{w} \cdot \vec{t} > 0.5$	1
	(painter, paint)	*freq(t)* $/100 > 62$	1
	(painter, paint)	*freq(t)* $/10 > 620$	1
Repetition	(planning, plan)	*Repeat=n*	1
Deletion	(deciding, decide)	*Delete=e*	1
Modification	(carried, carry)	*Modify=(y,i)*	1
Compound	(watchtower, tower)	*Compound=1*	1
	(watchtower, tower)	*Compound=1* $\times$ *bigram=er*	1

Table 2.2: Example of various types of features used in the model. All features except the cosine similarity are binary.

2.2.5 Compound Word

The next set of features capture the structure of compound words like *watchtower*. For a given pair (w, t), if t and the corresponding affix exist in the training wordlist, we activate the basic *compound word* feature, which is an indicator. We also cross this feature with the ending character unigram and bigram of the word to obtain new features. We include this set of features specifically to capture compound words, since they do not have affixes. Hence, in the absence of this feature, they are likely to Note that while calculating the features in the case of a compound word, we remove the affix features defined above, as the affixes in this case are likely to be UNK.

2.2.6 Stop Conditions

We also have the stop case where we do not want to proceed with the chain. To accommodate this, we include features such as the length of the word, as well as the starting and ending character unigrams and bigrams. We aim to avoid stopping at words which have potential suffixes or prefixes at their edges. On the other hand, if the unigrams and bigrams at the word boundaries are unlikely to be prefixes or suffixes in the language, we would like to stop the chain from proceeding further.

2.3 Inference

During inference, we first predict a good morphological chain for a given word. We then use this chain to output a morphological segmentation for the word. In the learning phase described above, we estimate parameters for single edges of the chain i.e for a word-parent pair. Hence, in order to predict a chain, we require several applications of the learnt predictors. We infer the morphological chain recursively as described below.

Given a word, we consider all possible parents, along with the stop case. This is done by first considering all possible splits of the word, and choosing either the left or right part as the parent. In addition to this, we also generate parent candidates for the 3 types of modification processes that introduce changes to the parent (viz. repetition, deletion and modification). We also add the stop case as an alternative choice. We use the model weights to estimate the probability multinomial over these choices and then select the parent with the maximum probability if it exists, or the stop case otherwise. We then predict the next parent in the chain, or stop if the stop case is chosen.

$$parent_{pred} = \arg\max_{t \in \mathcal{P}} e^{\theta \cdot \phi} \tag{2.4}$$

Generating the segmentation from the morphological chain is relatively straightforward and can be done by computing longest common substrings over the word pairs and placing segmentation points appropriately. We provide pseudocode for the prediction processes below in algorithms 1 and 2.

2.4 Experiments

This section details experiments we perform to evaluate our system on the task of morphological segmentation. We first start with a description of the data, and provide statistics. We then describe our experiments and provide quantitative results. We end the section with an analysis of the results.

Algorithm 1 Procedure to predict a parent for a word

```
1: procedure PREDICT(word)
2:     candidates ← GETCANDIDATES(word)
3:     bestScore ← 0
4:     bestType ← NORMAL
5:     parent ← word
6:     for (candidate, type) ∈ candidates do
7:         features ← GETFEATURES(word, candidate)
8:         score ← MODELSCORE(features)
9:         if score > bestScore then
10:            bestScore ← score
11:            bestType ← type
12:            parent ← candidate
13:    return parent, bestType
```

Algorithm 2 Procedure to predict a morphological chain

```
1: procedure GETMORPHOLOGICALCHAIN(word)
2:     parent, type ← PREDICT(word)
3:     if parent = word then return [(word, STOP)]
4:     return GETMORPHOLOGICALCHAIN(parent) + [(word, type)]
```

2.4.1 Data

We perform experiments on data from MorphoChallenges 2005-2010[1]. The data from MorphoChallenge is a standard dataset used for the task of morphological segmentation in the unsupervised case. The data consists of unannotated word lists which are to be used for training and gold annotated segmentations, which we use for testing. We run experiments on two of these languages - English and Turkish. The statistics of the data are in table 2.3.

For obtaining the word vectors, we train the word2vec tool [20] on an entire dump of Wikipedia (for English) and the Boun corpus [23] (for Turkish). We use 200-dimensional word vectors in all our experiments.

Lang	Train words	Test words
English	878k	2218
Turkish	617k	2534

Table 2.3: Statistics of the data from MorphoChallenges 2005-10

[1] http://research.ics.aalto.fi/events/morphochallenge2010/

Lang	Method	Prec	Recall	F-1
English	Morfessor	0.814	0.552	0.658
	Our model	0.742	0.693	0.717
Turkish	Morfessor	0.827	0.362	0.504
	Our model	0.603	0.592	0.597

Table 2.4: Results on morphological segmentation using data from MorphoChallenge. The scores are calculated across all segmentation points in the test data

Heuristic	Prec	Recall	F-1
No heuristic	0.232	0.238	0.235
$ParentFreq > 5 + 2\|parent\| > \|word\|$	0.722	0.682	0.702
$ParentFreq > 5 + Cosine > 0.0 + 2\|parent\| > \|word\|$	0.730	0.677	0.702
$ParentFreq > 5 + Cosine > 0.1 + 2\|parent\| > \|word\|$	0.732	0.642	0.684
$ParentFreq > 5 + Cosine > 0.2 + 2\|parent\| > \|word\|$	0.812	0.590	0.684
$ParentFreq > 5 + Cosine > 0.3 + 2\|parent\| > \|word\|$	**0.890**	0.539	0.671
$ParentFreq > 50 + Cosine > 0.0 + 2\|parent\| > \|word\|$	0.742	**0.693**	**0.717**

Table 2.5: Effect of various heuristics on model performance (English). The runs used words in the wordlist with frequency > 1000 for training. Bold represents the best scores

2.4.2 Analysis

We evaluate the performance of our model on individual segmentation points, which is standard in this task. We report Precision, Recall and F-1 scores for Morfessor and our model in table 2.4. We conducted several runs of Morfessor 2.0[2] with varying parameters, and chose the run with the best performance on the dataset.

From the table, we can see that our method significantly outperforms Morfessor, with a relative gain of 8.9% and 18.4% in F-score on English and Turkish, respectively. Our model has a much higher recall than Morfessor with a small drop in precision.

We also perform several experiments on the English dataset with varying heuristics to examine its effect. Table 2.5 shows that the heuristics are important for learning the model. However, between different heuristics, the F-1 score performance does not vary by much. The heuristics do impact the precision and recall scores, with a stricter heuristic resulting in higher precision scores.

In addition, we also performed multiple runs of the algorithm with varying sizes of the

