Unsupervised Concept Discovery In Hebrew Using Simple Unsupervised Word Prefix Segmentation for Hebrew and Arabic

Elad Dinur[1] **Dmitry Davidov**[2] **Ari Rappoport**[1]

[1]Institute of Computer Science
[2]ICNC
The Hebrew University of Jerusalem

Abstract

Fully unsupervised pattern-based methods for discovery of word categories have been proven to be useful in several languages. The majority of these methods rely on the existence of function words as separate text units. However, in morphology-rich languages, in particular Semitic languages such as Hebrew and Arabic, the equivalents of such function words are usually written as morphemes attached as prefixes to other words. As a result, they are missed by word-based pattern discovery methods, causing many useful patterns to be undetected and a drastic deterioration in performance. To enable high quality lexical category acquisition, we propose a simple unsupervised word segmentation algorithm that separates these morphemes. We study the performance of the algorithm for Hebrew and Arabic, and show that it indeed improves a state-of-art unsupervised concept acquisition algorithm in Hebrew.

1 Introduction

In many NLP tasks, we wish to extract information or perform processing on text using minimal knowledge on the input natural language. Towards this goal, we sometimes find it useful to divide the set of words in natural language to function words and content words, a division that applies in the vast majority of languages. Function words (or grammatical words, e.g., a, an, the, in, of, etc) are words that have little or highly ambiguous lexical meaning, and serve to express grammatical or semantic relationships with the other words in a sentence.

In some morphologically-rich languages, important function words are not written as space-separated units but as morphemes attached as prefixes to other words. This fact can cause problems when statistically analyzing text in these languages, for two main reasons: (1) the vocabulary of the language grows, as our lexical knowledge comes solely from a corpus (words appear with and without the function morphemes); (2) information derived from the presence of these morphemes in the sentence is usually lost.

In this paper we address the important task of a fully unsupervised acquisition of Hebrew lexical categories (or concepts – words sharing a significant aspect of their meaning). We are not aware of any previous work on this task for Hebrew. Due to the problem above, the performance of many acquisition algorithms deteriorates unacceptably. This happens, for example, in the (Davidov and Rappoport, 2006) algorithm that utilizes automatically detected function words as the main building block for pattern construction.

In order to overcome this problem, one should separate such prefixes from the compound words (words consisting of function morphemes attached to content words) in the input corpus. When we consider some particular word, there are frequently many options to split it to smaller strings. Fortunately, the set of function words is small and closed, and the set of grammatical sequences of function prefixes is also small. Hence we assume it does not cost us much to know in advance what are the possible sequences for a specific language.

Even when considering the small number of possible function words, the task of separating them is not simple, as some words may be ambiguous. When reading a word that starts with a prefix known to be a function morpheme, the word may

Proceedings of the EACL 2009 Workshop on Computational Approaches to Semitic Languages, pages 36–44,
Athens, Greece, 31 March, 2009.

be a compound word, or it may be a meaningful word by itself. For example, the word "hsws" in Hebrew[1] can be interpreted as "hsws" (hesitation), or "h sws" (the horse). The segmentation of the word is context dependent – the same string may be segmented differently in different contexts.

One way of doing such word prefix segmentation is to perform a complete morphological disambiguation of the sentence. The disambiguation algorithm finds for each word its morphological attributes (POS tag, gender, etc.), and decides whether a word is a compound word or a word without prefixes. A disambiguation algorithm generally relies on a language-specific morphological analyzer. It may also require a large manually tagged corpus, construction of which for some particular language or domain requires substantial human labor. We avoid the utilization of such costly and language-specific disambiguation algorithms and manually annotated data.

In this paper we present a novel method to separate function word prefixes, and evaluate it using manually labeled gold standards in Hebrew and Arabic. We incorporate the method into a pattern-based Hebrew concept acquisition framework and show that it greatly improves state-of-art results for unsupervised lexical category acquisition. This improvement allows the pattern-based unsupervised framework to use one-tenth of the Hebrew data in order to reach a similar level of results.

Section 2 discusses related work, and Section 3 reviews the word categories discovery algorithm. Section 4 presents the word prefix segmentation algorithm. Results are given in Section 5.

