A Single Generative Model for Joint Morphological Segmentation and Syntactic Parsing

Yoav Goldberg
Ben Gurion University of the Negev
Department of Computer Science
POB 653 Be'er Sheva, 84105, Israel
yoavg@cs.bgu.ac.il

Reut Tsarfaty
Institute for Logic Language and Computation
University of Amsterdam
Plantage Muidergracht 24, Amsterdam, NL
rtsarfat@science.uva.nl

Abstract

Morphological processes in Semitic languages deliver space-delimited words which introduce multiple, distinct, syntactic units into the structure of the input sentence. These words are in turn highly ambiguous, breaking the assumption underlying most parsers that the yield of a tree for a given sentence is known in advance. Here we propose a single joint model for performing both morphological segmentation and syntactic disambiguation which bypasses the associated circularity. Using a treebank grammar, a data-driven lexicon, and a linguistically motivated unknown-tokens handling technique our model outperforms previous pipelined, integrated or factorized systems for Hebrew morphological and syntactic processing, yielding an error reduction of 12% over the best published results so far.

1 Introduction

Current state-of-the-art broad-coverage parsers assume a direct correspondence between the lexical items ingrained in the proposed syntactic analyses (the *yields* of syntactic *parse-trees*) and the space-delimited tokens (henceforth, 'tokens') that constitute the unanalyzed surface forms (*utterances*). In Semitic languages the situation is very different.

In Modern Hebrew (Hebrew), a Semitic language with very rich morphology, particles marking conjunctions, prepositions, complementizers and relativizers are bound elements prefixed to the word (Glinert, 1989). The Hebrew token '*bcl*'[1], for example, stands for the complete prepositional phrase "in the shadow". This token may further embed into a larger utterance, e.g., '*bcl hneim*' (literally "in-the-shadow the-pleasant", meaning roughly "in the pleasant shadow") in which the dominated Noun is modified by a proceeding space-delimited adjective. It should be clear from the onset that the particle *b* ("in") in '*bcl*' may then attach higher than the bare noun *cl* ("shadow"). This leads to word- and constituent-boundaries discrepancy, which breaks the assumptions underlying current state-of-the-art statistical parsers.

One way to approach this discrepancy is to assume a preceding phase of *morphological segmentation* for extracting the different lexical items that exist at the token level (as is done, to the best of our knowledge, in all parsing related work on Arabic and its dialects (Chiang et al., 2006)). The input for the segmentation task is however highly ambiguous for Semitic languages, and surface forms (tokens) may admit multiple possible analyses as in (Bar-Haim et al., 2007; Adler and Elhadad, 2006). The aforementioned surface form *bcl*, for example, may also stand for the lexical item "onion", a Noun. The implication of this ambiguity for a parser is that the yield of syntactic trees no longer consists of space-delimited tokens, and the expected number of leaves in the syntactic analysis in not known in advance.

Tsarfaty (2006) argues that for Semitic languages determining the correct morphological segmentation is dependent on syntactic context and shows that increasing information sharing between the morphological and the syntactic components leads to improved performance on the joint task. Cohen and Smith (2007) followed up on these results and pro-

[1] We adopt here the transliteration of (Sima'an et al., 2001).

Proceedings of ACL-08: HLT, pages 371–379,
Columbus, Ohio, USA, June 2008.

posed a system for joint inference of morphological and syntactic structures using factored models each designed and trained on its own.

Here we push the single-framework conjecture across the board and present a single model that performs morphological segmentation and syntactic disambiguation in a fully generative framework. We claim that no particular morphological segmentation is a-priory more likely for surface forms before exploring the compositional nature of syntactic structures, including manifestations of various long-distance dependencies. Morphological segmentation decisions in our model are delegated to a *lexeme-based* PCFG and we show that using a simple treebank grammar, a data-driven lexicon, and a linguistically motivated unknown-tokens handling our model outperforms (Tsarfaty, 2006) and (Cohen and Smith, 2007) on the joint task and achieves state-of-the-art results on a par with current respective standalone models.[2]

2 Modern Hebrew Structure

Segmental morphology Hebrew consists of seven particles *m*("from") *f*("when"/"who"/"that") *h*("the") *w*("and") *k*("like") *l*("to") and *b*("in"). which may never appear in isolation and must always attach as prefixes to the following open-class category item we refer to as *stem*. Several such particles may be prefixed onto a single stem, in which case the affixation is subject to strict linear precedence constraints. Co-occurrences among the particles themselves are subject to further syntactic and lexical constraints relative to the stem.

While the linear precedence of segmental morphemes within a token is subject to constraints, the dominance relations among their mother and sister constituents is rather free. The relativizer *f*("that") for example, may attach to an arbitrarily long relative clause that goes beyond token boundaries. The attachment in such cases encompasses a long distance dependency that cannot be captured by Markovian processes that are typically used for morphological disambiguation. The same argument holds for resolving PP attachment of a prefixed preposition or marking conjunction of elements of any kind.

A less canonical representation of segmental morphology is triggered by a morpho-phonological process of omitting the definite article *h* when occurring after the particles *b* or *l*. This process triggers ambiguity as for the definiteness status of Nouns following these particles.We refer to such cases in which the concatenation of elements does not strictly correspond to the original surface form as *super-segmental morphology*. An additional case of super-segmental morphology is the case of Pronominal Clitics. Inflectional features marking pronominal elements may be attached to different kinds of categories marking their pronominal complements. The additional morphological material in such cases appears *after* the stem and realizes the extended meaning. The current work treats both segmental and super-segmental phenomena, yet we note that there may be more adequate ways to treat super-segmental phenomena assuming Word-Based morphology as we explore in (Tsarfaty and Goldberg, 2008).

Lexical and Morphological Ambiguity The rich morphological processes for deriving Hebrew stems give rise to a high degree of ambiguity for Hebrew space-delimited tokens. The form *fmnh*, for example, can be understood as the verb "lubricated", the possessed noun "her oil", the adjective "fat" or the verb "got fat". Furthermore, the systematic way in which particles are prefixed to one another and onto an open-class category gives rise to a distinct sort of morphological ambiguity: space-delimited tokens may be ambiguous between several different segmentation possibilities. The same form *fmnh* can be segmented as *f-mnh*, *f* ("that") functioning as a relativizer with the form *mnh*. The form *mnh* itself can be read as at least three different verbs ("counted", "appointed", "was appointed"), a noun ("a portion"), and a possessed noun ("her kind").

Such ambiguities cause discrepancies between token boundaries (indexed as white spaces) and constituent boundaries (imposed by syntactic categories) with respect to a surface form. Such discrepancies can be aligned via an intermediate level of PoS tags. PoS tags impose a unique morphological segmentation on surface tokens and present a unique valid yield for syntactic trees. The correct ambiguity resolution of the syntactic level therefore helps to resolve the morphological one, and vice versa.

[2] Standalone parsing models assume a segmentation Oracle.

3 Previous Work on Hebrew Processing

Morphological analyzers for Hebrew that analyze a surface form in isolation have been proposed by Segal (2000), Yona and Wintner (2005), and recently by the knowledge center for processing Hebrew (Itai et al., 2006). Such analyzers propose multiple segmentation possibilities and their corresponding analyses for a token in isolation but have no means to determine the most likely ones. Morphological disambiguators that consider a token in context (an utterance) and propose the most likely morphological analysis of an utterance (including segmentation) were presented by Bar-Haim *et al.* (2005), Adler and Elhadad (2006), Shacham and Wintner (2007), and achieved good results (the best segmentation result so far is around 98%).

The development of the very first Hebrew Treebank (Sima'an et al., 2001) called for the exploration of general statistical parsing methods, but the application was at first limited. Sima'an *et al.* (2001) presented parsing results for a DOP tree-gram model using a small data set (500 sentences) and semi-automatic morphological disambiguation. Tsarfaty (2006) was the first to demonstrate that fully automatic Hebrew parsing is feasible using the newly available 5000 sentences treebank. Tsarfaty and Sima'an (2007) have reported state-of-the-art results on Hebrew unlexicalized parsing (74.41%) albeit assuming oracle morphological segmentation.

The joint morphological and syntactic hypothesis was first discussed in (Tsarfaty, 2006; Tsarfaty and Sima'an, 2004) and empirically explored in (Tsarfaty, 2006). Tsarfaty (2006) used a morphological analyzer (Segal, 2000), a PoS tagger (Bar-Haim et al., 2005), and a general purpose parser (Schmid, 2000) in an integrated framework in which morphological and syntactic components interact to share information, leading to improved performance on the joint task. Cohen and Smith (2007) later on based a system for joint inference on factored, independent, morphological and syntactic components of which scores are combined to cater for the joint inference task. Both (Tsarfaty, 2006; Cohen and Smith, 2007) have shown that a single integrated framework outperforms a completely streamlined implementation, yet neither has shown a single generative model which handles both tasks.

4 Model Preliminaries

4.1 The Status Space-Delimited Tokens

A Hebrew surface token may have several readings, each of which corresponding to a sequence of segments and their corresponding PoS tags. We refer to different readings as different *analyses* whereby the segments are deterministic given the sequence of PoS tags. We refer to a segment and its assigned PoS tag as a *lexeme*, and so analyses are in fact sequences of lexemes. For brevity we omit the segments from the analysis, and so analysis of the form "*fmnh*" as *f*/REL *mnh*/VB is represented simply as REL VB.

Such tag sequences are often treated as "complex tags" (*e.g.* REL+VB) (cf. (Bar-Haim et al., 2007; Habash and Rambow, 2005)) and probabilities are assigned to different analyses in accordance with the likelihood of their tags (e.g., "*fmnh* is 30% likely to be tagged NN and 70% likely to be tagged REL+VB"). Here we do not submit to this view. When a token *fmnh* is to be interpreted as the lexeme sequence *f*/REL *mnh*/VB, the analysis introduces two distinct entities, the relativizer *f* ("that") and the verb *mnh* ("counted"), and not as the complex entity "that counted". When the same token is to be interpreted as a single lexeme *fmnh*, it may function as a single adjective "fat". There is no relation between these two interpretations other then the fact that their surface forms coincide, and we argue that the only reason to prefer one analysis over the other is compositional. A possible probabilistic model for assigning probabilities to complex analyses of a surface form may be

$$P(\mathsf{REL},\mathsf{VB}|\textit{fmnh},\text{context}) =$$
$$P(\mathsf{REL}|\textit{f})P(\mathsf{VB}|\textit{mnh},\mathsf{REL})P(\mathsf{REL},\mathsf{VB}|\text{ context})$$

and indeed recent sequential disambiguation models for Hebrew (Adler and Elhadad, 2006) and Arabic (Smith et al., 2005) present similar models.

We suggest that in unlexicalized PCFGs the syntactic context may be explicitly modeled in the derivation probabilities. Hence, we take the probability of the event *fmnh* analyzed as REL VB to be

$$P(\mathsf{REL} \rightarrow \textit{f}|\mathsf{REL}) \times P(\mathsf{VB} \rightarrow \textit{mnh}|\mathsf{VB})$$

This means that we generate *f* and *mnh* independently depending on their corresponding PoS tags,

and the context (as well as the syntactic relation between the two) is modeled via the derivation resulting in a sequence REL VB spanning the form *fmnh*.

4.2 Lattice Representation

We represent all morphological analyses of a given utterance using a lattice structure. Each lattice arc corresponds to a segment and its corresponding PoS tag, and a path through the lattice corresponds to a specific morphological segmentation of the utterance. This is by now a fairly standard representation for multiple morphological segmentation of Hebrew utterances (Adler, 2001; Bar-Haim et al., 2005; Smith et al., 2005; Cohen and Smith, 2007; Adler, 2007). Figure 1 depicts the lattice for a 2-words sentence *bclm hneim*. We use double-circles to indicate the space-delimited token boundaries. Note that in our construction arcs can never cross token boundaries. Every token is independent of the others, and the sentence lattice is in fact a concatenation of smaller lattices, one for each token. Furthermore, some of the arcs represent lexemes not present in the input tokens (*e.g.* *h*/DT, *fl*/POS), however these are parts of valid analyses of the token (cf. *super-segmental morphology* section 2). Segments with the same surface form but different PoS tags are treated as different lexemes, and are represented as separate arcs (*e.g.* the two arcs labeled *neim* from node 6 to 7).

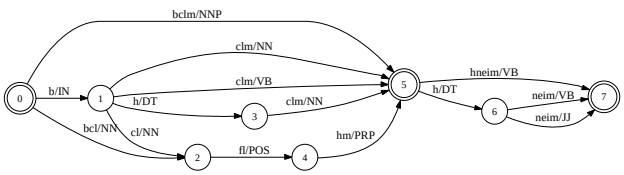

Figure 1: The Lattice for the Hebrew Phrase *bclm hneim*

A similar structure is used in speech recognition. There, a lattice is used to represent the possible sentences resulting from an interpretation of an acoustic model. In speech recognition the arcs of the lattice are typically weighted in order to indicate the probability of specific transitions. Given that weights on all outgoing arcs sum up to one, weights induce a probability distribution on the lattice paths. In sequential tagging models such as (Adler and Elhadad, 2006; Bar-Haim et al., 2007; Smith et al., 2005) weights are assigned according to a language model based on linear context. In our model, however, all lattice paths are taken to be a-priori equally likely.

5 A Generative PCFG Model

The input for the joint task is a sequence $W = w_1, \ldots, w_n$ of space-delimited tokens. Each token may admit multiple analyses, each of which a sequence of one or more lexemes (we use l_i to denote a lexeme) belonging a presupposed Hebrew lexicon LEX. The entries in such a lexicon may be thought of as meaningful surface segments paired up with their PoS tags $l_i = \langle s_i, p_i \rangle$, but note that a surface segment s need not be a space-delimited token.

The Input The set of analyses for a token is thus represented as a lattice in which every arc corresponds to a specific lexeme l, as shown in Figure 1. A morphological analyzer $M : \mathcal{W} \rightarrow \mathcal{L}$ is a function mapping sentences in Hebrew ($W \in \mathcal{W}$) to their corresponding lattices ($M(W) = L \in \mathcal{L}$). We define the lattice L to be the concatenation of the lattices L_i corresponding to the input words w_i (s.t. $M(w_i) = L_i$). Each connected path $\langle l_1 \ldots l_k \rangle \in L$ corresponds to one morphological segmentation possibility of W.

The Parser Given a sequence of input tokens $W = w_1 \ldots w_n$ and a morphological analyzer, we look for the most probable parse tree π s.t.

$$\hat{\pi} = \arg\max_{\pi} P(\pi | W, M)$$

Since the lattice L for a given sentence W is determined by the morphological analyzer M we have

$$\hat{\pi} = \arg\max_{\pi} P(\pi | W, M, L)$$

Hence, our parser searches for a parse tree π over lexemes $\langle l_1 \ldots l_k \rangle$ s.t. $l_i = \langle s_i, p_i \rangle \in LEX$, $\langle l_1 \ldots l_k \rangle \in L$ and $M(W) = L$. So we remain with

$$\hat{\pi} = \arg\max_{\pi} P(\pi | L)$$

which is precisely the formula corresponding to the so-called lattice parsing familiar from speech recognition. Every parse π selects a specific morphological segmentation $\langle l_1 ... l_k \rangle$ (a path through the lattice). This is akin to PoS tags sequences induced by different parses in the setup familiar from English and explored in e.g. (Charniak et al., 1996).

Our use of an unweighted lattice reflects our belief that all the segmentations of the given input sentence are a-priori equally likely; the only reason to prefer one segmentation over the another is due to the overall syntactic context which is modeled via the PCFG derivations. A compatible view is presented by Charniak et al. (1996) who consider the kind of probabilities a generative parser should get from a PoS tagger, and concludes that these should be $P(w|t)$ "and nothing fancier".[3] In our setting, therefore, the Lattice is *not* used to induce a probability distribution on a linear context, but rather, it is used as a common-denominator of state-indexation of all segmentations possibilities of a surface form. This is a unique object for which we are able to define a proper probability model. Thus our proposed model is a proper model assigning probability mass to all $\langle \pi, L \rangle$ pairs, where π is a parse tree and L is the one and only lattice that a sequence of characters (and spaces) W over our alpha-beth gives rise to.

$$\sum_{\pi, L} P(\pi, L) = 1; \text{L uniquely index } W$$

The Grammar Our parser looks for the most likely tree spanning a single path through the lattice of which the yield is a sequence of lexemes. This is done using a simple PCFG which is *lexeme-based*. This means that the rules in our grammar are of two kinds: (a) syntactic rules relating non-terminals to a sequence of non-terminals and/or PoS tags, and (b) lexical rules relating PoS tags to lattice arcs (lexemes). The possible analyses of a surface token pose constraints on the analyses of specific segments. In order to pass these constraints onto the parser, the lexical rules in the grammar are of the form $p_i \rightarrow \langle s_i, p_i \rangle$

Parameter Estimation The grammar probabilities are estimated from the corpus using simple relative frequency estimates. Lexical rules are estimated in a similar manner. We smooth $P_{rf}(p \rightarrow \langle s, p \rangle)$ for rare and OOV segments ($s \in l, l \in L, s$ unseen) using a "per-tag" probability distribution over rare segments which we estimate using relative frequency estimates for once-occurring segments.

[3] An English sentence with ambiguous PoS assignment can be trivially represented as a lattice similar to our own, where every pair of consecutive nodes correspond to a word, and every possible PoS assignment for this word is a connecting arc.

Handling Unknown tokens When handling unknown *tokens* in a language such as Hebrew various important aspects have to be borne in mind. Firstly, Hebrew unknown tokens are doubly unknown: each unknown token may correspond to several segmentation possibilities, and each segment in such sequences may be able to admit multiple PoS tags. Secondly, some segments in a proposed segment sequence may in fact be *seen* lexical events, i.e., for some p tag $P_{rf}(p \rightarrow \langle s, p \rangle) > 0$, while other segments have never been observed as a lexical event before. The latter arcs correspond to OOV words in English. Finally, the assignments of PoS tags to OOV segments is subject to language specific constraints relative to the token it was originated from.

Our smoothing procedure takes into account all the aforementioned aspects and works as follows. We first make use of our morphological analyzer to find all segmentation possibilities by chopping off all prefix sequence possibilities (including the empty prefix) and construct a lattice off of them. The remaining arcs are marked OOV. At this stage the lattice path corresponds to segments only, with no PoS assigned to them. In turn we use two sorts of heuristics, orthogonal to one another, to prune segmentation possibilities based on *lexical* and *grammatical* constraints. We simulate *lexical* constraints by using an external lexical resource against which we verify whether OOV segments are in fact valid Hebrew lexemes. This heuristics is used to prune all segmentation possibilities involving "lexically improper" segments. For the remaining arcs, if the segment is in fact a known lexeme it is tagged as usual, but for the OOV arcs which are valid Hebrew entries lacking tags assignment, we assign all possible tags and then simulate a *grammatical* constraint. Here, all token-internal collocations of tags unseen in our training data are pruned away. From now on all lattice arcs are tagged segments and the assignment of probability $P(p \rightarrow \langle s, p \rangle)$ to lattice arcs proceeds as usual.[4]

A rather pathological case is when our lexical heuristics prune away all segmentation possibilities and we remain with an empty lattice. In such cases we use the non-pruned lattice including all (possibly ungrammatical) segmentation, and let the statistics (including OOV) decide. We empirically control for

[4] Our heuristics may slightly alter $\sum_{\pi, L} P(\pi, L) \approx 1$

the effect of our heuristics to make sure our pruning does not undermine the objectives of our joint task.

6 Experimental Setup

Previous work on morphological and syntactic disambiguation in Hebrew used different sets of data, different splits, differing annotation schemes, and different evaluation measures. Our experimental setup therefore is designed to serve two goals. Our primary goal is to exploit the resources that are most appropriate for the task at hand, and our secondary goal is to allow for comparison of our models' performance against previously reported results. When a comparison against previous results requires additional pre-processing, we state it explicitly to allow for the reader to replicate the reported results.

Data We use the Hebrew Treebank, (Sima'an et al., 2001), provided by the knowledge center for processing Hebrew, in which sentences from the daily newspaper "Ha'aretz" are morphologically segmented and syntactically annotated. The treebank has two versions, v1.0 and v2.0, containing 5001 and 6501 sentences respectively. We use v1.0 mainly because previous studies on joint inference reported results w.r.t. v1.0 only.[5] We expect that using the same setup on v2.0 will allow a cross-treebank comparison.[6] We used the first 500 sentences as our dev set and the rest 4500 for training and report our main results on this split. To facilitate the comparison of our results to those reported by (Cohen and Smith, 2007) we use their data set in which 177 empty and "malformed"[7] were removed. The first 3770 trees of the resulting set then were used for training, and the last 418 are used testing. (we ignored the 419 trees in their development set.)

Morphological Analyzer Ideally, we would use an of-the-shelf morphological analyzer for mapping each input token to its possible analyses. Such resources exist for Hebrew (Itai et al., 2006), but unfortunately use a tagging scheme which is incompatible with the one of the Hebrew Treebank.[8] For this reason, we use a data-driven morphological analyzer derived from the training data similar to (Cohen and Smith, 2007). We construct a mapping from all the space-delimited tokens seen in the training sentences to their corresponding analyses.

Lexicon and OOV Handling Our data-driven morphological-analyzer proposes analyses for unknown tokens as described in Section 5. We use the HSPELL[9] (Har'el and Kenigsberg, 2004) wordlist as a lexeme-based lexicon for pruning segmentations involving invalid segments. Models that employ this strategy are denoted $\mathbf{hsp}$. To control for the effect of the HSPELL-based pruning, we also experimented with a morphological analyzer that does not perform this pruning. For these models we limit the options provided for OOV words by not considering the entire token as a valid segmentation in case at least some prefix segmentation exists. This analyzer setting is similar to that of (Cohen and Smith, 2007), and models using it are denoted $\mathbf{nohsp}$,

Parser and Grammar We used BitPar (Schmid, 2004), an efficient general purpose parser,[10] together with various treebank grammars to parse the input sentences and propose compatible morphological segmentation and syntactic analysis.

We experimented with increasingly rich grammars read off of the treebank. Our first model is $\mathbf{GT_{plain}}$, a PCFG learned from the treebank after removing all functional features from the syntactic categories. In our second model $\mathbf{GT_{vpi}}$ we also distinguished finite and non-finite verbs and VPs as

[5] The comparison to performance on version 2.0 is meaningless not only because of the change in size, but also conceptual changes in the annotation scheme

[6] Unfortunatley running our setup on the v2.0 data set is currently not possible due to missing tokens-morphemes alignment in the v2.0 treebank.

[7] We thank Shay Cohen for providing us with their data set and evaluation Software.

[8] Mapping between the two schemes involves non-deterministic many-to-many mappings, and in some cases require a change in the syntactic trees.

[9] An open-source Hebrew spell-checker.

[10] Lattice parsing can be performed by special initialization of the chart in a CKY parser (Chappelier et al., 1999). We currently simulate this by crafting a WCFG and feeding it to BitPar. Given a PCFG grammar G and a lattice L with nodes $n_1 \ldots n_k$, we construct the weighted grammar G_L as follows: for every arc (lexeme) $l \in L$ from node n_i to node n_j, we add to G_L the rule $[l \rightarrow t_{n_i}, t_{n_{i+1}}, \ldots, t_{n_{j-1}}]$ with a probability of 1 (this indicates the lexeme l spans from node n_i to node n_j). G_L is then used to parse the string $t_{n_1} \ldots t_{n_{k-1}}$, where t_{n_i} is a terminal corresponding to the lattice span between node n_i and n_{i+1}. Removing the leaves from the resulting tree yields a parse for L under G, with the desired probabilities. We use a patched version of BitPar allowing for direct input of probabilities instead of counts. We thank Felix Hageloh (Hageloh, 2006) for providing us with this version.

proposed in (Tsarfaty, 2006). In our third model $\mathbf{GT_{ppp}}$ we also add the distinction between general PPs and possessive PPs following Goldberg and Elhadad (2007). In our forth model $\mathbf{GT_{nph}}$ we add the definiteness status of constituents following Tsarfaty and Sima'an (2007). Finally, model $\mathbf{GT_v = 2}$ includes parent annotation on top of the various state-splits, as is done also in (Tsarfaty and Sima'an, 2007; Cohen and Smith, 2007). For all grammars, we use fine-grained PoS tags indicating various morphological features annotated therein.

Evaluation We use 8 different measures to evaluate the performance of our system on the joint disambiguation task. To evaluate the performance on the segmentation task, we report SEG, the standard harmonic means for segmentation Precision and Recall F_1 (as defined in Bar-Haim *et al.* (2005); Tsarfaty (2006)) as well as the segmentation accuracy SEG_{Tok} measure indicating the percentage of input tokens assigned the correct exact segmentation (as reported by Cohen and Smith (2007)). $SEG_{Tok}(noH)$ is the segmentation accuracy ignoring mistakes involving the implicit definite article *h*.[11] To evaluate our performance on the tagging task we report $CPOS$ and $FPOS$ corresponding to coarse- and fine-grained PoS tagging results (F_1) measure. Evaluating parsing results in our joint framework, as argued by Tsarfaty (2006), is not trivial under the joint disambiguation task, as the hypothesized yield need not coincide with the correct one. Our parsing performance measures (SYN) thus report the PARSEVAL extension proposed in Tsarfaty (2006). We further report SYN^{CS}, the parsing metric of Cohen and Smith (2007), to facilitate the comparison. We report the F_1 value of both measures. Finally, our U (unparsed) measure is used to report the number of sentences to which our system could not propose a joint analysis.

7 Results and Analysis

The accuracy results for segmentation, tagging and parsing using our different models and our standard data split are summarized in Table 1. In addition we report for each model its performance on gold-segmented input (GS) to indicate the upper bound for the grammars' performance on the parsing task.

The table makes clear that enriching our grammar improves the syntactic performance as well as morphological disambiguation (segmentation and POS tagging) accuracy. This supports our main thesis that decisions taken by single, improved, grammar are beneficial for both tasks. When using the segmentation pruning (using HSPELL) for unseen tokens, performance improves for all tasks as well. Yet we note that the better grammars without pruning outperform the poorer grammars using this technique, indicating that the syntactic context aids, to some extent, the disambiguation of unknown tokens.

Table 2 compares the performance of our system on the setup of Cohen and Smith (2007) to the best results reported by them for the same tasks.

Model	SEG_{Tok}	$CPOS$	$FPOS$	SYN^{CS}
$\mathbf{GT_{nohsp/pln}}$	89.50	81.00	77.65	62.22
$\mathbf{GT_{nohsp/\cdots+nph}}$	89.58	81.26	77.82	64.30
CS_{pln}	*91.10*	*80.40*	*75.60*	*64.00*
$CS_{v=2}$	*90.90*	*80.50*	*75.40*	*64.40*
$\mathbf{GT_{hsp/pln}}$	93.13	83.12	79.12	64.46
$\mathbf{GT_{nohsp/\cdots+v=2}}$	89.66	82.85	78.92	66.31
Oracle CS_{pln}	*91.80*	*83.20*	*79.10*	*66.50*
Oracle $CS_{v=2}$	*91.70*	*83.00*	*78.70*	*67.40*
$\mathbf{GT_{hsp/\cdots+v=2}}$	93.38	85.08	80.11	69.11

Table 2: Segmentation, Parsing and Tagging Results using the Setup of (Cohen and Smith, 2007) (sentence length ≤ 40). The Models' are Ordered by Performance.

We first note that the accuracy results of our system are overall higher on their setup, on all measures, indicating that theirs may be an easier dataset. Secondly, for all our models we provide better fine- and coarse-grained POS-tagging accuracy, and all pruned models outperform the Oracle results reported by them.[12] In terms of syntactic disambiguation, even the simplest grammar pruned with HSPELL outperforms their non-Oracle results. Without HSPELL-pruning, our simpler grammars are somewhat lagging behind, but as the grammars improve the gap is bridged. The addition of vertical markovization enables non-pruned models to outperform all previously reported re-

[11] Overt definiteness errors may be seen as a wrong feature rather than as wrong constituent and it is by now an accepted standard to report accuracy with and without such errors.

[12] Cohen and Smith (2007) make use of a parameter (α) which is tuned separately for each of the tasks. This essentially means that their model does not result in a true joint inference, as executions for different tasks involve tuning a parameter separately. In our model there are no such hyper-parameters, and the performance is the result of truly joint disambiguation.

Model	U	SEG_{Tok} / no H	SEG_F	$CPOS$	$FPOS$	SYN / SYN^{CS}	$GS\ SYN$
$\mathbf{GT_{nohsp/pln}}$	7	89.77 / 93.18	91.80	80.36	76.77	60.41 / 61.66	65.00
....+vpi	7	89.80 / 93.18	91.84	80.37	76.74	61.16 / 62.41	66.70
....+ppp	7	89.79 / 93.20	91.86	80.43	76.79	61.47 / 62.86	67.22
....+nph	7	89.78 / 93.20	91.86	80.43	76.87	61.85 / 63.06	68.23
....+v=2	9	89.12 / 92.45	91.77	82.02	77.86	64.53 / 66.02	70.82
$\mathbf{GT_{hsp/pln}}$	11	92.00 / 94.81	94.52	82.35	78.11	62.10 / 64.17	65.00
....+vpi	11	92.03 / 94.82	94.58	82.39	78.23	63.00 / 65.06	66.70
....+ppp	11	92.02 / 94.85	94.58	82.48	78.33	63.26 / 65.42	67.22
....+nph	11	92.14 / 94.91	94.73	82.58	78.47	63.98 / 65.98	68.23
....+v=2	13	91.42 / 94.10	94.67	84.23	79.25	66.60 / 68.79	70.82

Table 1: Segmentation, tagging and parsing results on the Standard dev/train Split, for all Sentences

sults. Furthermore, the combination of pruning and vertical markovization of the grammar outperforms the Oracle results reported by Cohen and Smith. This essentially means that a better grammar tunes the joint model for optimized syntactic disambiguation at least in as much as their hyper parameters do. An interesting observation is that while vertical markovization benefits all our models, its effect is less evident in Cohen and Smith.

On the surface, our model may seem as a special case of Cohen and Smith in which $\alpha = 0$. However, there is a crucial difference: the morphological probabilities in their model come from discriminative models based on linear context. Many morphological decisions are based on long distance dependencies, and when the global syntactic evidence disagrees with evidence based on local linear context, the two models compete with one another, despite the fact that the PCFG takes also local context into account. In addition, as the CRF and PCFG look at similar sorts of information from within two inherently different models, they are far from independent and optimizing their product is meaningless. