An Unsupervised Morpheme-Based HMM for Hebrew Morphological Disambiguation

Meni Adler
Department of Computer Science
Ben Gurion University of the Negev
84105 Beer Sheva, Israel
adlerm@cs.bgu.ac.il

Michael Elhadad
Department of Computer Science
Ben Gurion University of the Negev
84105 Beer Sheva, Israel
elhadad@cs.bgu.ac.il

Abstract

Morphological disambiguation is the process of assigning one set of morphological features to each individual word in a text. When the word is ambiguous (there are several possible analyses for the word), a disambiguation procedure based on the word context must be applied. This paper deals with morphological disambiguation of the Hebrew language, which combines morphemes into a word in both agglutinative and fusional ways. We present an unsupervised stochastic model – the only resource we use is a morphological analyzer – which deals with the data sparseness problem caused by the affixational morphology of the Hebrew language.

We present a text encoding method for languages with affixational morphology in which the knowledge of word formation rules (which are quite restricted in Hebrew) helps in the disambiguation. We adapt HMM algorithms for learning and searching this text representation, in such a way that segmentation and tagging can be learned in parallel in one step. Results on a large scale evaluation indicate that this learning improves disambiguation for complex tag sets. Our method is applicable to other languages with affix morphology.

1 Introduction

Morphological disambiguation is the process of assigning one set of morphological features to each individual word in a text, according to the word context.

In this work, we investigate morphological disambiguation in Modern Hebrew. We explore unsupervised learning method, which is more challenging than the supervised case. The main motivation for this approach is that despite the development of annotated corpora in Hebrew[1], there is still not enough data available for supervised training. The other reason, is that unsupervised methods can handle the dynamic nature of Modern Hebrew, as it evolves over time.

In the case of English, because morphology is simpler, morphological disambiguation is generally covered under the task of part-of-speech tagging. The main morphological variations are embedded in the tag name (for example, Ns and Np for noun singular or plural). The tagging accuracy of supervised stochastic taggers is around 96%-97% (Manning and Schutze, 1999, 10.6.1). Merialdo (1994) reports an accuracy of 86.6% for an unsupervised word-based HMM, trained on a corpus of 42,186 sentences (about 1M words), over a tag set of 159 different tags. Elworthy (1994), in contrast, reports an accuracy of 75.49%, 80.87% and 79.12% for unsupervised word-based HMM trained on parts of the LOB corpora, with a tagset of 134 tags. With good initial conditions, such as good approximation of the tag distribution for each word, Elworthy reports an improvement to 94.6%, 92.27% and 94.51% on the same data sets. Merialdo, on the other hand, reports an improvement to 92.6% and 94.4% for the case where 100 and 2000 sentences of the training corpus are manually tagged.

Modern Hebrew is characterized by rich morphology, with a high level of ambiguity. On average, in our corpus, the number of possible analyses per word reached 2.4 (in contrast to 1.4 for English). In Hebrew, several morphemes combine into a single word in both agglutinative and fusional ways. This results in a potentially high number of tags for each word.

In contrast to English tag sets whose sizes range from 48 to 195, the number of tags for Hebrew, based on all combinations of the morphological attributes (part-of-speech, gender, number, person, tense, status, and the affixes' properties[2]),

[*] This work is supported by the Lynn and William Frankel Center for Computer Sciences, and by the Knowledge Center for Hebrew Processing, Israel Science Ministry.

[1] The Knowledge Center for Hebrew processing is developing such corpora: http://mila.cs.technion.ac.il/

[2] The list of morphological attributes is described in (Yona and Wintner, 2005). An in-depth discussion of the Hebrew word form is provided in (Allon, 1995, pp.

Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL, pages 665–672,
Sydney, July 2006. ©2006 Association for Computational Linguistics

can grow theoretically to about 300,000 tags. In practice, we found only 1,934 tags in a corpus of news stories we gathered, which contains about 6M words.
The large size of such a tag set (about 10 times larger than the most comprehensive English tag set) is problematic in term of data sparseness. Each morphological combination appears rarely, and more samples are required in order to learn the probabilistic model.

In this paper, we hypothesize that the large set of morphological features of Hebrew words, should be modeled by a compact morpheme model, based on the segmented words (into prefix, baseform, and suffix). Our main result is that best performance is obtained when learning segmentation and morpheme tagging in one step, which is made possible by an appropriate text representation.

2 Hebrew and Arabic Tagging - Previous Work

Several works have dealt with Hebrew tagging in the past decade. In Hebrew, morphological analysis requires complex processing according to the rules of Hebrew word formation. The task of a morphological analyzer is to produce all possible analyses for a given word. Recent analyzers provide good performance and documentation of this process (Yona and Wintner, 2005; Segal, 2000). Morphological analyzers rely on a dictionary, and their performance is, therefore, impacted by the occurrence of unknown words. The task of a morphological disambiguation system is to pick the most likely analysis produced by an analyzer in the context of a full sentence.

Levinger et al. (1995) developed a context-free method in order to acquire the morpho-lexical probabilities, from an untagged corpus. Their method handles the data sparseness problem by using a set of similar words for each word, built according to a set of rules. The rules produce variations of the morphological properties of the word analyses. Their tests indicate an accuracy of about 88% for context-free analysis selection based on the approximated analysis distribution. In tests we reproduced on a larger data set (30K tagged words), the accuracy is only 78.2%. In order to improve the results, the authors recommend merging their method together with other morphological disambiguation methods – which is the approach we pursue in this work.

Levinger's morphological disambiguation system (Levinger, 1992) combines the above approximated probabilities with an expert system, based on a manual set of 16 syntactic constraints . In the first phase, the expert system is applied, disambiguating 35% of the ambiguous words with an accuracy of 99.6%. In order to increase the applicability of the disambiguation, approximated probabilities are used for words that were not disambiguated in the first stage. Finally, the expert system is used again over the new probabilities that were set in the previous stage. Levinger reports an accuracy of about 94% for disambiguation of 85% of the words in the text (overall 80% disambiguation). The system was also applied to prune out the least likely analyses in a corpus but without, necessarily, selecting a single analysis for each word. For this task, an accuracy of 94% was reported while reducing 92% of the ambiguous analyses.

Carmel and Maarek (1999) use the fact that on average 45% of the Hebrew words are unambiguous, to rank analyses, based on the number of disambiguated occurrences in the text, normalized by the total number of occurrences for each word. Their application – indexing for an information retrieval system – does not require all of the morphological attributes but only the lemma and the PoS of each word. As a result, for this case, 75% of the words remain with one analysis with 95% accuracy, 20% with two analyses and 5% with three analyses.

Segal (2000) built a transformation-based tagger in the spirit of Brill (1995). In the first phase, the analyses of each word are ranked according to the frequencies of the possible lemmas and tags in a training corpus of about 5,000 words. Selection of the highest ranked analysis for each word gives an accuracy of 83% of the test text – which consists of about 1,000 words. In the second stage, a transformation learning algorithm is applied (in contrast to Brill, the observed transformations are not applied, but used for re-estimation of the word couples probabilities). After this stage, the accuracy is about 93%. The last stage uses a bottom-up parser over a hand-crafted grammar with 150 rules, in order to select the analysis which causes the parsing to be more accurate. Segal reports an accuracy of 95%. Testing his system over a larger test corpus, gives poorer results: Lembersky (2001) reports an accuracy of about 85%.

Bar-Haim et al. (2005) developed a word segmenter and PoS tagger for Hebrew. In their architecture, words are first segmented into morphemes, and then, as a second stage, these morphemes are tagged with PoS. The method proceeds in two sequential steps: segmentation into morphemes, then tagging over morphemes. The segmentation is based on an HMM and trained over a set of 30K annotated words. The segmentation step reaches an accuracy of 96.74%. PoS tagging, based on unsupervised estimation which combines a small annotated corpus with an untagged corpus of 340K

24–86).

Word	Segmentation	Tag	Translation
bclm	bclm	PNN	name of a human rights association (Betselem)
bclm	bclm	VB	while taking a picture
bclm	bcl-m	cons-NNM-suf	their onion
bclm	b-cl-m	P1-NNM-suf	under their shadow
bclm	b-clm	P1-NNM	in a photographer
bclm	b-clm	P1-cons-NNM	in a photographer
bclm	b-clm	P1-h-NNM	in the photographer
hn'im	h-n'im	P1-VBR	that are moving
hn'im	hn'im	P1-h-JJM	the lovely
hn'im	hn'im	VBP	made pleasant

Table 1: Possible analyses for the words *bclm hn'im*

words by using smoothing technique, gives an accuracy of 90.51%.

As noted earlier, there is as yet no large scale Hebrew annotated corpus. We are in the process of developing such a corpus, and we have developed tagging guidelines (Elhadad et al., 2005) to define a comprehensive tag set, and assist human taggers achieve high agreement. The results discussed above should be taken as rough approximations of the real performance of the systems, until they can be re-evaluated on such a large scale corpus with a standard tag set.

Arabic is a language with morphology quite similar to Hebrew. Theoretically, there might be 330,000 possible morphological tags, but in practice, Habash and Rambow (2005) extracted 2,200 different tags from their corpus, with an average number of 2 possible tags per word. As reported by Habash and Rambow, the first work on Arabic tagging which used a corpus for training and evaluation was the work of Diab et al. (2004). Habash and Rambow were the first to use a morphological analyzer as part of their tagger. They developed a supervised morphological disambiguator, based on training corpora of two sets of 120K words, which combines several classifiers of individual morphological features. The accuracy of their analyzer is 94.8% – 96.2% (depending on the test corpus). An unsupervised HMM model for dialectal Arabic (which is harder to be tagged than written Arabic), with accurracy of 69.83%, was presented by Duh and Kirchhoff (2005). Their supervised model, trained on a manually annotated corpus, reached an accuracy of 92.53%.

Arabic morphology seems to be similar to Hebrew morphology, in term of complexity and data sparseness, but comparison of the performances of the baseline tagger used by Habash and Rambow – which selects the most frequent tag for a given word in the training corpus – for Hebrew and Arabic, shows some intriguing differences: 92.53% for Arabic and 71.85% for Hebrew. Furthermore, as mentioned above, even the use of a sophisticated context-free tagger, based on (Levinger et al., 1995), gives low accuracy of 78.2%. This might imply that, despite the similarities, morphological disambiguation in Hebrew might be harder than in Arabic. It could also mean that the tag set used for the Arabic corpora has not been adapted to the specific nature of Arabic morphology (a comment also made in (Habash and Rambow, 2005)).

We propose an unsupervised morpheme-based HMM to address the data sparseness problem. In contrast to Bar-Haim et al., our model combines segmentation and morphological disambiguation, in parallel. The only resource we use in this work is a morphological analyzer. The analyzer itself can be generated from a word list and a morphological generation module, such as the HSpell wordlist (Har'el and Kenigsberg, 2004).

3 Morpheme-Based Model for Hebrew

3.1 Morpheme-Based HMM

The lexical items of word-based models are the words of the language. The implication of this decision is that both lexical and syntagmatic relations of the model, are based on a *word-oriented tagset*. With such a tagset, it must be possible to tag any word of the language with at least one tag.

Let us consider, for instance, the Hebrew phrase *bclm hn'im*[3], which contains two words. The word *bclm* has several possible morpheme segmentations and analyses[4] as described in Table 1. In word-based HMM, we consider such a phrase to be generated by a Markov process, based on the word-oriented tagset of $N = 1934$ tags/states and about $M = 175K$ word types. Line W of Table 2 describes the size of a first-order word-based HMM, built over our corpus. In this model, we found 834 entries for the Π vector (which models the distribution of tags in first position in sentences) out of possibly $N = 1934$, about 250K entries for the A matrix (which models the transition probabilities from tag to tag) out of possibly $N^2 \approx 3.7M$, and about 300K entries for the B matrix (which models

[3] Transcription according to Ornan (2002).

[4] The tagset we use for the annotation follows the guidelines we have developed (Elhadad et al., 2005).

	States	PI	A	A2	B	B2
W	1934	834	250K	7M	300K	5M
M	202	145	20K	700K	130K	1.7M

Table 2: Model Sizes

the emission probabilities from tag to word) out of possibly $M \cdot N \approx 350M$. For the case of a second-order HMM, the size of the $A2$ matrix (which models the transition probabilities from two tags to the third one), grows to about 7M entries, where the size of the $B2$ matrix (which models the emission probabilities from two tags to a word) is about 5M. Despite the sparseness of these matrices, the number of their entries is still high, since we model the whole set of features of the complex word forms.

Let us assume, that the right segmentation for the sentence is provided to us – for example: *b clm hn‘im* – as is the case for English text. In such a way, the observation is composed of morphemes, generated by a Markov process, based on a morpheme-based tagset. The size of such a tagset for Hebrew is about 200, where the size of the Π,A,B,$A2$ and $B2$ matrices is reduced to 145, 16K, 140K, 700K, and 1.7M correspondingly, as described in line M of Table 2 – a reduction of 90% when compared with the size of a word-based model.

The problem in this approach, is that "someone" along the way, agglutinates the morphemes of each word leaving the observed morphemes uncertain. For example, the word *bclm* can be segmented in four different ways in Table 1, as indicated by the placement of the '-' in the Segmentation column, while the word *hn‘im* can be segmented in two different ways. In the next section, we adapt the parameter estimation and the searching algorithms for such uncertain output observation.

3.2 Learning and Searching Algorithms for Uncertain Output Observation

In contrast to standard HMM, the output observations of the above morpheme-based HMM are ambiguous. We adapted Baum-Welch (Baum, 1972) and Viterbi (Manning and Schutze, 1999, 9.3.2) algorithms for such uncertain observation. We first formalize the output representation and then describe the algorithms.

Output Representation The learning and searching algorithms of HMM are based on the output sequence of the underlying Markov process. For the case of a morpheme-based model, the output sequence is uncertain – we don't see the emitted morphemes but the words they form. If, for instance, the Markov process emitted the morphemes *b clm h n‘im*, we would see two words (*bclm hn‘im*) instead. In order to handle the output ambiguity, we use static knowledge of how morphemes are combined into a word, such as the four known combinations of the word *bclm*, the two possible combinations of the word *hn‘im*, and their possible tags within the original words. Based on this information, we encode the sentence into a structure that represents all the possible "readings" of the sentence, according to the possible morpheme combinations of the words, and their possible tags.

The representation consists of a set of vectors, each vector containing the possible morphemes and their tags for each specific "time" (sequential position within the morpheme expansion of the words of the sentence). A morpheme is represented by a tuple (symbol, state, prev, next), where *symbol* denotes a morpheme, *state* is one possible tag for this morpheme, *prev* and *next* are sets of indexes, denoting the indexes of the morphemes (of the previous and the next vectors) that precede and follow the current morpheme in the overall lattice, representing the sentence. Fig. 2 describes the representation of the sentence *bclm hn‘im*. An emission is denoted in this figure by its symbol, its state index, directed edges from its previous emissions, and directed edges to its next emissions.

In order to meet the condition of Baum-Eagon inequality (Baum, 1972) that the polynomial $P(O|\mu)$ – which represents the probability of an observed sequence O given a model μ – be homogeneous, we must add a sequence of special EOS (end of sentence) symbols at the end of each path up to the last vector, so that all the paths reach the same length.

The above text representation can be used to model multi-word expressions (MWEs). Consider the Hebrew sentence: *hw' ‘wrk dyn gdwl*, which can be interpreted as composed of 3 units (he lawyer great / he is a great lawyer) or as 4 units (he edits law big / he is editing an important legal decision). In order to select the correct interpretation, we must determine whether *‘wrk dyn* is an MWE. This is another case of uncertain output observation, which can be represented by our text encoding, as done in Fig. 1.

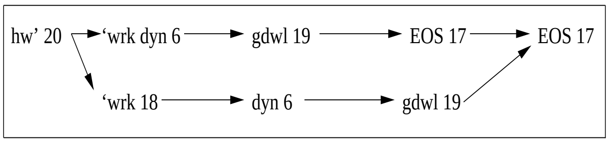

Figure 1: The sentence *hw' ‘wrk dyn gdwl*

This representation seems to be expensive in term of the number of emissions per sentence. However, we observe in our data that most of the words have only one or two possible segmentations, and most of the segmentations consist of at most one affix. In practice, we found the average number of emissions per sentence in our corpus (where each symbol is counted as the number of its predecessor emissions) to be 455, where the average number of words per sentence is about 18. That is, the cost of operating over an ambiguous sentence representation increases the size of the sentence (from 18 to 455), but on the other hand, it reduces the probabilistic model by a factor of 10 (as discussed above).

Morphological disambiguation over such a sequence of vectors of uncertain morphemes is similar to words extraction in automatic speech recognition (ASR)(Jurafsky and Martin, 2000, chp. 5,7). The states of the ASR model are phones, where each observation is a vector of spectral features. Given a sequence of observations for a sentence, the encoding – based on the lattice formed by the phones distribution of the observations, and the language model – searches for the set of words, made of phones, which maximizes the acoustic likelihood and the language model probabilities. In a similar manner, the supervised training of a speech recognizer combines a training corpus of speech wave files, together with word-transcription, and language model probabilities, in order to learn the phones model.

There are two main differences between the typical ASR model and ours: (1) an ASR decoder deals with one aspect - segmentation of the observations into a set of words, where this segmentation can be modeled at several levels: subphones, phones and words. These levels can be trained individually (such as training a language model from a written corpus, and training the phones model for each word type, given transcripted wave file), and then combined together (in a hierarchical model). Morphological disambiguation over uncertain morphemes, on the other hand, deals with both morpheme segmentation and the tagging of each morpheme with its morphological features. Modeling morpheme segmentation, within a given word, without its morphology features would be insufficient. (2) The supervised resources of ASR are not available for morphological disambiguation: we don't have a model of morphological features sequences (equivalent to the language model of ASR) nor a tagged corpus (equivalent to the transcripted wave files of ASR).

These two differences require a design which combines the two dimensions of the problem, in order to support unsupervised learning (and searching) of morpheme sequences and their morphological features, simultaneously.

Parameter Estimation We present a variation of the Baum-Welch algorithm (Baum, 1972) which operates over the lattice representation we have defined above. The algorithm starts with a probabilistic model μ (which can be chosen randomly or obtained from good initial conditions), and at each iteration, a new model $\bar{\mu}$ is derived in order to better explain the given output observations. For a given sentence, we define T as the number of words in the sentence, and $\bar{T}$ as the number of vectors of the output representation $O = \{o_t\}, 1 \leq t \leq \bar{T}$, where each item in the output is denoted by $o_t^l = (sym, state, prev, next), 1 \leq t \leq \bar{T}, 1 \leq l \leq |o_t|$. We define $\alpha(t,l)$ as the probability to reach o_t^l at time t, and $\beta(t,l)$ as the probability to end the sequence from o_t^l. Fig. 3 describes the expectation and the maximization steps of the learning algorithm for a first-order HMM. The algorithm works in $O(\dot{T})$ time complexity, where $\dot{T}$ is the total number of symbols in the output sequence encoding, where each symbol is counted as the size of its *prev* set.

Searching for best state sequence The searching algorithm gets an observation sequence O and a probabilistic model μ, and looks for the best state sequence that generates the observation. We define $\delta(t,l)$ as the probability of the best state sequence that leads to emission o_t^l, and $\psi(t,l)$ as the index of the emission at time $t-1$ that precedes o_t^l in the best state sequence that leads to it. Fig. 4 describes the adaptation of the Viterbi (Manning and Schutze, 1999, 9.3.2) algorithm to our text representation for first-order HMM, which works in $O(\dot{T})$ time.

4 Experimental Results

We ran a series of experiments on a Hebrew corpus to compare various approaches to the full morphological disambiguation and PoS tagging tasks. The training corpus is obtained from various newspaper sources and is characterized by the following statistics: 6M word occurrences, 178,580 distinct words, 64,541 distinct lemmas. Overall, the ambiguity level is 2.4 (average number of analyses per word).

We tested the results on a test corpus, manually annotated by 2 taggers according to the guidelines we published and checked for agreement. The test corpus contains about 30K words. We compared two unsupervised models over this data set: Word model [W], and Morpheme model [M]. We also tested two different sets of initial conditions. Uniform distribution [Uniform]: For each word, each analysis provided by the analyzer is estimated with an equal likelihood. Context Free approximation [CF]: We applied the CF algorithm of Levinger et al.(1995) to estimate the likelihood of each analysis.

Table 3 reports the results of full morphological disambiguation. For each morpheme and word models, three types of models were tested: [1] First-order HMM, [2-] Partial second-order HMM - only state transitions were modeled (excluding B2 matrix), [2] Second-order HMM (including the B2 matrix).

Analysis If we consider the tagger which selects the most probable morphological analysis for each

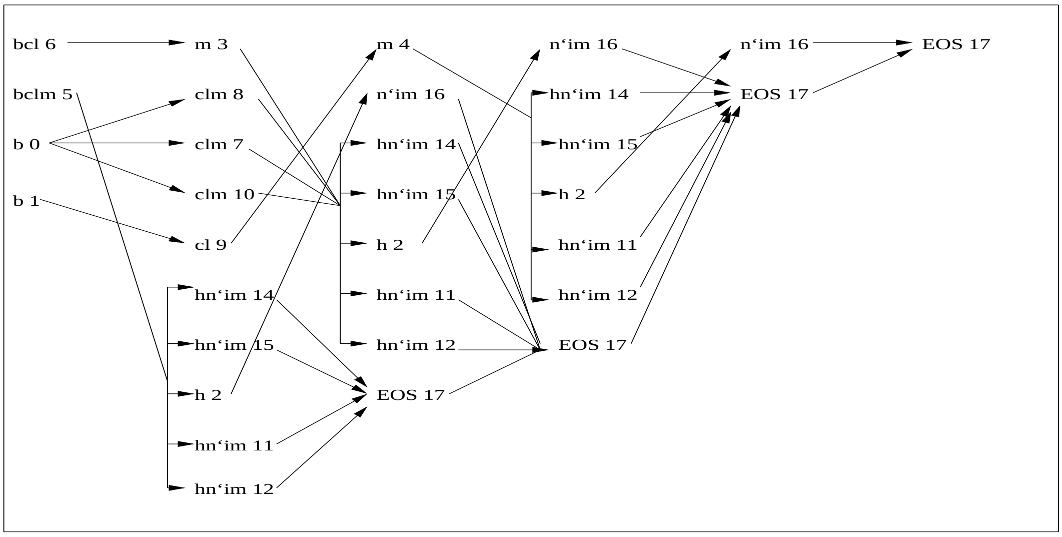

Figure 2: Representation of the sentence *bclm hn‘im*

Expectation

$$\alpha(1,l) = \pi_{o_1^l.state} b_{o_1^l.state,o_1^l.sym} \quad (1)$$

$$\alpha(t,l) = b_{o_t^l.state,o_t^l.sym} \sum_{l' \in o_t^l.prev} \alpha_(t-1,l') a_{o_{t-1}^{l'}.state,o_t^l.state}$$

$$\beta(\bar{T},l) = 1 \quad (2)$$

$$\beta(t,l) = \sum_{l' \in o_t^l.next} a_{o_t^l.state,o_{t+1}^{l'}.state} b_{o_{t+1}^{l'}.state,o_{t+1}^{l'}.sym} \beta(t+1,l')$$

Maximization

$$\bar{\pi}_i = \frac{\sum_{l:o_1^l.state=i} \alpha(1,l)\beta(1,l)}{\sum_l \alpha(1,l)\beta(1,l)} \quad (3)$$

$$\bar{a}_{i,j} = \frac{\sum_{t=2}^{\bar{T}} \sum_{l:o_t^l.state=j} \sum_{l' \in o_t^l.prev:o_{t-1}^{l'}.state=i} \alpha(t-1,l') a_{i,j} b_{j,o_t^l.sym} \beta(t,l)}{\sum_{t=1}^{\bar{T}-1} \sum_{l:o_t^l.state=i} \alpha(t,l)\beta(t,l)} \quad (4)$$

$$\bar{b}_{i,k} = \frac{\sum_{t=1}^{\bar{T}} \sum_{l:o_t^l.sym=k,o_t^l.state=i} \alpha(t,l)\beta(t,l)}{\sum_{t=1}^{\bar{T}} \sum_{l:o_t^l.state=i} \alpha(t,l)\beta(t,l)} \quad (5)$$

Figure 3: The learning algorithm for first-order model

Initialization

$$\delta(1,l) = \pi_{o_1^l.state} b_{o_1^l.state,o_1^l.sym} \quad (6)$$

Induction

$$\delta(t,l) = \max_{l' \in o_t^l.prev} \delta(t-1,l') a_{o_{t-1}^{l'}.state,o_t^l.state} b_{o_t^l.state,o_t^l.sym} \quad (7)$$

$$\psi(t,l) = \operatorname*{argmax}_{l' \in o_t^l.prev} \delta(t-1,l') a_{o_{t-1}^{l'}.state,o_t^l.state} b_{o_t^l.state,o_t^l.sym} \quad (8)$$

Termination and path readout

$$\bar{X}_{\bar{T}} = \operatorname*{argmax}_{1 \le l \le |\bar{T}|} \delta(\bar{T},l) \quad (9)$$

$$\bar{X}_t = \psi(t+1,\bar{X}_{t+1})$$

$$P(\bar{X}) = \max_{1 \le l \le |O_{\bar{T}}|} \delta(\bar{T},l) \quad (10)$$

Figure 4: The searching algorithm for first-order model

	Order	Uniform	CF
W	1	82.01	84.08
W	2-	80.44	85.75
W	2	79.88	85.78
M	1	81.08	84.54
M	2-	81.53	**88.5**
M	2	83.39	85.83

Table 3: Morphological Disambiguation

word in the text, according to Levinger et al. (1995) approximations, with accuracy of 78.2%, as the baseline tagger, four steps of error reduction can be identified. **(1)** Contextual information: The simplest first-order word-based HMM with uniform initial conditions, achieves error reduction of 17.5% (78.2 – 82.01). **(2)** Initial conditions: Error reductions in the range: 11.5% – 37.8% (82.01 – 84.08 for word model 1, and 81.53 – 88.5 for morhpeme model 2-) were achieved by initializing the various models with context-free approximations. While this observation confirms Elworthy (1994), the impact of error reduction is much less than reported there for English - about 70% (79 – 94). The key difference (beside the unclear characteristic of Elworthy initial condition - since he made use of an annotated corpus) is the much higher quality of the uniform distribution for Hebrew. **(3)** Model order: The partial second-order HMM [2-] produced the best results for both word (85.75%) and morpheme (88.5%) models over the initial condition. The full second-order HMM [2] didn't upgrade the accuracy of the partial second-order, but achieved the best results for the uniform distribution morpheme model. This is because the context-free approximation does not take into account the tag of the previous word, which is part of model 2. We believe that initializing the morpheme model over a small set of annotated corpus will set much stronger initial condition for this model. **(4)** Model type: The main result of this paper is the error reduction of the morpheme model with respect to the word model: about 19.3% (85.75 – 88.5).

In addition, we apply the above models for the simpler task of segmentation and PoS tagging, as reported in Table 4. The task requires picking the correct morphemes of each word with their correct PoS (excluding all other morphological features). The best result for this task is obtained with the morpheme model 2: 92.32%. For this simpler task, the improvement brought by the morpheme model over the word model is less significant, but still consists of a 5% error reduction.

Unknown words account for a significant chunk of the errors. Table 5 shows the distribution of errors contributed by unknown words (words that cannot be analyzed by the morphological analyzer). 7.5% of the words in the test corpus are unknown: 4% are not recognized at all by the morphological analyzer (marked as [None] in the table), and for 3.5%, the set of analyses proposed by the analyzer does not contain the correct analysis [Missing]. We extended the lexicon to include *missing* and *none* lexemes of the closed sets. In addition, we modified the analyzer to extract all possible segmentations of unknown words, with all the possible tags for the segmented affixes, where the remaining unknown baseforms are tagged as UK. The model was trained over this set. In the next phase, the corpus was automatically tagged, according to the trained model, in order to form a tag distribution for each unknown word, according to its context and its form. Finally, the tag for each unknown word were selected according to its tag distribution. This strategy accounts for about half of the 7.5% unknown words.

	Order	Uniform	CF
W	1	91.07	91.47
W	2-	90.45	91.93
W	2	90.21	91.84
M	1	89.23	91.42
M	2-	89.77	91.76
M	2	91.42	**92.32**

Table 4: Segmentation and PoS Tagging

	None	Missing	%
Proper name	26	36	62
Closed Set	8	5.6	13.6
Other	16.5	5.4	21.9
Junk	2.5	0	2.5
	53	47	100

Table 5: Unknown Word Distribution

Table 6 shows the confusion matrix for known words (5% and up). The key confusions can be attributed to linguistic properties of Modern Hebrew: most Hebrew proper names are also nouns (and they are not marked by capitalization) – which explains the PN/N confusion. The verb/noun and verb/adjective confusions are explained by the nature of the participle form in Hebrew (beinoni) – participles behave syntactically almost in an identical manner as nouns.

Correct	Error	%
proper name	noun	17.9
noun	verb	15.3
noun	proper name	6.6
verb	noun	6.3
adjective	noun	5.4
adjective	verb	5.0

Table 6: Confusion Matrix for Known Words

5 Conclusions and Future Work

In this work, we have introduced a new text encoding method that captures rules of word formation in a language with affixational morphology such as Hebrew. This text encoding method allows us to learn in parallel segmentation and tagging rules in an unsupervised manner, despite the high ambiguity level of the morphological data (average number of 2.4 analyses per word). Reported results on a large scale corpus (6M words) with fully unsupervised learning are 92.32% for PoS tagging and 88.5% for full morphological disambiguation.

In this work, we used the backoff smoothing method, suggested by Thede and Harper (1999), with an extension of additive smoothing (Chen, 1996, 2.2.1) for the lexical probabilities (B and B2 matrices). To complete this study, we are currently investigating several smoothing techniques (Chen, 1996), in order to check whether the morpheme model is critical for the data sparseness problem, or whether it can be handled with smoothing over a word model.

We are currently investigating two major methods to improve our results: first, we have started gathering a larger corpus of manually tagged text and plan to perform semi-supervised learning on a corpus of 100K manually tagged words. Second, we plan to improve the unknown word model, such as integrating it with named entity recognition system (Ben-Mordechai, 2005).

References

Emmanuel Allon. 1995. *Unvocalized Hebrew Writing*. Ben Gurion University Press. (in Hebrew).

Roy Bar-Haim, Khalil Sima'an, and Yoad Winter. 2005. Choosing an optimal architecture for segmentation and pos-tagging of modern Hebrew. In *Proceedings of ACL-05 Workshop on Computational Approaches to Semitic Languages.*

Leonard E. Baum. 1972. An inequality and associated maximization technique in statistical estimation for probabilistic functions of a Markov process. *Inequalities*, 3:1–8.

Na'ama Ben-Mordechai. 2005. Named entities recognition in Hebrew. Master's thesis, Ben Gurion University of the Negev, Beer Sheva, Israel. (in Hebrew).

Eric Brill. 1995. Transformation-based error-driven learning and natural languge processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21:543–565.

David Carmel and Yoelle S. Maarek. 1999. Morphological disambiguation for Hebrew search systems. In *Proceeding of NGITS-99.*

Stanley F. Chen. 1996. *Building Probabilistic Models for Natural Language.* Ph.D. thesis, Harvard University, Cambridge, MA.

Mona Diab, Kadri Hacioglu, and Daniel Jurafsky. 2004. Automatic tagging of Arabic text: From raw text to base phrase chunks. In *Proceeding of HLT-NAACL-04.*

Kevin Duh and Katrin Kirchhoff. 2005. Pos tagging of dialectal Arabic: A minimally supervised approach. In *Proceedings of ACL-05 Workshop on Computational Approaches to Semitic Languages.*

Michael Elhadad, Yael Netzer, David Gabay, and Meni Adler. 2005. Hebrew morphological tagging guidelines. Technical report, Ben Gurion University, Dept. of Computer Science.

David Elworthy. 1994. Does Baum-Welch re-estimation help taggers? In *Proceeding of ANLP-94.*

Nizar Habash and Owen Rambow. 2005. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proceeding of ACL-05.*

Nadav Har'el and Dan Kenigsberg. 2004. HSpell - the free Hebrew spell checker and morphological analyzer. *Israeli Seminar on Computational Linguistics, December 2004.*

Daniel Jurafsky and James H. Martin. 2000. *Speech and language processing.* Prentice-Hall.

Gennady Lembersky. 2001. Named entities recognition; compounds: approaches and recognitions methods. Master's thesis, Ben Gurion University of the Negev, Beer Sheva, Israel. (in Hebrew).

Moshe Levinger, Uzi Ornan, and Alon Itai. 1995. Learning morpholexical probabilities from an untagged corpus with an application to Hebrew. *Computational Linguistics*, 21:383–404.

Moshe Levinger. 1992. Morhphological disambiguation in hebrew. Master's thesis, Technion, Haifa, Israel. (in Hebrew).

Christopher D. Manning and Hinrich Schutze. 1999. *Foundation of Statistical Language Processing.* MIT Press.

Bernard Merialdo. 1994. Tagging English text with probabilistic model. *Computatinal Linguistics*, 20:155–171.

Uzi Ornan. 2002. Hebrew in latin script. *Lĕšonénu*, LXIV:137–151. (in Hebrew).

Erel Segal. 2000. Hebrew morphological analyzer for Hebrew undotted texts. Master's thesis, Technion, Haifa, Israel. (in Hebrew).

Scott M. Thede and Mary P. Harper. 1999. A second-order hidden Markov model for part-of-speech tagging. In *Proceeding of ACL-99.*

Shlomo Yona and Shuly Wintner. 2005. A finite-state morphological grammar of Hebrew. In *Proceedings of ACL-05 Workshop on Computational Approaches to Semitic Languages.*