Natural Language Engineering **1** (1): 1–25. Printed in the United Kingdom

Incorporating Word Embeddings in Unsupervised Morphological Segmentation

AHMET ÜSTÜN, BURCU CAN
The University of Groningen, Hacettepe University
The Netherlands, Turkey

(*Received 15 January 2019*)

Abstract

We study the effects of using semantic information in morphological segmentation, which is obtained from dense continuous vector space since words that are derived from each other will remain semantically related. We use mathematical models such as maximum likelihood estimate (MLE) and maximum a posteriori estimate (MAP) by incorporating semantic information obtained from the dense word vector representations. Our approach does not require any annotated data which makes it fully unsupervised and requires only a small amount of raw data and pre-trained word embeddings for training purposes.The results show that using dense vector representations helps in morphological segmentation especially for low resource languages. We present results for Turkish, English, and German. Our semantic MLE model outperforms other suggested unsupervised models for Turkish language. Our proposed models could be also used for any other low resource language with concatenative morphology.

1 Introduction

Morphological segmentation is the task of splitting words into smallest meaningful units, called *morphemes*. For example, the word *transformations* is split into *trans*, *form*, *ation*, and *s*. The task is crucial in many natural language processing applications such as information retrieval, machine translation, or sentiment analysis. Morphological segmentation mitigates the sparsity in these tasks. Especially in agglutinative languages like Turkish, the number of word forms is theoretically infinite (Hankamer, 1986). Character-based approaches have also been recently introduced to tackle with the sparsity problem in NLP tasks (Cao and Rei, 2016; Bojanowski et al., 2017).

Morphology is tightly coupled with semantics and syntax. Morphological derivation stipulates the protection of the semantic meaning of the derived word. For example, the words *sell-seller*, *believe-believer* are semantically related to each other, where one of them describes the action of the verb and the other one describes the performer of the same action. Therefore, each word is generated depending on the syntactic context within a sentence, e.g. actions generally follow actors in a SVO

(subject-verb-object) grammatical structure. Therefore, morphology is also tightly coupled with syntax.

The greater part of the work on unsupervised morphological segmentation makes use of the orthographic features without considering the semantics. In recent years, there have been studies that employ deep learning techniques in morphology to incorporates semantics with morphological structure (Cao and Rei, 2016; Cotterell and Schütze, 2015; Lazaridou et al., 2013). In these approaches, morphemes are used to learn the meaning of words compositionally based on the morpheme and word embeddings.

In this study, we propose two approaches. In the first approach, we detect the potential morpheme boundary points using word embeddings and use those potential morphemes in order to learn the final morphological segmentation of the words based on maximum likelihood estimate (MLE). Thus, the segmentation is performed in two stages.

In the second approach, we combine the semantic information with the likelihood using maximum a posteriori estimate (MAP) by employing semantic similarity between different word forms as prior information. We obtain the semantic features of words from the word embeddings that are learned by word2vec (Mikolov et al., 2013). Word meanings are represented in a low-dimensional vector space (i.e. 200 dimensions for Turkish and 300 dimensions for English and German in our experiments) and we use the cosine similarity between the word embeddings in order to compute the semantic similarity between different word forms to decide whether two words are morphologically derived/inflected from each other or not. The semantic relatedness between different word forms will be used as prior information to find the correct morpheme boundaries. For example, *inform* and *information* are semantically related enough to be counted as derived from each other, which leads to a valid morpheme boundary such as *inform+ation*. However, *car* and *care* are not semantically related and cannot be derived from each other. A similar intuition has also been used by Schone and Jurafsky (2001) (see Section 2).

Both of the methods we propose in this paper address agglutinative morphology where each morpheme expresses a single meaning. Agglutinative languages can produce very long words where each of them has its own meaning. However, the meaning deviation usually is not extensive and there is still semantic relatedness between the original word and the derived/inflected word. During inflection, the word's existing meaning remains the same because each inflectional morpheme has its own function such as person, tense, accusative case etc. During derivation the derived word still remains related to the original word. For example, *tuz+luk* in Turkish (means *saltcellar*) is derived from the word *tuz* (means *salt*) and both words are still closely semantically related. Here, we perform the experiments on Turkish which is an agglutinative language, on English which has a comparably poorer morphology, and on German which has an extensive usage of compounds. Therefore, we show how our proposed methods work on different morphological typologies. As far as we know, this is the first attempt that explicitly incorporate semantics in unsupervised morphological segmentation from the perspective of the agglutinative morphological typology.

The article is organized as follows: Section 2 addresses the related work on unsupervised morphological segmentation, section 3 describes the MLE and MAP models proposed for unsupervised morphological segmentation, section 4 explains the inference algorithm to learn the MAP model, section 5 presents the experimental results with comparison to other state-of-art models on morphological segmentation, and finally section 6 concludes the article with general findings of the article along with the potential future work.

2 Related Work

Morphological segmentation is one of the oldest tasks in natural language processing, that goes back to Harris (Harris, 1955, 1970b) with the term *letter successor* that defines the distributional properties of letters within in a word. The variety of letter successors in any position within a word would indicate that there could be a morpheme boundary at that point. In other words, when a set of words is inserted on a trie[1] structure, the branching occurs in the potential segmentation points (except the root where all words are rooted).

One of the well-known unsupervised morphological segmentation systems is Morfessor, which is actually a family of different models. Morfessor Baseline (Creutz and Lagus, 2005b) employs the Minimum Description Length (MDL) principle which was formerly applied by de Marcken (1996) and Goldsmith (2001). The MDL principle is based on the idea that the best hypothesis that explains the data is the one which leads to the best compression of the data. In Morfessor, the morpheme set that leads to the minimum corpus length is searched, which will give the potential morphological segmentation of each word. Morfessor CatMAP (Creutz and Lagus, 2007) extends the Morfessor Baseline by modeling the transition between the morpheme types (stem, prefix or suffix) using the hidden Markov models (HMMs).

Goldwater et al. (2006) present a two-stage Bayesian model, where morphemes are drawn from a multinomial distribution (i.e. *generator*) and tokens are drawn from a Pitman-Yor process to have a power-law distribution over word frequencies (i.e. *adaptor*).

Besides orthographic features, syntactic features have also been used in unsupervised morphological segmentation. Can and Manandhar (2010) learn morphological paradigms by using the syntactic categories obtained by context distribution clustering by Clark (2000). Lee et al. (2011) also incorporate part-of-speech (PoS) information in morphological segmentation for the Arabic language.

Semantics has also been used in unsupervised morphological segmentation. Schone and Jurafsky (2001) use Latent Semantic Analysis (LSA) to learn the semantic vectors of each word. A segmentation is proposed when the stem and stem+affix are semantically similar enough. We also follow the same intuition in this study,

[1] A trie is a tree-like data structure where the words can be retrieved by traversing the tree from the root till the leaf nodes. Each node refers to either a letter in the alphabet or a sequence of letters. If each node that is the only child is merged with its parent, then the trie is called radix tree.

but this time using neural word embeddings obtained by Mikolov et al. (2013) to measure the semantic similarity between different word tokens. However, word embeddings learned by various methods could also been incorporated in our proposed models.

Progress in learning word embeddings from neural networks has made the semantic information available in computational linguistics in the recent years. Narasimhan et al. (2015) introduce a log-linear model based on morphological chains where each word and its forms are defined as child-parent relations. For example, *respect→respectful→respectfulness* forms a morphological chain, where *respect* is the parent of *respectful* and *respectful* is the child of *respect*, and the same holds for *respectful→respectfulness*. Narasimhan et al. (2015) use semantic similarity based on word embeddings as a feature to find child-parent relationships in their log-linear model.

Soricut and Och (2015) observe morphological regularities within a word embedding space that is estimated by the SkipGram model of Mikolov et al. (2013). The study shows that word embedding space exhibits the morphological transformations between words. The morphological rules for adding a prefix or a suffix lead to a similar shift in the meaning of any word in the space. The authors use this information to create the morphological families of words which can be used for further morphological analysis without using any external morphological analyzer.

Our model is mostly similar to the works by Schone and Jurafsky (2001) and Narasimhan et al. (2015). We also utilize the semantic similarity between different word forms in order to learn morpheme boundary points. Our work is also similar to other works that use orthographic features (i.e. frequency of each morpheme in the corpus). We combine both orthographic and semantic features in the same model to learn morphological segmentation in an unsupervised framework.

Besides all the mentioned work, Morpho Challenge (Kurimo et al., 2011) was held as a competition for statistical machine learning algorithms designed for discovering morphemes that each word consists of. The competition was organized between 2005-2010 and many unsupervised models were introduced to the community for the morphological segmentation problem. The challenge provided publicly available datasets for several languages including English, German, Turkish, Arabic etc. The datasets involve training sets that include raw words along with their frequency information obtained from another corpus, test sets that include gold segmentations of a comparably smaller set of words, and development sets that are used for parameter tuning. The experiments reported here use the datasets made publicly available by Morpho Challenge, to be able to compare our work with other prominent unsupervised models in the literature.

3 Using Word Embeddings in Morphological Segmentation

We present two frameworks based on MLE and MAP for morphological segmentation. In the MLE approach, semantic similarity is used in a separate step to obtain the initial split points in each word, whereas in the MAP approach seman-

tic similarity is used as prior information that is integrated in the same learning framework.

3.1 MLE Model

In the MLE model, we learn segmentation in two steps (Üstün and Can, 2016). In the first step, we detect the potential morpheme boundary points within each word in the corpus. In the second step, we learn the final segmentation points by using the potential morphemes obtained from the first stage.

The initial segmentation is performed based on the semantic similarity between word forms that are derived from each other. For example, the substrings *recreat*, *recreation*, *recreational* are all semantically related to *recreationally*. We use the semantic relatedness similar to parent-child relations introduced in MorphoChain model (Narasimhan et al., 2015). We use the parent-child relations for the initial segmentation of the corpus. Word embeddings obtained from word2vec (Mikolov et al., 2013) are used analogously to compute the semantic relatedness between the prefixes of each word. The cosine similarity between the embeddings of two prefixes in n dimensional space is computed as follows:

$$\cos(v(w_1), v(w_2)) \quad = \quad \frac{|v(w_1) \cdot v(w_2)|}{\|v(w_1)\| \cdot \|v(w_2)\|} \tag{1}$$

where w_1 and w_2 denote the prefixes, $v(w_1)$ denotes the word embedding of w_1, and $v(w_2)$ denotes the word embedding of w_2.

Each word is scanned from right-to-left, with a letter removed at each step. If the cosine similarity between two prefixes is above a manually set threshold value[2], then the segmentation point is saved as a potential segmentation and the segmented suffix is saved in a potential morpheme list. This process is repeated until the beginning of word is reached.

The process is given in Algorithm 1. Here, *GetSuffix*$(charNo, word)$ returns a suffix from the end of the *word* with length $charNo$. *GetPrefix* $(charNo, word)$ returns a prefix from the beginning of the *word* with length $charNo$, and *Cosine*$(prefix, word)$ calculates the cosine similarity between the $prefix$ and the $word$.

An example stripping process is given in Figure 1. The cosine similarity values for the same example are given in Table 1. It is clear that cosine similarity is noticeably higher at the real segmentation points than at non-segmentation points. The cosine similarity becomes very small if the prefix and the word are not semantically related. We assign -1 for the cosine similarity if either the prefix or the word does not exist in the word embedding space.

[2] We assign a threshold value based on the experiments performed on a separate development set for each language.

Algorithm 1 Baseline segmentation algorithm that detects potential morpheme boundary points based on cosine similarity between word embeddings.

procedure SEMANTIC_PARSING(word, threshold)
  $suffixList \leftarrow \emptyset$,
  $wordLength \leftarrow LENGTH(word)$
  $charNo \leftarrow 1$
  $counter \leftarrow wordLength$
  while $counter > 1$ **do**
    $suffix \leftarrow GetSuffix(charNo, word)$.
    $prefix \leftarrow GetPrefix(wordLen - charNo, word)$.
    $distance \leftarrow Cosine(prefix, word)$
    if $distance > threshold$ **then**
      $suffixList \leftarrow PUT(suffix)$
      $word \leftarrow prefix$
      $charNo \leftarrow 1$
      $wordLength \leftarrow LENGTH(word)$
    else
      $charNo \leftarrow charNo + 1$
    $counter \leftarrow counter - 1$
  return $suffixList$

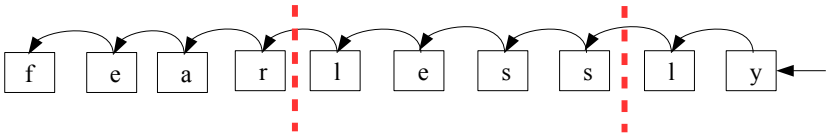

Fig. 1. An illustration that shows the segmentation of the word *fearlessly*. The segmentation is performed from the right end of the word by removing letters using a manually set semantic similarity threshold.

$$p(W|S) = \prod_{i=1}^{T} p(w_i|S) \tag{2}$$

$$= \prod_{i=1}^{T} p(w_i = m_i^0 + \cdots + m_i^N|S) \tag{3}$$

$$= \prod_{i=1}^{T} p(m_i^0|S) \prod_{j=1}^{N} p(m_i^j|m_i^{j-1}, S) \tag{4}$$

In Equation 2, W is the word list of size T and S is the initial segmentation suggested by the inner-word cosine similarities. Therefore, the probability of each word w_i with morphemes $m_i^0, \cdots, m_i^N$ is defined by using the likelihood under the

Table 1. The cosine similarity between the prefixes of the word *fearlessly* is given. 0.25 is assigned for the cosine similarity threshold in this example.

	Word	Prefix	Cosine similarity	Segmentation
1	fearlessly	fearlessl	-1	fearlessly
2	fearlessly	fearless	0.34	fearless-ly
3	fearless	fearles	0.14	fearless-ly
4	fearless	fearle	-1	fearless-ly
5	fearless	fear	0.26	fear-less-ly
6	fear	fea	-1	fear-less-ly
7	fear	fe	-1	**fear-less-ly**

initial segmentation. For each word, we search for the segmentation that leads to the maximum likelihood under the given segmentation:

$$\arg \max_{w_i = m_i^0 + \cdots + m_i^N \in S_{w_i}} P(w_i = m_i^0 + m_i^1 + \cdots + m_i^N | S) \tag{5}$$

where m_i^j denotes the ith morpheme in w_i that consists of N morphemes. Here, S_{w_i} denotes the set of all possible segmentations of w_i.

We replace the root morphemes with a start symbol F to reduce the sparsity. We assume that the root is always the leftmost morpheme, which may not be always the case in English, German or the Semitic languages. However, due to the usage of word embeddings, our model requires all tokens to be available in the dataset on their own with a proper embedding in the embedding space. However, this is not always true for the prefixes[3].Hence, the probability of the first morpheme after the root being *less* is computed as follows:

$$p(less|F) = \frac{n(< F, less >)}{n(F)} \tag{6}$$

where $n(< F, less >)$ is the frequency of *less* being the first morpheme coming just after the root. Both bigram and unigram frequencies are obtained from the initial segmentations. Here, we use the word types, and not tokens. We also discard the frequency information provided by Morpho Challenge data. We do not distinguish root morphemes and bound morphemes in normalization. In other words, $p(m_i^0)$ is normalized by the total frequency of all morphemes, and not just the root morphemes.

Viterbi algorithm is used to find the segmentation with the maximum likelihood according to Equation 5. We apply Laplace smoothing to eliminate the zero probabilities in the model[4].

[3] The limitations of the model is discussed further in Section 3.3.
[4] We applied additive smoothing with a parameter constant 1.

3.2 MAP Model

In the MAP model, we adopt the Bayesian notion of the probability to place some prior belief on the segmentation of the corpus. For that purpose, we use the cosine similarity between the embeddings of the different forms of a word because we know that words that are inflected or derived from each other are semantically related to each other and this semantic relatedness is known *a priori*. Therefore, we use both orthographic features (such as the observed morpheme counts as likelihood) and semantic features (cosine similarity between word embeddings). The posterior probability of the segmentation for a given corpus is defined as follows:

$$
\begin{aligned}
p(S|W) &\propto p(W|S)p(S) && (7)\\
&\propto \prod_{i=1}^{T} p(w_i|S)p(S) && (8)\\
&\propto \prod_{i=1}^{T} \Big(P(w_i = m_i^0 + m_i^1 + \cdots + m_i^N|S) \prod_{k=1}^{N} p(s_i^k) \Big) && (9)\\
&\propto \prod_{i=1}^{T} P(m_i^0|S)P(m_i^1|m_i^0, S)p(s_i^1) && (10)\\
&\quad \cdots P(m_i^N|m_i^0, \cdots, m_i^{N-1}, S)p(s_i^N)
\end{aligned}
$$

where S refers to the segmentation of all the words in the corpus that is learned during inference (whereas S denotes the initial segmentation of the words in the corpus which does not change in a separate inference step). N is the number of morphemes in each word. Here, s_i^k refers to the kth morpheme boundary in w_i that splits m_i^k from the rest of the word, i.e. $m_i^0 + \cdots + m_i^{k-1}$. The probability of splitting a word at any point is proportional to the cosine similarity between the embeddings of the two prefixes. Thus, the probability of splitting the word $w_i = m_i^0 + \cdots + m_i^{k-1} + m_i^k$ after the $k-1$th morpheme is estimated as follows (the word is split such as $m_i^0 + \cdots + m_i^{k-1}$ and m_i^k):

$$
p(s_i^k) = \frac{cos(m_i^0 + \cdots + m_i^{k-1}, m_i^0 + \cdots + m_i^k)}{\sum_{z=1}^{N} cos(m_i^0 + \cdots + m_i^{z-1}, m_i^0 + \cdots + m_i^z)} \tag{11}
$$

Therefore, we convert the cosine similarity into a proper probability distribution by normalizing with the summation of cosine similarities through all possible binary split points. Here, m may not be a valid morpheme. It corresponds to any split point in a word during inference. However, at the end of the inference, it normally corresponds to valid morphemes.

We use the MAP model by adopting two language models: unigram and bigram language model for the likelihood estimation.

3.2.1 Unigram Model

In the unigram model, each morpheme is assumed to be independent from the other morphemes within the same word. The dependency is incorporated through

the prior information which adopts the cosine similarity coming from the word embeddings. The unigram language model for the likelihood is defined as follows:

$$\begin{aligned} p(w_i|S) &= P(w_i = m_i^0 + m_i^1 + \cdots + m_i^N|S) \\ &= \prod_{j=1}^{N} p(m_i^j|S) \end{aligned} \tag{12}$$

3.2.2 Bigram Model

In the bigram model, we adopt a first order Markov chain where each morpheme depends only on the previous morpheme:

$$\begin{aligned} p(w_i|S) &= P(w_i = m_i^0 + m_i^1 + \cdots + m_i^N|S) \\ &= p(m_i^0) \prod_{j=1}^{N} p(m_i^j|m_i^{j-1}, S) \end{aligned} \tag{13}$$

3.3 Limitations

Since both MAP and MLEmodels utilize semantic information, they require word embedding of each word in the dataset. Otherwise, our models cannot suggest any segmentation for those words that do not have embeddings. This is one of the limitations of the models. This limitation causes prefixes not to be learned by the models since they do not exist on their own in the datasets and they do not have embeddings. One possible way to overcome this limitation is to use the n-gram or morpheme embeddings instead of using word embeddings. However, this will require defining the word similarities again based on n-gram based word embeddings since we postulate that inflected forms of a word will remain semantically similar to each other. We did further analysis by using n-gram based word embeddings and they are presented in Section 5.2.4.

Another limitation of our approach is the requirement of embeddings for each prefix of a word in order to compare the prefix with the entire word to assess a possible morpheme boundary. Since prefix affixes do not appear as separate words, their embeddings are not learned with standard word embedding methods such as word2vec (Mikolov et al., 2013). There are methods that learn morpheme embeddings such as morph2vec (Üstün and Can, 2016), which could be employed in our approach. Additionally, it would be possible to assign embeddings to such prefixes as the average of the differences between the words that contain it and the same words without the prefix (e.g. v(impossible)-v(possible) where v denotes the word embedding). The prefix issue exposes a limitation for languages with extensive usage of prefixes such as German and English. However, some languages such as Turkish do not have prefixes. We leave including prefixes in our approach as a future work.

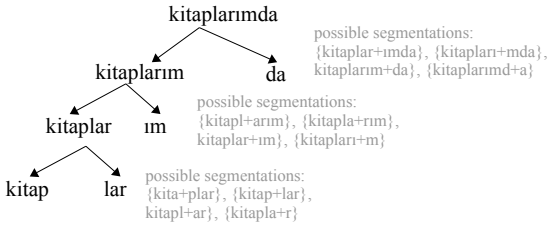

Fig. 2. An illustration that shows how the Turkish word *kitaplarımda* is segmented during recursive Gibbs sampling. In each iteration of the Gibbs sampling, a binary segmentation of the word is drawn from the probability distribution over the binary segmentations of the word. Segmentation continues for each prefix until the word itself is selected with a left-recursion.

4 Inference

In the MLE model, we use Viterbi algorithm to find the maximum likelihood estimates based on the morphemes learned in the first step that makes use of word embeddings (Üstün and Can, 2016). Therefore, we do not need a separate inference step for the MLE model. In the MAP model, we need a separate inference algorithm to learn the morphemes. For this purpose, we use a stochastic inference method, Gibbs sampling (Geman and Geman, 1984). We sample a split point that splits a word in two segments (presumably a stem and a suffix) in each iteration. These samples are drawn from the pre-defined posterior probability distribution:

$$p(S|W) = \frac{p(W|S)p(S)}{p(W)} \tag{14}$$

$$\propto p(W|S)p(S) \tag{15}$$

$$\propto \prod_i p(w_i = m_i^0 + m_i^1|S)p(s_i^1) \tag{16}$$

We can discard the normalization constant $p(W)$ since it is constant for all segmentations.

We sample a word in each iteration during the inference. Once the sampled word is removed from the corpus, the binary segmentations of the word form a probability distribution. One of the binary segmentations is drawn from the following posterior distribution:

$$p(w_i = m_i^0 + m_i^1|S^{-m_{old,i}^0,-m_{old,i}^1})p(s_i^1) \tag{17}$$

where the likelihood is estimated based on either the unigram model or the bigram model. Here, $S^{-m_{old,i}^0,-m_{old,i}^1}$ denotes the set of current morphemes in the model that excludes the morphemes of w_i, that are $m_{old,i}^0$ and $m_{old,i}^1$ in the old segmentation of the word w_i. Here, $m_{old,i}^0$ and $m_{old,i}^1$ correspond to a morpheme boundary point that split the word w_i into $m_{old,i}^0$ and $m_{old,i}^1$ that are the first and the second morphemes in the old segmentation of the word. The second factor $p(s_i^1)$ denotes the probability of splitting the word into m_i^0 and m_i^1 that corresponds to a proposed

Algorithm 2 RGS (Recursive Gibbs Sampling) algorithm that applies left-recursive binary segmentation of a given word.

procedure RGS(w_i, S)
    Remove the current morphemes of w_i, that are $m^0_{old,i}$ and $m^1_{old,i}$
    Draw a binary segmentation from the given posterior distribution: $p(w_i = m^0_i + m^1_i | S^{-m^0_{old,i}, -m^1_{old,i}}) p(s^1_i)$
    if $m^0_i \neq w_i$ **then**
        $S \leftarrow S \cup m^1_i$
        $RGS(m^0_i, S)$
    else
        $S \leftarrow S \cup m^0_i$

segmentation that will be replaced with the old segmentation . The same process is repeated for the left part of the word m^0_i (i.e. presumably the stem). Therefore a left-recursion is applied until it reaches the beginning of the word as long as a new split point is drawn from the posterior probability distribution. This process is repeated for the entire corpus until the model converges to the global maximum point. The inference procedure is given in Algorithm 2. An example is given in Figure 2. The example shows the sampling process for the word *kitaplarımda* in Turkish (means *in my books - kitap:'book', lar:'s', ım*:'my', da:'in'). In each step of the recursive Gibbs sampling algorithm, we choose one of the binary segmentations from the right end of the word. We only consider k binary segmentations that correspond to the last k letters (assuming that the longest suffix has got k letters)[5]. We repeat the process until the word itself is sampled (without splitting the word).

5 Experiments & Results

5.1 Data

We used publicly available Morpho Challenge (Kurimo et al., 2011) datasets for both training and testing. We did experiments on three languages: Turkish, English, and German. We built the training sets by choosing most frequent 10K and 50K words from the full word lists provided by Morpho Challenge. We combined the Morpho Challenge training and development sets for testing purposes. We did not use a separate development set for parameter tuning[6]. Manually collected newspaper archives were used for learning the Turkish word embeddings with a dimen-

[5] We assign $k = 4$ for all languages assuming that the longest suffix has got 4 letters. There are suffixes longer than 4 letters, such as *ation* in English. However, such suffixes are not common in many languages.

[6] The only parameter we have is the semantic similarity threshold in S-MLE model. It should be noted that there is not a manually set threshold value in S-MAP model. As for the similarity parameter in S-MLE model, we did not do any hyperparameter tuning in a separate development set. We assigned a threshold value of 0.25 just to include a semantic effect in the model. The results in Section 5.2.3 show that this parameter is also not optimized for our datasets.

Table 2. Size of the datasets for Turkish, English, and German

	Turkish	English	German
Word Embeddings Train Set	361M	129M	651M
Size of the Train Set	10K & 50K	10K & 50K	10K & 50K
Size of the Test Set	1686	1760	1779

sionality of 200. We used word2vec (Mikolov et al., 2013) and its open source Java implementation Deeplearning4J (Team, 2016) to generate the word embeddings. English word embeddings are pre-trained on Google News dataset and German word embeddings are pre-trained on Wikipedia dataset. Both word embeddings have a dimensionality of 300. . The size of all datasets are given in Table 2.

We follow the same evaluation methodology provided by Morpho Challenge. A number of word pairs are randomly sampled from the result analyses that have at least one morpheme in common. For each sample word pair that really has a morpheme in common according to the gold standard segmentation, one point is given. Total number of points is divided by the total number of sampled word pairs. This gives the precision. For instance, given that the proposed segmentation of the English word *access* is *acces*+*s*, two word pairs are selected randomly from the result set. Let's assume that *access* shares the morpheme *acces* with the word *accesses*. In this case, we obtain the word pair *access* - *accesses*. Let's also assume that *access* shares the morpheme +*s* with the word *books*. This gives another word pair, which is *access* − *books*. Based on the gold standard segmentations, `access_N`, `access_N` +`PLU`, `book_N` +`PLU`, the pair *access* − *accesses* is correct (shared morpheme is `access_N`) and the pair *access* − *books* is incorrect (no common morpheme). Therefore, the precision for the word *access* becomes 1/2 = 50%. The same is applied for recall by sampling word pairs randomly that have one morpheme in common according to the gold standard segmentations. It is checked from the result analyses whether they really have a morpheme in common. If the word pair has really a morpheme in common, one point is given and the total number of points is divided by the total number of sampled word pairs. Finally F-measure is computed as the harmonic mean of the precision and recall:

$$F - measure = \frac{1}{\frac{1}{precision} + \frac{1}{recall}} \tag{18}$$

We assigned $d = 0.25$ for the cosine similarity threshold to extract the potential morphemes in the initial step in the MLE model. We performed several experiments on the development sets (Kurimo et al., 2011) in order to assign the threshold value. Such a threshold value is not needed in the MAP model.

5.2 Evaluation and Results

We did experiments with four models we proposed in this article. The models are MLE unigram model (S-MLE unigram), MLE bigram model (S-MLE bigram), unigram MAP model (S-Unigram MAP), and bigram MAP model (S-Bigram MAP). Here S stands for semantic. We evaluated the initial segmentations obtained from the first step of the MLE model as a baseline model (S-Baseline). Additionally, we did two more iterations by updating the parameters with the new MLE estimates obtained from the Viterbi algorithm in the MLE model. The results are given as S-MLE unigram and S-MLE bigram with the number of iterations.

We compare our model with Morfessor Baseline (Creutz and Lagus, 2005b) (M-Baseline), Morfessor CatMAP (Creutz and Lagus, 2005a) (M-CatMAP) and MorphoChain system (Narasimhan et al., 2015). Additionally, we compare our models with a baseline model where the most frequent morphemes[7] are split off from each word using the longest common subsequence from the end of the word until the word cannot be split anymore. We trained Morfessor Baseline and the Morfessor CatMAP by assigning the perplexity threshold and the iteration number as 10 (which are the default parameters). As for the MorphoChain, we assigned frequency threshold=0, vector size=200, minimum segmentation lenght=3 and the top affix selection number=100, which are also the default parameters of the model. For that purpose, we trained these models on the same training sets and tested on the same test sets for the evaluation. The results obtained from 10K of training set size are given in Table 3, Table 4, and 5. The results obtained from 50K of training set size are given in Table 6, 7, and 8.

The results obtained from 10K training set show that our MLE bigram model outperforms the other models on Turkish. Our MAP models also outperform Morfessor Baseline, Morfessor CatMAP, and MorphoChain, however the MAP model does not perform as well as our proposed MLE models. This could be due to oversegmentation because of high cosine similarities obtained from the word embeddings. For English, again MLE bigram model outperforms our other proposed models. However, the best scores are obtained from Morfessor Baseline. Morfessor Baseline also outperforms the other models on German. Our MLE bigram model comes the second with further one iteration for the parameter estimates in German. However, our MAP models do not perform very well on German analogously to other two languages. This may be due to the heavy usage of compounds in German language. Since we postulate a semantic similarity between the derivations of the words, this may not apply to compounds. For example, *hundemüde* (meaning *exhausted* in English) is composed of two words: *hund* (*dog*) and *müde* (*tired*), which are not semantically similar. Moreover, connector sounds might be inserted while combining different words together in a compound in German language. For example, an extra 'e' letter is inserted in the compound *(das) schwein-e-fleisch* (meaning *pork meat*). This hinders from obtaining the word embeddings of prefixes in a word in

[7] We used the most frequent 100 morphemes.

Table 3. Results obtained from the 10K datasets for Turkish.

	Precision (%)	Recall (%)	F-measure (%)
S-MLE unigram (1 iteration)	70.20	25.06	36.93
S-MLE unigram (2 iterations)	63.33	27.51	38.36
S-MLE bigram (1 iteration)	57.52	42.27	**48.73**
S-MLE bigram (2 iterations)	54.72	42.77	48.01
S-Unigram MAP	30.25	**74.53**	43.03
S-Bigram MAP	33.69	72.82	46.07
Baseline (most freq.)	35.95	48.94	41.45
S-Baseline (Üstün and Can, 2016)	64.41	30.38	41.28
S-MLE bigram (Üstün and Can, 2016)	55.56	40.12	46.59
M-Baseline (Creutz and Lagus, 2002)	68.81	29.40	41.20
M-CatMAP (Creutz and Lagus, 2005a)	**79.69**	27.01	40.34
MorphoChain (Narasimhan et al., 2015)	65.32	20.79	30.18

Table 4. Results obtained from the 10K datasets for English.

	Precision (%)	Recall (%)	F-measure (%)
S-MLE unigram (1 iteration)	3.71	46.09	49.61
S-MLE unigram (2 iterations)	47.57	52.05	49.71
S-MLE bigram (1 iteration)	53.96	48.88	51.30
S-MLE bigram (2 iterations)	44.48	53.56	48.60
S-Unigram MAP	19.63	**82.19**	31.69
S-Bigram MAP	24.83	80.82	37.99
Baseline (most freq.)	27.84	64.98	38.98
S-Baseline (Üstün and Can, 2016)	71.21	37.75	49.37
S-MLE bigram (Üstün and Can, 2016)	17.99	41.60	26.74
M-Baseline (Creutz and Lagus, 2002)	65.06	61.68	**63.33**
M-CatMAP (Creutz and Lagus, 2005a)	53.02	71.02	60.71
MorphoChain (Narasimhan et al., 2015)	**77.98**	17.20	28.18

Table 5. Results obtained from the 10K datasets for German.

	Precision (%)	Recall (%)	F-measure (%)
S-MLE unigram (1 iteration)	28.17	45.95	34.93
S-MLE unigram (2 iterations)	27.47	47.82	34.90
S-MLE bigram (1 iteration)	25.14	50.29	33.52
S-MLE bigram (2 iterations)	26.09	54.50	35.28
S-Unigram MAP	12.97	83.00	22.44
S-Bigram MAP	14.63	**81.54**	24.81
Baseline (most freq.)	12.97	76.35	22.17
S-Baseline (Üstün and Can, 2016)	28.68	42.80	34.35
S-MLE bigram (Üstün and Can, 2016)	17.99	41.60	25.12
M-Baseline (Creutz and Lagus, 2002)	53.29	42.87	**47.51**
M-CatMAP (Creutz and Lagus, 2005a)	19.98	43.12	27.31
MorphoChain (Narasimhan et al., 2015)	**58.08**	14.38	23.05

the initial segmentation. Our model does not perform well for compounds because of these two reasons.

If we compare the baseline models in Turkish, we can see that the baseline model that splits off the most frequent morphemes from the end of each word performs

Table 6. Results obtained from the 50K datasets for Turkish.

	Precision (%)	Recall (%)	F-measure (%)
S-MLE unigram (1 iteration)	69.85	25.06	36.88
S-MLE unigram (2 iterations)	70.27	25.10	36.98
S-MLE bigram (1 iteration)	60.54	42.27	**49.78**
S-MLE bigram (2 iterations)	56.35	44.35	49.75
S-Unigram MAP	34.43	**59.43**	43.61
S-Bigram MAP	39.70	47.37	43.20
Baseline (most freq.)	35.11	49.59	41.11
S-Baseline (Üstün and Can, 2016)	64.41	30.38	41.28
S-MLE bigram (Üstün and Can, 2016)	60.14	40.44	48.36
M-Baseline (Creutz and Lagus, 2002)	**78.67**	24.68	37.75
M-CatMAP (Creutz and Lagus, 2005a)	71.65	29.67	41.96
MorphoChain (Narasimhan et al., 2015)	71.24	21.26	32.75

Table 7. Results obtained from the 50K datasets for English.

	Precision (%)	Recall (%)	F-measure (%)
S-MLE unigram (1 iteration)	57.83	46.09	51.29
S-MLE unigram (2 iterations)	50.74	49.61	50.17
S-MLE bigram (1 iteration)	49.35	48.88	49.11
S-MLE bigram (2 iterations)	45.63	51.30	48.30
S-Unigram MAP	29.99	69.58	41.92
S-Bigram MAP	29.52	66.91	40.96
Baseline (most.freq)	21.34	**74.55**	33.18
S-Baseline (Üstün and Can, 2016)	**71.21**	37.75	49.37
S-MLE bigram (Üstün and Can, 2016)	50.91	30.84	39.50
M-Baseline (Creutz and Lagus, 2002)	64.86	65.34	**65.10**
M-CatMAP (Creutz and Lagus, 2005a)	29.05	26.36	27.64
MorphoChain (Narasimhan et al., 2015)	80.62	24.51	37.59

at par with the semantic baseline model (S-Baseline) (see Table 3 and Table 6). However, the semantic baseline model performs slightly better than the baseline model using only the most frequent morpheme list on the 50K dataset. In English and in German, this difference is much more significant compared to Turkish. The results show that using semantic information through the word embeddings has a substantial impact on segmentation regardless of the morphological structure of the language. However, the final result is still far behind the other methods such as Morfessor (Creutz and Lagus, 2002) because of the reasons mentioned above for German. In English, there is an extensive usage of irregular words and since we assume a concatenative morphology here, still the baseline model and other models fall behind the Morfessor (Creutz and Lagus, 2002). In Turkish, the semantic baseline model still performs very well compared to the other models, however the baseline model using the most frequent morphemes performs as well as the semantic baseline model due to the concatenative nature of the baseline algorithm that allows to find many of the valid morphemes.

The results obtained from 10K and 50K training sets show how the size of the training set affects the accuracy of the models. Our MLE bigram model outperforms other models with a F-measure of 49.78% on Turkish with two further iterations in

Table 8. Results obtained from the 50K datasets for German.

	Precision (%)	Recall (%)	F-measure (%)
S-MLE unigram (1 iteration)	26.63	45.95	33.72
S-MLE unigram (2 iterations)	26.90	46.33	34.04
S-MLE bigram (1 iteration)	22.15	50.29	30.75
S-MLE bigram (2 iterations)	22.26	53.37	31.41
S-Unigram MAP	20.88	**72.01**	32.37
S-Bigram MAP	24.10	65.79	35.28
Baseline (most freq.)	10.74	74.69	18.78
S-Baseline (Üstün and Can, 2016)	28.68	42.80	34.35
S-MLE bigram (Üstün and Can, 2016)	19.17	43.73	26.66
M-Baseline (Creutz and Lagus, 2002)	**59.72**	39.53	**47.58**
M-CatMAP (Creutz and Lagus, 2005a)	28.33	31.77	29.96
MorphoChain (Narasimhan et al., 2015)	58.66	15.37	24.36

parameter estimation. Our unigram and bigram MAP models also perform better than other models such as Morfessor Baseline and Morfessor CatMAP on Turkish. Morfessor Baseline outperforms other models on English with a F-measure of 65.10%. However, our S-MLE unigram model comes the second and outperforms our other models. Bigram models improve the results significantly for morphologically complex languages, whereas it is not the case in morphologically poor languages such as English. Morfessor Baseline outperforms all models again with a F-measure of 47.58% on German. Differently from the experiments on smaller training sets, our bigram MAP model comes the second and both the semantic baseline model and the MLE models outperform Morfessor CatMAP and MorphoChain for German. This shows that our model performs better when it is trained larger training sets for three of the languages.

S-MLE method works well for concatenative morphology since it begins from the end of the word and gradually strips off the suffixes by using the semantic information. English does not have a heavy morphology compared to Turkish so it cannot benefit from the semantic segmentation as much as Turkish. The same also applies for German but for a different reason. Since German relies heavily on compounds, it also cannot benefit from iterative segmentation from the end of the word as suggested in the baseline model.

To sum up, it shows that our model outperforms other models on Turkish in both training settings. This shows that using only semantic information gives promising results. Our MLE model performs the best among our models on English although the highest obtained scores fall behind that of Morfessor Baseline, which outperforms all models for English. However, our MLE scores are the highest among all models after Morfessor Baseline. For German, Morfessor Baseline is again the best, but this time our bigram MAP-based model performs better than other models.

5.2.1 Analysis of the results

Regarding the oversegmentation vs undersegmentation in different models, S-MLE model seems to be more robust to oversegmentation compared to the S-MAP model

in Turkish and English. The precision and recall scores of S-MLE model are balanced, although the precision scores are slightly higher than recall scores for both English and Turkish. Morfessor CatMAP (Creutz and Lagus, 2005a) and MorphoChain (Narasimhan et al., 2015) suffer from undersegmentation in all of the three languages and therefore the recall scores obtained from Morfessor CatMAP and MorphoChain are very low and precision scores are comparably higher although the F-measure is fair average among all scores. In contrast, S-MLE model shows a different picture for German, where the recall scores are much higher than precision scores and thereby the model suffers from oversegmentation in German. This is actually a similar outcome for German language for all models except Morfessor Baseline (Creutz and Lagus, 2002) and MorphoChain (Narasimhan et al., 2015) which give a higher precision unlike other models. This is again due to the heavy usage of compounds in German and therefore due to very long words that are prone to be oversegmented. As for S-MAP model, the precision and recall are less balanced compare to S-MLE model. There is a slight oversegmentation issue in both English and German, where the recall scores are substantially higher than precision scores. However, although the recall scores in Turkish are higher than precision scores for the S-MAP model, they are more balanced compared to English and German, therefore oversegmentation is not an issue in Turkish for the S-MAP model.

Some correct and incorrect examples in English, Turkish, and German results are given in Table 9, Table 10, and Table 11 respectively. MAP model performs better than MLE model, where MLE model is prone to undersegmentation in German. However, MLE model performs better in English and Turkish compared to MAP model where the MAP model is prone to oversegmentation.

If we compare three languages from the morphological complexity point of view, we can easily claim that Turkish is the most morphological complex language out of these languages. This morphological complexity is due to the agglutination where each word is composed of many morphemes and the semantics is still preserved while deriving new word forms from a root. In English, each word usually takes 2.2 morphemes in average at the end, however this number in Turkish is 3.3 and it is 3.1 in German in average (as we analyzed from the test sets). In German, this number includes also the number of the additional roots in each word. Therefore, we can argue that Turkish is comparably morphologically richer than both English and German. Table 9 shows that our models can still learn the correct segmentations in English if the roots did not change after the inflection or derivation, such as *switch+ing*, *help+less+ness* etc. However, we cannot expect from our models to work for irregular word forms since we aim to extract the morpheme boundary points in each word. English being a language with a heavy usage of irregular word forms is also not in the focus of our proposed models.

In German, the structure is completely different compared to English and Turkish. Words are usually composed of several different roots. Since we are aiming to take advantage of using the semantic similarity between derived word forms, this is not an advantage for German where semantically unrelated words can come together to produce a completely semantically unrelated word. The low German results also validate this situation.

Table 9. Some segmentation examples for the different models on English 10K results. The bold segmentation refers to the correct segmentation.

S-MLE Unigram	S-MLE Bigram	S-Unigram MAP	S-Bigram MAP
incline-d	**incline-d**	inc-line-d	inc-lin-e-d
critic-ally	**critical-ly**	critic-ally	**critical-ly**
switch-ing	**switch-ing**	**switch-ing**	**switch-ing**
alt-ernative-s	alt-ernative-s	alt-er-native-s	alt-er-native-s
pray-ers	**pray-er-s**	pray-e-r-s	**pray-er-s**
help-less-ness	**help-less-ness**	**help-less-ness**	**help-less-ness**

Table 10. Some segmentation examples for the different models on Turkish 10K results. The bold segmentation refers to the correct segmentation.

S-MLE Unigram	S-MLE Bigram	S-Unigram MAP	S-Bigram MAP
hük-ü-met-ler-e	hük-üm-et-le-r-e	hük-ü-m-e-tl-e-re	hüküm-et-ler-e
tan-ım-lar-a	**tanım-lar-a**	**tanım-lar-a**	**tanım-lar-a**
takım-ı-nın	takım-ı-n-ı-n	takım-ın-ın	tak-ı-mın-ın
geti-r-dim	getir-dim	**getir-di-m**	getir-dim
insan-lık	**insan-lık**	**insan-lık**	**insan-lık**
kum-aşa	kum-a-ş-a	kum-aşa	kum-aş-a

We performed also boundary evaluation for the highest scores obtained for English and Turkish as applied in MorphoChain (Narasimhan et al., 2015). The results are given in Table 12. All models are trained on 50K datasets and tested on Morpho Challenge gold segmentations. The boundary evaluation requires the surface forms to be evaluated. Since we could not find such a gold standard segmentation data for German that has only the surface forms, we did not include the German scores for this evaluation. MorphoChain evaluation gives higher scores compared to Morpho Challenge evaluation. There are several reasons for this (as explained in (Can and Manandhar, 2018)). First, morpheme labels are used in Morpho Challenge evaluation, which increases the total number of points that is computed over all word pairs, and it lowers the scores. Second, only the gold segmentation that has the maximum match with the result segmentation is considered in MorphoChain evaluation, and not all the given segmentations as in Morpho Challenge. Therefore, generally all scores are higher in MorphoChain evaluation.

In order to compute the upper bound of the methods, we investigated the performance based on the existence of the words in the word embedding space. To this end, we performed segmentation if any prefix of the word has a word embedding without using any semantic similarity between the prefixes of the word, which will

Table 11. Some segmentation examples for the different models on German 10K results. The bold segmentation refers to the correct segmentation.

S-MLE Unigram	S-MLE Bigram	S-Unigram MAP	S-Bigram MAP
wesentlic-h-er	wes-ent-lic-h-e-r	**wesentlich-er**	wes-e-nt-li-ch-er
bus-se	**bus-se**	**bus-se**	**bus-se**
amtlic-h	am-t-lich	**amt-lich**	**amt-lich**
einlaufe-n	einlaufe-n	ein-la-ufen	**ein-lauf-en**
erbos-t-e	erbos-t-e	**erbos-te**	erb-oste
bei-n-e-n	**bein-en**	bei-n-en	**bein-en**

give an upper bound for the methods. The upper bound for the F-measure in Turkish is 40.53% if no semantic information is incorporated but only the existence of the prefixes of a word is exploited during segmentation, which is lower than 41.28% when the semantic similarity threshold is incorporated in the model. This shows that the incorporation of the semantic similarity improves the upper bound. We also investigated the MLE method's upper bound analogously. We obtained a F-measure of 39.69% from the bigram MLE model using 50K training set in Turkish, whereas we obtained 48.36% from the bigram MLE model using the semantic similarity in the initial segmentations. Results are coherent with the baseline model and again it shows that the incorporation of the semantic similarity improves the scores significantly.

We analyzed how many segments in the gold standard are actually represented in their word embeddings to see if our method is applicable to other languages. To this end, we created a word list from the gold standard set by combining the prefixes of each given word. For example, we obtained *book*, *booking* and *bookings* from the gold segmentation *book+ing+s*. As a result, in Turkish, 86% of words in the gold standard are represented in the word embedding space. This coverage seems to be reasonable to apply our method. In English, this coverage is at around 72% which is lower than Turkish and this coverage also might be hindering the performance of our method in English. Since the gold segmentations of the German words is not provided by Morpho Challenge, we could not compute the coverage of the words in the German word embedding space.

5.2.2 The effect of the word embedding dimensionality

In order to evaluate the effect of the word embedding dimensionality size on the performance of our models, we trained word2vec (Mikolov et al., 2013) for various dimensions on Turkish. We again used the manually collected newspaper archives. We performed experiments on the unigram and bigram MAP models for dimensions ranging from 50 to 300. Figure 3 shows the F-measure scores for different vector dimensions. The scores show that the optimal results are obtained from 100

Table 12. MorphoChain (Narasimhan et al., 2015) evaluation for Turkish and English.

	TURKISH		
M-CatMAP (Creutz and Lagus, 2005a)	74.12	67.13	**70.46**
MorphoChain (Narasimhan et al., 2015)	70.50	67.75	69.09
S-MLE bigram (1 iteration)	59.49	71.35	64.88
M-Baseline (Creutz and Lagus, 2002)	71.79	56.67	63.33
	ENGLISH		
MorphoChain (Narasimhan et al., 2015)	71.52	79.94	**75.50**
M-Baseline (Creutz and Lagus, 2002)	59.30	78.82	67.68
S-MLE unigram (1 iteration)	45.18	80.35	57.83
M-CatMAP (Creutz and Lagus, 2005a)	28.07	80.28	41.59

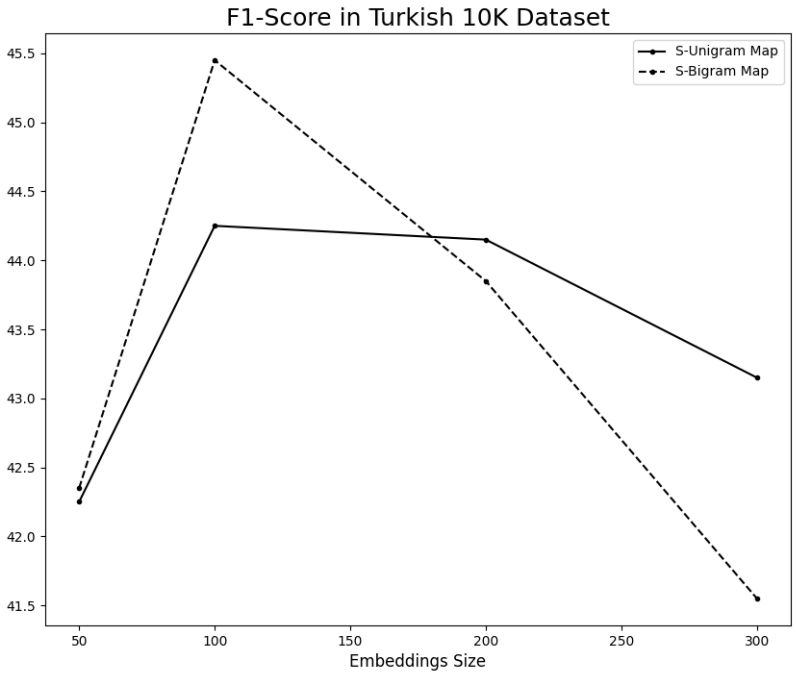

Fig. 3. Results obtained from S-Unigram MAP and S-Bigram MAP models on Turkish 10K training dataset for different word embedding sizes.

dimension. Smaller dimensions give poor results and larger dimensions also begin to give poorer results when the dimension size is increased gradually.

Table 13. F-measures obtained from S-Baseline and S-MLE models on Turkish for different cosine similarity threshold values.

Threshold (d)	S-Baseline (%)	S-MLE (%)
0.15	40.51	47.51
0.25	37.42	47.82
0.35	30.16	43.58
0.45	25.14	39.95

Table 14. The F-scores on Turkish that show the effect of semantic similarity by changing the effect of the cosine terms in Equation 11 with various exponent values: 0.25, 0.5, 0.75, 1

Threshold (d)	0.25	0.5	0.75	1	1.25
S-Unigram MAP (%)	43.89	41.81	44.01	43.03	42.24
S-Bigram MAP (%)	44.66	44.84	47.16	46.07	43.06

5.2.3 The effect of the semantic similarity

Additionally, we performed another set of experiments to discover the effect of the cosine similarity threshold used in segmentation. Table 13 gives the F-measure scores obtained from S-Baseline and S-MLE models. The results show that a cosine threshold of 0.25 gives the highest score for the MLE model, whereas 0.15 gives the highest score for the baseline model.

It should be noted that there is not any manually set threshold value in the MAP models. In order to measure the effect of the semantic features, we added a non-negative exponent to all the cosine terms in Equation 11. The results for different values of the exponent are given in Table 14 for Turkish. The results show that more effect of the semantic similarity is incorporated, the higher the scores are in general. However, the scores are the highest when the exponent is reduced to 0.75.

5.2.4 The effect of using n-gram level word embeddings

In order to investigate the effect of the word embeddings, we also used fastText (Bojanowski et al., 2017) on S-Baseline model using a semantic similarity threshold of 0.25. FastText (Bojanowski et al., 2017) is another method to learn word embeddings using the n-grams of each word rather than taking each word as a distinct unit during learning the embedding space as in word2vec (Mikolov et al., 2013). The results obtained for Turkish are given in Table 15. The results obtained from fastText are significantly higher than that of the ones obtained from S-Baseline that uses word2vec (Mikolov et al., 2013) embeddings in Turkish. We also analyzed

Table 15. The F-scores on Turkish that show the effect of using n-gram level word embeddings, i.e. fastText Bojanowski et al. (2017)

Model	F-measure (%)
upper bound - fastText	40.69
upper bound - word2vec	40.53
S-Baseline - fastText	**49.72**
S-Baseline - word2vec	41.28

Table 16. The computation times and memory requirements for the models (S-MLE Unigram, S-MLE Bigram, S-Unigram MAP, S-Bigram MAP) trained on Turkish 10K dataset.

	Computation Time (min.)	Memory Requirement (Gb)
S-MLE Unigram	2.15 for 1 iteration	10
S-MLE Bigram	2.28 for 1 iteration	10
S-Unigram MAP	33.54 for 5000 iterations	10
S-Bigram MAP	38.41 for 5000 iterations	10

the upper bounds of using both word2vec and fastText when the prefixes are segmented if their embeddings exist in the embedding space without incorporating any semantic information. The results obtained from fastText is still higher than that of word2vec. This shows that using n-gram level word embeddings improves the scores significantly, since it also learns the word embeddings out of their n-grams and therefore it already incorporates the morpheme level information within the embeddings. Moreover, using fastText embeddings shows the semantic similarity effect better than the word2vec embeddings.

5.2.5 Computational times and memory requirements

The computation times and memory requirements for all the methods are given in Table 16[8]. We performed all the experiments on a server with 24 Intel Xeon processors of 2.40GHz.

[8] S-MLE models are trained for one iteration for which the computation time is given, whereas the S-MAP models are trained for 5000 iterations and the total required time is given.

6 Conclusion and Future Work

We propose two unsupervised frameworks for morphological segmentation by incorporating semantic information in MLE and MAP models. Our results show that using semantics helps in unsupervised morphological segmentation. A simple baseline model based on only semantic information can give promising results on a language with concatenative morphology. We also show that how semantic information obtained from word embeddings can be utilized in a simple MLE or MAP model.

Our models do not require a large dataset for training. The results show that our results do not change much if we increase the training set from 10K to 50K. Therefore, our models can be trained by using only 10K raw word list, which is not annotated. We use pre-trained word embeddings but embedding models do not require an annotated dataset but just a raw corpus. Therefore, our model can be applied to any low resource language and does not require annotation in advance, because it is fully unsupervised.

In this work, we learn the word embeddings from word2vec (Mikolov et al., 2013). We aim to learn the embeddings concurrently with the segmentation of words in the future without using any pre-trained word embeddings.

Acknowledgements

This research was supported by TUBITAK (The Scientific and Technological Research Council of Turkey) with grant number 115E464.

References

Bojanowski, P., **Grave, E.**, **Joulin, A.**, **and Mikolov, T.** 2017. Enriching word vectors with subword information. In *Transactions of the Association of Computational Linguistics*, TACL, pp. 135–146.

Can, B. and Manandhar, S. 2010. Clustering morphological paradigms using syntactic categories. In *Proceedings of the Multilingual Information Access Evaluation I. Text Retrieval Experiments: 10th Workshop of the Cross-Language Evaluation Forum, Revised Selected Papers*, pp. 641–648. Springer Berlin Heidelberg.

Can, B. and Manandhar, S. 2018. Tree structured dirichlet processes for hierarchical morphological segmentation. *Computational Linguistics*, 44(2):349–374.

Cao, K. and Rei, M. 2016. A joint model for word embedding and word morphology. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pp. 18–26.

Clark, A. 2000. Inducing syntactic categories by context distribution clustering. In *Proceedings of the 2nd Workshop on Learning Language in Logic and the 4th Conference on Computational Natural Language Learning - Volume 7*, ConLL '00, pp. 91–94. Association for Computational Linguistics.

Cotterell, R. and Schütze, H. 2015. Morphological word-embeddings. In *Proceedings of the Human Language Technologies: The 2015 Annual Conference of*

the North American Chapter of the ACL, pp. 1287–1292, Denver, Colorado. Association for Computational Linguistics.

Creutz, M. and Lagus, K. 2002. Unsupervised discovery of morphemes. In *Proceedings of the ACL-02 workshop on Morphological and phonological learning - Volume 6*, MPL '02, pp. 21–30. Association for Computational Linguistics.

Creutz, M. and Lagus, K. 2005a. 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 2005)*, pp. 106–113.

Creutz, M. and Lagus, K. 2005b. Unsupervised morpheme segmentation and morphology induction from text corpora using morfessor 1.0. *Technical Report A81*.

Creutz, M. and Lagus, K. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions Speech Language Processing*, 4:3:1–3:34.

de Marcken, C. 1996. Linguistic structure as composition and perturbation. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pp. 335–341, Santa Cruz, California, USA. Association for Computational Linguistics.

Geman, S. and Geman, D. 1984. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741.

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

Goldwater, S., **Johnson, M.**, **and Griffiths, T. L.** 2006. Interpolating between types and tokens by estimating power-law generators. In *Proceedings of the Advances in Neural Information Processing Systems 18*, pp. 459–466. MIT Press.

Hankamer, J. 1986. Finite state morphology and left to right phonology. In *Proceedings of the West Coast Conference on Formal Linguistics (WCCFL-5)*.

Harris, Z. S. 1955. From phoneme to morpheme. *Language*, 31(2):190–222.

Harris, Z. S. 1970a. *Papers in Structural and Transformational Linguistics*, pp. 68–77.

Harris, Z. S. 1970b. Morpheme boundaries within words: report on a computer test. *Papers in Structural and Transformational Linguistics.* Reprinted in Harris (1970a), pages 26–46.

Kurimo, M., **Lagus, K.**, **Virpioja, S.**, **and Turunen, V. T.** 2011. Morpho Challenge 2010. `http://research.ics.tkk.fi/events/morphochallenge2010/`. Online; accessed 10-February-2017.

Lazaridou, A., **Marelli, M.**, **Zamparelli, R.**, **and Baroni, M.** 2013. Compositional-ly derived representations of morphologically complex words in distributional semantics. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1517–1526, Sofia, Bulgaria. Association for Computational Linguistics.

Lee, Y. K., **Haghighi, A.**, **and Barzilay, R.** 2011. Modeling syntactic context improves morphological segmentation. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, CoNLL '11, pp. 1–9, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mikolov, T., **Chen, K.**, **Corrado, G.**, **and Dean, J.** 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Narasimhan, K., **Barzilay, R.**, **and Jaakkola, T. S.** 2015. An unsupervised method for uncovering morphological chains. *Transactions of the Association for Computational Linguistics (TACL)*, 3:157–167.

Schone, P. and Jurafsky, D. 2001. Knowledge-free induction of inflectional morphologies. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics on Language Technologies*, NAACL '01, pp. 1–9. Association for Computational Linguistics.

Soricut, R. and Och, F. 2015. Unsupervised morphology induction using word embeddings. In *Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL*, pp. 1627–1637.

Team, D. D. 2016. Deeplearning4j: Open-source distributed deep learning for the JVM, Apache Software Foundation License 2.0. `http://deeplearning4j.org/`. Online; accessed 10-February-2017.

Üstün, A. and Can, B. 2016. Unsupervised morphological segmentation using neural word embeddings. In *Proceedings of the Statistical Language and Speech Processing: 4th International Conference, SLSP 2016, Pilsen, Czech Republic, October 11-12, 2016*, pp. 43–53. Springer International Publishing.