2 Related Work

In this paper we develop an unsupervised framework for segmentation of the function words for languages where context is important for correct segmentation. Our main target language is Hebrew, and we experimented with Arabic as well. As far as we know, there is no work on unsupervised segmentation of words in Hebrew which does not utilize language-specific tools such as morphological analyzers.

Lee et al. (2003) addressed supervised word segmentation in Arabic and have some aspects similar to our approach. As in their study, we also have a pre-supplied list of possible prefix sequences and assume a trigram model in order to find the most probable morpheme sequence. Both studies evaluate performance on a segmented text, and not just on words in the lexicon. However, their algorithm, while achieving good performance (97% accuracy), relies on a training set – a manually segmented corpus of about 110,000 words, while our unsupervised framework does not require any annotation and is thus easier to implement and to apply to different domains and languages.

Snyder and Barzilay (2008) study the task of unsupervised morphological segmentation of multiple languages. Their algorithm automatically induces a segmentation and morpheme alignment of short parallel phrases from a multilingual corpus. Their corpus (The Hebrew Bible and translations) contains parallel phrases in English, Arabic, Hebrew and Aramaic. They obtain 63.87 F-Score for Hebrew words segmentation (prefix and suffix), where recall and precision is calculated based on all possible segmentation points.

Another type of segmentation algorithms involves utilization of language-specific morphological analyzers for complete morphological disambiguation. In Hebrew each word usually has more than one possible POS (along with other attributes, such as gender, number, etc.). Assuming we have a morphological analyzer (producing the set of possible analyses for a given word), we can try to discover the correct segmentation of each word.

Levinger et al. (1995) developed a method for disambiguation of the results provided by a morphological analyzer for Hebrew. Adler and Elhadad (2006) proposed an unsupervised algorithm for word segmentation. They estimate an initial language model (using (Levinger et al., 1995)) and improve this model with EM. Direct comparison to their work is problematic, however, since we avoid utilization of a language-specific morphology/POS analyzer. There are also studies of this type that utilize labeled data (Bar-Haim et al., 2005), where the language model is learned from the training data.

Extensive research has been done on word segmentation, where, unlike in our study, the segmentation is evaluated for every *word*, regardless of its context. Creutz (2003) presents an algorithm for unsupervised segmentation under these assumptions. He proposes a probabilistic model which

[1]Transcription is according to (Ornan, 2005), except for Shin which is denoted by "$".

utilizes the distributions of morpheme length and frequency to estimate the quality of the induced morphemes. Dasgupta and Ng (2007) improves over (Creutz, 2003) by suggesting a simpler approach. They segment a prefix using the word frequency with and without a prefix. Other recent studies that follow the context-independent setup include (Creutz and Lagus, 2005; Keshava and Pitler, 2005; Demberg, 2007). They test their methods on English, Finnish and Turkish. All of these studies, however, assume context-independency of segmentation, disregarding the ambiguity that may come from context. This makes it problematic to apply the proposed methods to context-dependent morphology types as in Hebrew and Arabic.

The guiding goal in the present paper is the concept acquisition problem. Concept acquisition of different kinds has been studied extensively. The two main classification axes for this task are the type of human input and annotation, and the basic algorithmic approach used. The two main algorithmic approaches are clustering of context feature vectors and pattern-based discovery.

The first approach is to map each word to a feature vector and cluster these vectors. Example of such algorithms are (Pereira et al., 1993) and (Lin, 1998) that use syntactic features in the vector definition. Pantel and Lin (2002) improves on the latter by clustering by committee.

Recently, there is a growing interest in the second main algorithmic approach, usage of lexico-syntactic patterns. Patterns have been shown to produce more accurate results than feature vectors, at a lower computational cost on large corpora (Pantel et al., 2004). Thus (Dorow et al., 2005) discover categories using two basic pre-specified patterns ("x and y", "x or y").