[2]http://www.cis.hut.fi/projects/morpho/morfessor2.shtml

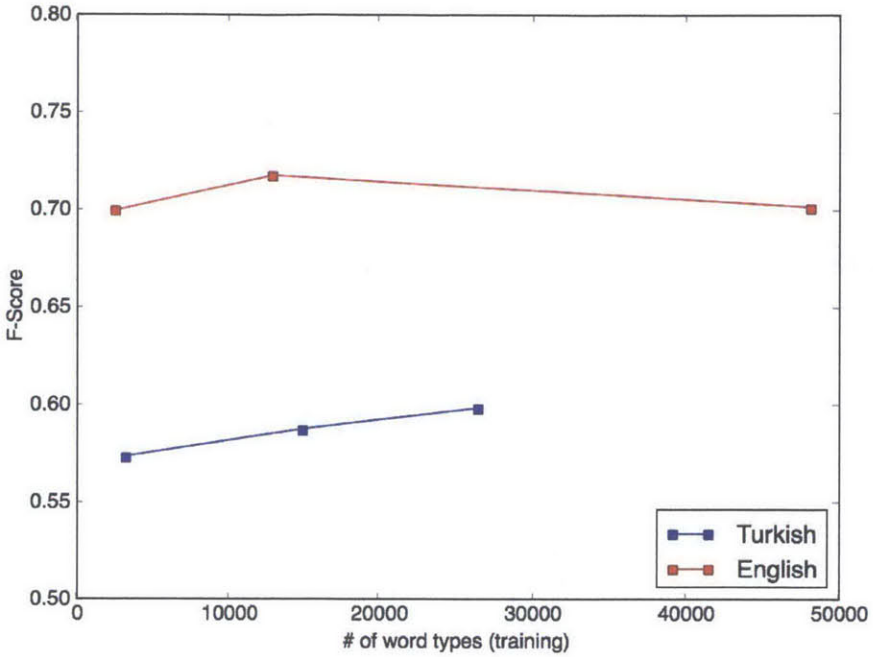

Figure 2-2: Effect of training data on F-1 scores for English and Turkish. The data size was varied using different frequency thresholds for the words

training dictionary. We varied the sizes by choosing different frequency thresholds for the words. From figure 2-2, we see that performance of the model increases in general with increasing dictionary size. We should note that the training time is also proportional to the size of the dictionary, which is evident from the optimization criterion.

Chapter 3

Morphological Segmentation in Keyword Spotting

In this chapter, we study the impact of morphological analysis on the keyword spotting (KWS) task. The aim of KWS is to find instances of a given keyword in a corpus of speech data. The task is particularly challenging for morphologically rich languages as many target keywords are unseen in the training data. For instance, in the Turkish dataset [1] we use, from the March 2013 IARPA Babel evaluations, 36.06% of the test words are unseen in the training data. However, 81.44% of these unseen words have a morphological variant in the training data. Similar patterns are observed in other languages used in the Babel evaluations. This observation strongly supports the use of morphological analysis to handle out-of-vocabulary (OOV) words in KWS systems.

Despite this potential promise, state-of-the-art KWS systems do not commonly use morphological information. This surprising development can be due to multiple reasons, ranging from the accuracy of existing morphological analyzers to the challenge of integrating morphological information into existing KWS architectures. While using morphemes is likely to increase coverage, it makes recognition harder due to the inherent ambiguity in the recognition of smaller units. Moreover, it is not clear a priori that morphemes, which are based on the semantics of written language, are the appropriate segmentation units for a speech-based application.

We investigate the above hypotheses in the context of a standard KWS architecture [2].

We augment word lattices with smaller units obtained via segmentation of words, and use these modified lattices for keyword spotting. We consider multiple segmentation algorithms, ranging from near-perfect supervised segmentations to random segmentations, along with unsupervised segmentations and purely phonetic and syllabic segmentations. Our experiments reveal the following findings:

- Using sub-word units improves overall performance of KWS systems.
- Syllabic units rival the performance of morphological units when used in KWS systems.
- Improving the accuracy of morphological segmentation beyond a certain level does not necessarily translate into improved KWS performance.
- Adding phonetic information improves the quality of morphological segmentation.
- Combining morphological, phonemic and syllabic segmentations gives better KWS results than either in isolation.

3.1 Segmentation Methods

We now describe the different types of segmentations we consider in our experiments. Table 3.1 shows examples of these segmentations for the Turkish word *takacak*.

3.1.1 Supervised Morphological Segmentation

An ideal scenario would be where we could use gold morphological segmentations in our analysis. However, since such annotations are not available for our corpus [1], we use a resource-rich supervised system as a proxy. As training data for this system, we use the MorphoChallenge 2010 corpus[1] which consists of 1760 gold segmentations for Turkish.

We consider two supervised frameworks for filtering. Each supervised system construct segmentations in two stages. In the first stage, which is common in both systems, we use

[1] http://research.ics.aalto.fi/events/morphochallenge2010/

Sub-word units	Example
Morphemes	tak - acak
Random	t - aka - c - a - k
Phones	t - a - k - 1v - dZ - a - k
Syllables	ta - k1v - dZak

Table 3.1: Segmentations of the word *takacak* into different types of sub-word units.

Feature	Example
morpheme unigrams	tak, acak
morpheme bigram	⟨tak, acak⟩
phonemic seq. unigrams	t.a.k., 1v.dZ.a.k.
phonemic seq. bigram	⟨t.a.k., 1v.dZ.a.k.⟩
number of morphemes	2
morpheme lengths	3, 4

Table 3.2: Example of features for the segmentation *tak-acak* used in the supervised filters. Each phoneme is followed by a dot for clarity.

a Finite-State-Transducer-based morphological parser [6] that generates a set of candidate segmentations, leveraging a large database of Turkish roots and affixes. This stage tends to over generate, segmenting each word in eight different ways on average. In the next stage, we filter the resulting segmentations using one of two supervised systems (described below) trained on the MorphoChallenge corpus.

In the first approach, we use a binary log-linear classifier to predict the correctness of each segmentation hypothesis. For each word, this classifier may select multiple segmentations as correct, or rule out all the alternatives. In the second approach, to control the number of valid segmentations, we train a log-linear ranker that orders the alternative segmentations for a word in decreasing order of likelihood. In our training corpus, each word has on average 2.5 gold segmentations. Hence, we choose the top two segmentations per word from the output of the ranker to use in our KWS system. Table 3.2 contains a list of the features used by the supervised systems.

Since the supervised systems look at the entire segmentation at once, we can encode features that go beyond individual boundaries, such as the total number of morphemes in the segmentation. This global view distinguishes our classifier/ranker from traditional approaches that model segmentation as a sequence tagging task [22, 14, 13]. Another depar-

ture of our approach is the use of phonetic information, in the form of phonemic sequences corresponding to the morpheme unigrams and bigrams. The phonemic sequences for words are obtained using a publicly available Text-to-Phone (T2P) system [15].

3.1.2 Unsupervised Morphological Segmentation

We employ a language-agnostic unsupervised system Morfessor [7] which achieves state-of-the-art unsupervised performance in the MorphoChallenge evaluation. Morfessor uses probabilistic generative models with sparse priors which are motivated by the Minimum Description Length (MDL) principle. The system derives segmentations from raw data, without reliance on extra linguistic sources. It outputs a single segmentation per word.

3.1.3 Random Segmentation

As a baseline, we include random segmentations in our analysis, where we mark a segmentation boundary at each character position in a word with a fixed probability p. For comparison purposes, we consider two types of random segmentations that match the supervised morphological segmentations in terms of the number of uniques morphemes and the average morpheme length, respectively. These segmentations are obtained by adjusting the segmentation probability p appropriately.

3.1.4 Acoustic-Based Segmentation

In addition to letter-based segmentation, we also consider other sub-word units that stem from word acoustics. In particular, we consider segmentation using phones and syllables, which are available for the Babel data we work with.

3.2 Keyword Spotting

In this section, we first briefly describe the basic KWS architecture, and then focus on modifications we introduce to account for sub-word units.

The keyword spotting system used in this work follows, to a large extent, the pipeline of [2]. Using standard speech recognition machinery, the system produces a detailed lattice of word hypotheses. The resulting lattice is used to extract keyword hits with nominal posterior probability scores. By design, the system employs several scoring methods. For example, we can use whole-word extraction methods for words in vocabulary, but rely on phonetic models for OOV words.

We modify this basic architecture to incorporate sub-word units. There are two main changes in our system. First, we use morphemes instead of whole-words in the decoding lexicon. Second, we represent keywords as sequences of morphemes in all possible combinations. The major modifications to the pipeline are as follows:

1. The segmented lexicon is used in order to represent the acoustic transcripts in terms of morphemes. Subsequently, a trigram language model is trained based on sequences of such morphemes. The dictionary used in the language model is represented as strings of compounded morphemes (for the whole words) and is augmented with the morphemes themselves.

2. A text-to-phone mapping is learned using the provided pronunciations. This mapping is subsequently applied to all morphemes which are not part of the original dictionary. This allows individual morphemes to be recognized by the ASR system; when they appear next to other morphemes in the lattices, they have the potential to make up new (out-of-vocabulary) words.

3. The ASR system generates lattices which contain morphemes, as well as compounded versions of these morphemes. The compounded tokens are split into their constituent morphemes, by adding new arcs in the lattice. Finally, the lattice is converted to a confusion network [17], consisting of morphemes only.

4. The given keywords are represented as sequences of morphemes in all possible combinations. For each sequence of matching arcs, the posteriors of these arcs are multiplied together to form the score of the detection record (hit). A post-processing step adds up (or takes the max of) the scores of all hits of each keyword which have significant overlap in time.

Finally, the hit lists are processed by the score normalization and combination method described in Karakos et al. [11].

3.3 Experimental Setup

3.3.1 Data

The segmentation algorithms described in Section 3.1 are tested using the setup of the KWS system described in Section 3.2. Our experiments are conducted using the IARPA Babel Program Turkish language collection release IARPA-babel105b-v0.4 [1]. The dataset contains audio corpora and a set of keywords. The training corpus for KWS consists of 10 hours of speech, while the development and test sets have durations of 10 and 5 hours, respectively. We evaluate KWS performance over the OOV keywords in the data, which are unseen in the training set, but appear in the development/test set. The development set comprises of 403 OOV keywords, while the test consists of 226 OOV keywords.

In our experiments, we consider the pre-indexed condition of keyword search, which means that the keywords are known only after the decoding of the speech has taken place.

3.3.2 Evaluation Measures

We consider two different evaluation metrics. To evaluate the accuracy of the different segmentations, we compare them against gold segmentations from the MorphoChallenge data. This set consists of 1760 words, which are manually segmented. We use a measure of word accuracy (**WordAcc**), which captures the accuracy of all segmentation decisions within the word. If one of the segmentation boundaries is wrong in a proposed segmentation, then that segmentation does not contribute towards the WordAcc score. We use 10-fold cross-validation for computing accuracies for the supervised methods, while we use the entire set for comparing unsupervised and acoustic-based methods.

We evaluate the performance of the KWS system using the Actual Term Weighted Value (ATWV) measure as described in [8]. This measure uses a combination of penalties for misses and false positives to score the system. The maximum score achievable is 1.0,

if there are no misses and false positives, while the score can be lower than 0.0 if there are a lot of misses or false positives. If we consider only the keywords present in the training vocabulary, the ATWV scores on the development and testing datasets are 0.385 and 0.400, respectively. These numbers give us an idea of the upper bound on the performance of the KWS system.

3.4 Results

Table 3.3 summarizes the performance of all considered segmentation systems in the KWS task. The quality of the segmentations compared to the gold standard is also shown. We summarize below our conclusions based on these results.

3.4.1 Effect of sub-units

We find that using sub-word units improves overall KWS performance. If we use a word-based KWS system, the ATWV score will be 0.0 since the OOV keywords are not present in the lexicon. Enriching our KWS system with sub-word segments yields performance gains for all the segmentation methods, including random segmentations. However, the observed gain exhibits significant variance across the segmentation methods. For instance, the gap between the performance of the KWS system using the best supervised classifier-based segmenter (*CP*) and that using the unsupervised segmenter (*U*) is 0.059, which corresponds to a 43.7% in relative gain. Table 3.3 also shows that while methods with shorter sub-units (*U*, *P*) yield lower OOV rate, they do not necessarily fare better in the KWS evaluation.

3.4.2 Syllabic units vs Morphological units

A surprising discovery of our experiments is the high performance of the syllabic segmentation-based KWS system (*S*). It outperforms all the alternative segmentations on the test set, and ranks second on the development set behind the supervised classifier-based system (*CP*). These units are particularly attractive as they can easily be computed from acoustic input and do not require any prior linguistic knowledge. We hypothesize that the granularity

of this segmentation is crucial to its success. For instance, a finer-grained phone-based segmentation (*P*) performs substantially worse than other segmentation algorithms as the derived sub-units are shorter and harder to recognize.

3.4.3 Extent of improvement

We also conclude that improving morphological accuracy beyond a certain level does not necessarily translate into improved KWS performance. By analyzing Table 3.3, we observe that the relation between segmentation accuracy and KWS performance is not straightforward. Clearly, bad segmentations translate into poor ATWV scores, as in the case of random and unsupervised segmentations. However, gains on segmentation accuracy do not always result in better KWS performance. For instance, the ranker systems (*RP, RNP*) have better accuracies on MC2010, while the classifier systems (*CP, CNP*) perform better on the KWS task. Figure 3-1 provides a visualization of the lack of correlation between the different scores. This discrepancy in performance suggests that further gains can be obtained by optimizing morphological segmentation directly with respect to KWS metrics.

3.4.4 Phonetic information

Another conclusion we can draw is that adding phonetic information improves the quality of morphological segmentation. For all the morphological systems, adding phonetic information results in consistent performance gains. For instance, it increases segmentation accuracy by 4% when added to the classifier (see *CNP* and *CP* in table 3.3). The phonetic information used in our experiments is computed automatically using a T2P system [15], and can be easily obtained for a range of languages. This finding sheds new light on the relation between phonemic and morphological systems, and can be beneficial for morphological analyzers developed for other applications.

3.4.5 Combination Systems

Combining morphological, phonemic and syllabic segmentations gives better results than either in isolation. As table 3.3 shows, the best KWS results are achieved when syllabic and

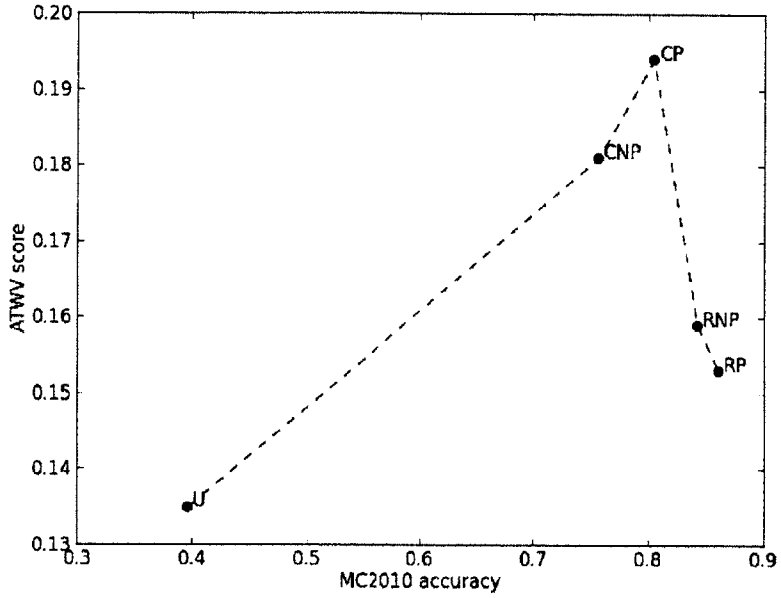

Figure 3-1: Plot of ATWV score vs MC2010 accuracy for various segmentations

morphemic systems are combined. The best combination system (*CP+P+S*) outperforms the best individual system (*S*) by 5.5%. This result suggests that morphemic, phonemic and syllabic segmentations encode complementary information which benefits KWS systems in handling OOV keywords.

Method	Unique units	Avg. unit length	Reduction in OOV (abs)	WordAcc	Dev ATWV	Test ATWV
Word-based for OOV words (W)	-	-	-	-	0.0	0.0
Phone-based (P)	51	1	36.06%	0.06%	0.099	0.164
Syllable-based (S)	2095	3.62	23.91%	10.29%	0.127	0.201
Classifier w/ phonetic info (CP)	18477	6.39	18.20%	80.41%	0.146	0.194
Classifier w/o phonetic info (CNP)	19094	6.42	21.50%	75.66%	0.133	0.181
Ranker w/ phonetic info (RP)	10286	5.62	16.86%	**86.03%**	0.104	0.153
Ranker w/o phonetic info (RNP)	10321	5.71	16.44%	84.19%	0.109	0.159
Unsupervised (U)	2390	5.44	22.45%	39.57%	0.080	0.135
RANDLen-Classifier	11687	6.39	0.73%	5.11%	0.061	0.086
RANDNum-Classifier	18224	3.03	8.56%	3.69%	0.111	0.154
RANDLen-Ranker	11651	5.62	1.94%	5.79%	0.072	0.136
RANDNum-Ranker	11713	6.13	1.15%	5.34%	0.081	0.116
CP + P	-	-	-	-	0.190	0.246
RP + P	-	-	-	-	0.150	0.210
CP + P + S	-	-	-	-	**0.208**	**0.257**
RP + P + S	-	-	-	-	0.186	0.249
Word-based for IV words	-	-	-	-	0.385	0.400

Table 3.3: Segmentation Statistics and ATWV scores on Babel data along with WordAcc on MorphoChallenge 2010 data for various segmentation methods. Absolute OOV reduction is from 36.06%. Higher ATWV scores are better. Best system scores are shown in bold. Combination systems are represented using the abbreviated forms of their components.

Chapter 4

Discussion

4.1 Contributions

We present two contributions in this thesis. First, we describe a new unsupervised algorithm for morphological segmentation that utilizes pseudo-semantic information, in addition to orthographic cues. We formulate a log-linear model with features to incorporate semantic cues from word vectors trained on large corpora. Using the notion of a morphological chain, we train the model to identify good word-parent pairs using heuristics to guide the learning of model weights. We demonstrate the effectiveness of our approach on data from MorphoChallenge, a standard morphological dataset, on two languages - English and Turkish. The model achieves significant gains in Segmentation Point F-Score, beating the baseline Morfessor system by 8.9% and 18.4% on English and Turkish, respectively. We also show that varying the heuristics gives us control over the precision and recall of the model.

Second, we explore the impact of morphological segmentation on the speech recognition task of Keyword Spotting (KWS). To investigate this issue, we augmented a KWS system with sub-word units derived by multiple segmentation algorithms. Our experiments demonstrate that morphemes improve the overall performance of KWS systems. Syllabic units, however, rival the performance of morphemes in the KWS task. Furthermore, we demonstrate that substantial performance gains in KWS performance are obtained by combining morphological, phonemic and syllabic segmentations. Finally, we also show that

adding phonetic information improves the quality of morphological segmentation.

4.2 Future Work

We recognize several areas for future investigation in this work. The inference scheme presented in the log-linear model is equivalent to performing a greedy search for a morphological chain, with access to an edge-scoring function. An interesting extension could be to experiment with other search approaches, such as Beam Search, to boost the chances of finding the correct chain. Another direction for research is modeling the entire morphological chain instead of single edges. This is computationally harder, but provides the opportunity to incorporate global chain-level features.

In the Keyword Spotting task, we have shown that producing morphemes according to the semantic and lexical criteria does not necessarily help the performance. A future direction could be to utilize feedback from the KWS system to produce sub-units that are optimized for the task. Another possible extension is to perform segmentation of the phonetic sequences directly, instead of words, to avoid the errors introduce by the text-to-phone (T2P) system.

Bibliography

[1] IARPA Babel. Turkish language collection release (iarpa-babel105b-v0.4), 2013.

[2] I. Bulyko, O. Kimball, M.-H. Siu, J. Herrero, and D. Blum. Detection of unseen words in conversational Mandarin. In *Proc. of ICASSP*, Kyoto, Japan, Mar 2012.

[3] Richard H Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

[4] Victor Chahuneau, Eva Schlinger, Noah A. Smith, and Chris Dyer. Translating into morphologically rich languages with synthetic phrases. In *EMNLP*, pages 1677–1687. ACL, 2013.

[5] Victor Chahuneau, Noah A. Smith, and Chris Dyer. Knowledge-rich morphological priors for bayesian language models. In *HLT-NAACL*, pages 1206–1215. The Association for Computational Linguistics, 2013.

[6] Çağrı Çöltekin. A freely available morphological analyzer for Turkish. In *Proceedings of the 7th International conference on Language Resources and Evaluation (LREC2010)*, pages 820–827, 2010.

[7] Mathias Creutz and Krista Lagus. 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 (AKRR)*, pages 106–113, 2005.

[8] J. G. Fiscus, J. Ajot, J. S. Garofolo, and G. Doddington. Results of the 2006 spoken term detection evaluation. In *Workshop on Searching Spontaneous Conversational Speech*, 2007.

[9] Nizar Habash. Four techniques for online handling of out-of-vocabulary words in arabic-english statistical machine translation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, HLT-Short '08, pages 57–60, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[10] Harald Hammarström and Lars Borin. Unsupervised learning of morphology. *Computational Linguistics*, 37(2):309–350, June 2011.

[11] D. Karakos, R. Schwartz, S. Tsakalidis, L. Zhang, S. Ranjan, T. Ng, R. Hsiao, G. Saikumar, I. Bulyko, L. Nguyen, J. Makhoul, F. Grezl, M. Hannemann, M. Karafiat, I. Szoke, K. Vesely, L. Lamel, and V.-B. Le. Score normalization and system combination for improved keyword spotting. In *Proc. ASRU 2013*, Olomouc, Czech Republic, 2013.

[12] Katrin Kirchhoff, Dimitra Vergyri, Jeff Bilmes, Kevin Duh, and Andreas Stolcke. Morphology-based language modeling for conversational arabic speech recognition. *Computer Speech and Language*, 20(4):589–608, 2006.

[13] Canasai Kruengkrai, Virach Sornlertlamvanich, and Hitoshi Isahara. A conditional random field framework for thai morphological analysis. In *LREC*, 2006.

[14] Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. Applying conditional random fields to japanese morphological analysis. In *In Proc. of EMNLP*, pages 230–237, 2004.

[15] Kevin Lenzo. Text-to-phoneme converter builder. `http://www.cs.cmu.edu/afs/cs.cmu.edu/user/lenzo/html/areas/t2p/`, 1998. Accessed: 2014-03-11.

[16] Thang Luong, Richard Socher, and Christopher Manning. Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[17] L. Mangu, E. Brill, and A. Stolcke. Finding consensus in speech recognition: word error minimization and other applications of confusion networks. *Computer Speech and Language*, 14(4):373–400, 2000.

[18] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[19] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013.

[20] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013.

[21] Hoifung Poon, Colin Cherry, and Kristina Toutanova. 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*, NAACL '09, pages 209–217, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[22] Teemu Ruokolainen, Oskar Kohonen, Sami Virpioja, and Mikko Kurimo. Supervised morphological segmentation in a low-resource learning setting using conditional random fields. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 29–37, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

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

[24] P.J. Schone. *Toward Knowledge-free Induction of Machine-readable Dictionaries*. University of Colorado, 2001.

[25] Noah A. Smith and Jason Eisner. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 354–362, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[26] David Stallard, Jacob Devlin, Michael Kayser, Yoong Keok Lee, and Regina Barzilay. Unsupervised morphology rivals supervised morphology for arabic mt. In *ACL (2)*, pages 322–327. The Association for Computer Linguistics, 2012.