Some recent studies have proposed frameworks that attempt to avoid any implicit or explicit pre-specification of patterns. Davidov and Rappoport (2006) proposed a method that detects function words by their high frequency, and utilizes these words for the discovery of symmetric patterns. Their method is based on two assumptions: (1) some function words in the language symmetrically connect words belonging to the same category; (2) such function words can be detected as the most frequent words in language. While these assumptions are reasonable for many languages, for some morphologically rich languages the second assumption may fail. This is due to the fact that some languages like Hebrew and Arabic may express relationships not by isolated function words but by morphemes attached in writing to other words.

As an example, consider the English word "and', which was shown to be very useful in concept acquisition (Dorow et al., 2005). In Hebrew this word is usually expressed as the morpheme "w" attached to the second word in a conjunction ("... *w*sws" – "... and horse"). Patterns discovered by such automatic pattern discovery algorithms are based on isolated words, and hence fail to capture "and"-based relationships that are very useful for detection of words belonging to the same concept. Davidov and Rappoport (2006) reports very good results for English and Russian. However, no previous work applies a fully unsupervised concept acquisition for Hebrew.

In our study we combine their concept acquisition framework with a simple unsupervised word segmentation technique. Our evaluation confirms the weakness of word-based frameworks for morphology-rich languages such as Hebrew, and shows that utilizing the proposed word segmentation can overcome this weakness while keeping the concept acquisition approach fully unsupervised.

3 Unsupervised Discovery of Word Categories

In this study we use word segmentation to improve the (Davidov and Rappoport, 2006) method for discovery of word categories, sets of words sharing a significant aspect of their meaning. An example for such a discovered category is the set of verbs {dive, snorkel, swim, float, surf, sail, drift, ...}. Below we briefly describe this category acquisition algorithm.

The algorithm consists of three stages as follows. First, it discovers a set of pattern candidates, which are defined by a combination of high frequency words (denoted by H) and slots for low frequency (content) words (denoted by C). An example for such a pattern candidate is 'x belongs to y', where 'x' and 'y' stand for content word slots. The patterns are found according to a predefined set of possible meta-patterns. The meta-patterns are language-independent[2] and consist of up to 4

[2]They do not include any specific words, only a relative order of high/low frequency words, and hence can be used on

words in total, from which two are (non-adjacent) content words. Four meta-patterns are used: CHC, CHCH, CHHC, HCHC.

Second, those patterns which give rise to symmetric lexical relationships are identified. The meaning of phrases constructed from those patterns is (almost) invariant to the order of the content words contained in them. An example for such a pattern is 'x and y'. In order to identify such useful patterns, for each pattern we build a graph following (Widdows and Dorow, 2002). The graph is constructed from a node for each content word, and a directed arc from the node 'x' to 'y' if the corresponding content words appear in the pattern such that 'x' precedes 'y'. Then we calculate several symmetry measures on the graph structure and select the patterns with best values for these measures.

The third stage is the generation of categories. We extract tightly connected sets of words from the unified graph which combines all graphs of selected patterns. Such sets of words define the desired categories.

The patterns which include the 'x and y' substring are among the most useful patterns for generation of categories (they were used in (Dorow et al., 2005) and discovered in all 5 languages tested in (Davidov and Rappoport, 2006)). However, in Hebrew such patterns can not be found in the same way, since the function word 'and' is the prefix 'w' and not a standalone high frequency word.

Another popular set of patterns are ones including 'x or y'. Such patterns can be identified in Hebrew, as 'or' in Hebrew is a separate word. However, even in this case, the content word represented by 'x' or 'y' may appear with a prefix. This damages the construction of the pattern graph, since two different nodes may be created instead of one – one for a regular content word, the other for the same word with a prefix. Consequently, it is reasonable to assume that segmenting the corpus in advance should improve the results of discovery of word categories.

4 Word Segmentation Algorithm

We assume we know the small and closed set of grammatical function word prefix sequences in the language[3]. Our input is a sentence, and our objective is to return the correct segmentation of the sentence. A sentence L is a sequence of words $\{w_1, w_2, ..., w_n\}$. A segmentation S_i of L is a sequence of morphemes $\{m_1, m_2, ..., m_k\}$ and $l(S_i)$ is the number of morphemes in the sequence. Note that $l(S_i)$ may be different for each segmentation. The best segmentation S will be calculated by:

$$P(S_i) = p(m_1)p(m_2|m_1)\prod_{i=3}^{l(S_i)} p(m_i|m_{i-1}m_{i-2})$$

$$S = \arg\max_{S_i} P(S_i)$$

Calculation of joint probabilities requires a trigram model of the language. Below we describe the construction of the trigram model and then we detail the algorithm for efficient calculation of S.

4.1 Construction of trigram model

Creating the trigram language model is done in two stages: (1) we segment a corpus automatically, and (2) we learn a trigram language model from the segmented corpus.

4.1.1 Initial corpus segmentation

For initial corpus segmentation, we define a statistical measure for the segmentation of individual words. Let wx be a word, such that w is the prefix of the word composed of a sequence of function word prefixes and x is a string of letters. Let $f(x)$ be the frequency of the word x in the corpus. Denote by al the average length of the strings (with prefixes) in the language. This can be easily estimated from the corpus – every string that appears in the corpus is counted once. $l(x)$ is the number of characters in the word x. We utilize two parameters G, H, where $G < H$ (we used $G = 2.5, H = 3.5$) and define the following functions :

$$factor(x) = \begin{cases} \frac{al-G-l(x)}{al-H} & l(x) < al - G \\ 0 & otherwise \end{cases}$$

$$Rank(wx) = \frac{f(wx)}{f(wx) + f(x)} + factor(x)$$

Note that the expression $\frac{f(wx)}{f(wx)+f(x)}$ is a number in $(0, 1]$, inversely correlated with the frequency of the prefixed word. Thus higher $Rank(wx)$ values indicate that the word is less likely to be composed of the prefix w followed by the word x.

any languages with explicit word segmentation.

[3]Unlike development of labeled training data, handcrafting such a closed set is straightforward for many languages and does not requires any significant time/human labor

The expression $\frac{al-G-l(x)}{al-H}$ is a number in $(0, 1]$, therefore $factor(x) \in [0, 1]$. H is $G - 1$ in order to keep the expression smaller than 1. The term $factor(x)$ is greater as x is shorter. The factor is meant to express the fact that short words are less likely to have a prefix. We have examined this in Hebrew – as there are no words of length 1, two letter words have no prefix. We have analyzed 102 randomly chosen three letter words, and found that only 19 of them were prefixed words. We have analyzed 100 randomly chosen four letter words, and found that 40 of them were prefixed words. The result was about the same for five letter words. In order to decide whether a word needs to be separated, we define a threshold $T \in [0, 1]$. We allow word separation only when $Rank(wx)$ is lower than T. When there are more than two possible sequences of function word prefixes ("$mhsws$","$m\ hsws$", "$mh\ sws$"), we choose the segmentation with the lower rank.

4.1.2 Learning the trigram model

The learning of the language model is based on counts of the corpus, assigning a special symbol, "u/k" (unknown) for all words that do not appear in the corpus. As estimated by (Lee et al., 2003), we set the probability of "u/k" to be $1E - 9$. The value of the symbol "u/k" was observed to be significant. We found that the value proposed by (Lee et al., 2003) for Arabic gives good results also for Hebrew.

4.2 Dynamic programming approach for word segmentation

The naive method to find S is to iterate over all possible segmentations of the sentence. This method may fail to handle long sentences, as the number of segmentations grows exponentially with the length of the sentence. To overcome this problem, we use dynamic programming.

Each morpheme has an index i to its place in a segmentation sequence. Iteratively, for index i, for every morpheme which appears in some segmentation in index i, we calculate the best segmentation of the sequence $m_1 \dots m_i$. Two problems arise here: (1) we need to calculate which morphemes may appear in a given index; (2) we need to constrain the calculation, such that only valid segmentations would be considered.

To calculate which morphemes can appear in a given index we define the object *Morpheme*. It contains the morpheme (string), the index of a word in the sentence the morpheme belongs to, reference to the preceding *Morpheme* in the same word, and indication whether it is the last morpheme in the word. For each index of the sentence segmentation, we create a list of *Morphemes* (index-list).

For each word w_i, and for segmentation $m_i^1, .., m_i^k$, we create *Morphemes* $M_i^1, .., M_i^k$. We traverse sequentially the words in the sentence, and for each segmentation we add the sequence of *Morphemes* to all possible index-lists. The index-list for the first *Morpheme* M_i^1 is the combination of successors of all the index-lists that contain a *Morpheme* M_{i-1}^k. The constraints are enforced easily – if a *Morpheme* M_i^j is the first in a word, the preceding *Morpheme* in the sequence must be the last *Morpheme* of the previous word. Otherwise, the preceding *Morpheme* must be M_i^{j-1}, which is referenced by M_i^j.

4.3 Limitations

While our model handles the majority of cases, it does not fully comply with a linguistic analysis of Hebrew, as there are a few minor exceptions. We assumed that there is no ambiguity in the function word prefixes. This is not entirely correct, as in Hebrew we have two different kinds of exceptions for this rule. For example, the prefix "k$" (when), can also be interpreted as the prefix "k" (as) followed by the prefix "$" (that). As the second interpretation is rare, we always assumed it is the prefix "k$". This rule was applied wherever an ambiguity exists. However, we did not treat this problem as it is very rare, and in the development set and test set it did not appear even once.

A harder problem is encountered when processing the word "bbyt". Two interpretations could be considered here: "b byt" ("in a house"), and "b h byt" ("in the house"). Whether this actually poses a problem or not depends on the application. We assume that the correct segmentation here is "b byt". Without any additional linguistic knowledge (for example, diacritical vowel symbols should suffice in Hebrew), solving these problems requires some prior discriminative data.

5 Evaluation and Results

We evaluate our algorithm in two stages. First we test the quality of our unsupervised word segmentation framework on Hebrew and Arabic, comparing our segmentation results to a manually anno-

	With $factor(x)$				Without $factor(x)$			
T	Prec.	Recall	F-Measure	Accuracy	Prec.	Recall	F-Measure	Accuracy
0.70	0.844	0.798	0.820	0.875	0.811	0.851	0.830	0.881
0.73	0.841	0.828	0.834	0.883	0.808	0.866	0.836	0.884
0.76	0.837	0.846	0.841	0.886	0.806	0.882	0.842	0.887
0.79	**0.834**	**0.870**	**0.851**	**0.893**	0.803	0.897	0.847	0.890
0.82	0.826	0.881	0.852	0.892	0.795	0.904	0.846	0.888
0.85	0.820	0.893	0.854	0.892	0.787	0.911	0.844	0.886
0.88	0.811	0.904	0.855	0.891	0.778	0.917	0.841	0.882

Table 1: Ranks vs. Threshold T for Hebrew.

	With $factor(x)$				Without $factor(x)$			
T	Prec.	Recall	F-Measure	Accuracy	Prec.	Recall	F-Measure	Accuracy
0.91	0.940	0.771	0.846	0.892	0.903	0.803	0.850	0.891
0.93	0.930	0.797	0.858	0.898	0.903	0.840	0.870	0.904
0.95	0.931	0.810	0.866	0.904	0.902	0.856	0.878	0.909
0.97	0.927	0.823	0.872	0.906	0.896	0.869	0.882	0.911
0.99	0.925	0.848	0.872	0.915	0.878	0.896	0.886	0.913
1.00	**0.923**	**0.852**	**0.886**	**0.915**	0.841	0.896	0.867	0.895

Table 2: Ranks vs. Threshold T for Arabic.

Algorithm	P	R	F	A
Rank seg.	**0.834**	**0.870**	**0.851**	**0.893**
Baseline	0.561	0.491	0.523	0.69
Morfessor	0.630	0.689	0.658	0.814

Table 3: Segmentation results comparison.

tated gold standard. Then we incorporate word segmentation into a concept acquisition framework and compare the performance of this framework with and without word segmentation.

5.1 Corpora and annotation

For our experiments in Hebrew we used a 19MB Hebrew corpus obtained from the "Mila" Knowledge Center for Processing Hebrew[4]. The corpus consists of 143,689 different words, and a total of 1,512,737 word tokens. A sample text of size about 24,000 words was taken from the corpus, manually segmented by human annotators and used as a gold standard in our segmentation evaluation. In order to estimate the quality of our algorithm for Arabic, we used a 7MB Arabic news items corpus, and a similarly manually annotated test text of 4715 words. The Arabic corpus is too small for meaningful category discovery, so we used it only in the segmentation evaluation.

5.2 Evaluation of segmentation framework

In order to estimate the performance of word segmentation as a standalone algorithm we applied our algorithm on the Hebrew and Arabic corpora, using different parameter settings. We first calculated the word frequencies, then applied initial segmentation as described in Section 4. Then we used SRILM (Stolcke, 2002) to learn the trigram model from the segmented corpus. We utilized Good-Turing discounting with Katz backoff, and we gave words that were not in the training set the constant probability $1E-9$. Finally we utilized the obtained trigram model to select sentence segmentations. To test the influence of the $factor(x)$ component of the $Rank$ value, we repeated our experiment with and without usage of this component. We also ran our algorithm with a set of different threshold T values in order to study the influence of this parameter.

Tables 1 and 2 show the obtained results for Hebrew and Arabic respectively. Precision is the ratio of correct prefixes to the total number of detected prefixes in the text. Recall is the ratio of prefixes that were split correctly to the total number of prefixes. Accuracy is the number of correctly segmented words divided by the total number of words.

As can be seen from the results, the best F-score with and without usage of the $factor(x)$ component are about the same, but usage of this component gives higher precision for the same F-score. From comparison of Arabic and Hebrew performance we can also see that segmentation decisions for the task in Arabic are likely to be easier, since the accuracy for T=1 is very high. It means that, unlike in Hebrew (where the best results were obtained for T=0.79), a word which starts with a pre-

[4]http://mila.cs.technion.ac.il.

Method	us	k-means	random
avg 'shared meaning'(%)	**85**	24.61	10
avg triplet score(1-4)	**1.57**	2.32	3.71
avg category score(1-10)	**9.35**	6.62	3.5

Table 4: Human evaluation results.

abuse, robbery, murder, assault, extortion
good, cheap, beautiful, comfortable
son, daughter, brother, parent
when, how, where
essential, important, central, urgent

Table 5: A sample from the lexical categories discovered in Hebrew (translated to English).

fix should generally be segmented.

We also compared our best results to the baseline and to previous work. The baseline draws a segmentation uniformly for each word, from the possible segmentations of the word. In an attempt to partially reproduce (Creutz and Lagus, 2005) on our data, we also compared our results to the results obtained from Morfessor Categories-MAP, version 0.9.1 (Described in (Creutz and Lagus, 2005)). The Morfessor Categories-MAP algorithm gets a list of words and their frequencies, and returns the segmentation for every word. Since Morfessor may segment words with prefixes which do not exist in our predefined list of valid prefixes, we did not segment the words that had illegal prefixes as segmented by Morfessor.

Results for this comparison are shown in Table 3. Our method significantly outperforms both the baseline and Morfessor-based segmentation. We have also tried to improve the language model by a self training scheme on the same corpus but we observed only a slight improvement, giving 0.848 Precision and 0.872 Recall.

5.3 Discovery of word categories

We divide the evaluation of the word categories discovery into two parts. The first is evaluating the improvement in the quantity of found lexical categories. The second is evaluating the quality of these categories. We have applied the algorithm to a Hebrew corpus of size 130MB[5], which is sufficient for a proof of concept. We compared the output of the categories discovery on two different settings, with function word separation and without such separation. In both settings we omitted all punctuation symbols. In both runs of the algorithm we used the same parameters. Eight symmetric patterns were automatically chosen for each run. Two of the patterns that were chosen by the algorithm in the unseparated case were also chosen in the separated case.

[5]Again obtained from the "Mila" Knowledge Center for Processing Hebrew.

	N	A	J
With Separation	148	4.1	1
No Separation	36	2.9	0

Table 6: Lexical categories discovery results comparison. N: number of categories. A: average category size. J: 'junk' words.

5.3.1 Manual estimation of category quality

Evaluating category quality is challenging since no exhaustive lists or gold standards are widely accepted even in English, certainly so in resource-poor languages such as Hebrew. Hence we follow the human judgment evaluation scheme presented in (Davidov and Rappoport, 2006), for the categories obtained from the segmented corpus.

We compared three methods of word categories discovery. The first is random sampling of words into categories. The second is k-means, where each word is mapped to a vector, and similarity is calculated as described in (Pantel and Lin, 2002). We applied k-means to the set of vectors, with similarity as a distance function. If a vector had low similarity with all means, we leave it unattached. Therefore some clusters contained only one vector. Running the algorithm 10 times, with different initial means each time, produced 60 clusters with three or more words. An interesting phenomenon we observed is that this method produces very nice clusters of named entities. The last method is the one in (Davidov and Rappoport, 2006).

The experiment contained two parts. In Part I, subjects were given 40 triplets of words and were asked to rank them using the following scale: (1) the words definitely share a significant part of their meaning; (2) the words have a shared meaning but only in some context; (3) the words have a shared meaning only under a very unusual context/situation; (4) the words do not share any meaning; (5) I am not familiar enough with some/all of the words.

The 40 triplets were obtained as follows. 20 of our categories were selected at random from the non-overlapping categories we have discovered, and three words were selected from each of these

at random. 10 triplets were selected in the same manner from the categories produced by k-means, and 10 triplets were selected at random from content words in the same document.

In Part II, subjects were given the full categories represented by the triplets that were graded as 1 or 2 in Part I (the full "good" categories in terms of sharing of meaning). Subjects were asked to grade the categories from 1 (worst) to 10 (best) according to how much the full category had met the expectations they had when seeing only the triplet.

Nine people participated in the evaluation. A summary of the results is given in Table 4.

The categories obtained from the unsegmented corpus are too few and too small for a significant evaluation. Therefore we applied the evaluation scheme only for the segmented corpus.

The results from the segmented corpus contain some interesting categories, with a 100% precision, like colors, Arab leaders, family members and cities. An interesting category is {Arabic, English, Russian, French, German, Yiddish, Polish, Math}. A sample of some other interesting categories can be seen in Table 5.

5.3.2 Segmentation effect on category discovery

In Table 6, we find that there is a major improvement in the number of acquired categories, and an interesting improvement in the average category size. One might expect that as a consequence of an incorrect segmentation of a word, "junk" words may appear in the discovered categories. As can be seen, only one "junk" word was categorized.

Throughout this paper we have assumed that function word properties of languages such as Hebrew and Arabic decrease performance of whole-word pattern-based concept acquisition methods. To check this assumption, we have applied the concept acquisition algorithm on several web-based corpora of several languages, while choosing corpora size to be exactly equal to the size of the Hebrew corpus (130Mb) and utilizing exactly the same parameters. We did not perform quality evaluation[6], but measured the number of concepts and concept size. Indeed the number of categories was (190, 170, 159, 162, 150, 29) for Russian, English, Spanish, French, Turkish and Arabic respectively, clearly inferior for Arabic in comparison to these European and Slavic languages. A similar tendency was observed for average concept size. At the same time prefix separation does help to extract 148 concepts for Hebrew, making it nearly inline with other languages. In contrast, our preliminary experiments on English and Russian suggest that the effect of applying similar morphological segmentation on these languages in insignificant.

In order to test whether more data can substitute segmentation even for Hebrew, we have obtained by means of crawling and web queries a larger (while potentially much more noisy) web-based 2GB Hebrew corpus which is based on forum and news contents. Our goal was to estimate which unsegmented corpus size (if any) can bring similar performance (in terms of concept number, size and quality). We gradually increased corpus size and applied the concept acquisition algorithm on this corpus. Finally, we have obtained similar, nearly matching, results to our 130MB corpus for a 1.2GB Hebrew subcorpus of the 2GB Hebrew corpus. The results remain stable for 4 different 1.2GB subsets taken from the same 2GB corpus. This suggests that while segmentation can be substituted with more data, it may take roughly x10 more data for Hebrew to obtain the same results without segmentation as with it.

6 Summary

We presented a simple method for separating function word prefixes from words. The method requires very little language-specific knowledge (the prefixes), and it can be applied to any morphologically rich language. We showed that this segmentation dramatically improves lexical acquisition in Hebrew, where nearly $\times 10$ data is required to obtain the same number of concepts without segmentation.

While in this paper we evaluated our framework on the discovery of concepts, we have recently proposed fully unsupervised frameworks for the discovery of different relationship types (Davidov et al., 2007; Davidov and Rappoport, 2008a; Davidov and Rappoport, 2008b). Many of these methods are mostly based on function words, and may greatly benefit from the proposed segmentation framework.

References

Meni Adler, Michael Elhadad, 2006. An Unsupervised Morpheme-Based HMM for Hebrew Morphological Disambiguation. *ACL '06.*

[6] Brief manual examination suggests no significant drops in concept quality.

Roy Bar-Haim, Khalil Simaan, Yoad Winter, 2005. Choosing an Optimal Architecture for Segmentation and POS-Tagging of Modern Hebrew. *ACL Workshop on Computational Approaches to Semitic Languages '05.*

Mathias Creutz, 2003. Unsupervised Segmentation of Words Using Prior Distributions of Morph Length and Frequency. *ACL '03.*

Mathias Creutz and Krista Lagus, 2005. Unsupervised Morpheme Segmentation and Morphology Induction from Text Corpora Using Morfessor 1.0. *In Computer and Information Science, Report A81, Helsinki University of Technology.*

Sajib Dasgupta and Vincent Ng, 2007. High Performance, Language-Independent Morphological Segmentation. *NAACL/HLT '07.*

Dmitry Davidov, Ari Rappoport, 2006. Efficient Unsupervised Discovery of Word Categories Using Symmetric Patterns and High Frequency Words. *ACL '06.*

Dmitry Davidov, Ari Rappoport, Moshe Koppel, 2007. Fully Unsupervised Discovery of Concept-Specific Relationships by Web Mining. *ACL '07.*

Dmitry Davidov, Ari Rappoport, 2008a. Classification of Semantic Relationships between Nominals Using Pattern Clusters. *ACL '08.*

Dmitry Davidov, Ari Rappoport, 2008b. Unsupervised Discovery of Generic Relationships Using Pattern Clusters and its Evaluation by Automatically Generated SAT Analogy Questions. *ACL '08.*

Vera Demberg, 2007. A Language-Independent Unsupervised Model for Morphological Segmentation. *ACL '07.*

Beate Dorow, Dominic Widdows, Katarina Ling, Jean-Pierre Eckmann, Danilo Sergi, Elisha Moses, 2005. Using Curvature and Markov Clustering in Graphs for Lexical Acquisition and Word Sense Discrimination. *MEANING '05.*

Samarth Keshava, Emily Pitler, 2006. A Simpler, Intuitive Approach to Morpheme Induction. *In Proceedings of 2nd Pascal Challenges Workshop, Venice, Italy.*

Young-Suk Lee, Kishore Papineni, Salim Roukos, Ossama Emam, Hany Hassan, 2003. Language Model Based Arabic Word Segmentation. *ACL '03.*

Moshe Levinger, Uzzi Ornan, Alon Itai, 1995. Learning Morpho-Lexical Probabilities from an Untagged Corpus with an Application to Hebrew. *Comput. Linguistics, 21:383:404.*

Dekang Lin, 1998. Automatic Retrieval and Clustering of Similar Words. *COLING '98.*

Uzzi Ornan, 2005. Hebrew in Latin Script. *Lesonenu, LXIV:137:151 (in Hebrew).*

Patrick Pantel, Dekang Lin, 2002. Discovering Word Senses from Text. *SIGKDD '02.*

Patrick Pantel, Deepak Ravichandran, Eduard Hovy, 2004. Towards Terascale Knowledge Acquisition. *COLING '04.*

Fernando Pereira, Naftali Tishby, Lillian Lee, 1993. Distributional Clustering of English Words. *ACL '93.*

Benjamin Snyder, Regina Bazilay, 2008. Unsupervised Multilingual Learning for Morphological Segmentation. *ACL/HLT '08.*

Andreas Stolcke, 2002. SRILM – an Extensible Language Modeling Toolkit. *ICSLP, pages 901-904, Denver, Colorado.*

Dominic Widdows, Beate Dorow, 2002. A Graph Model for Unsupervised Lexical Acquisition. *COLING '02.*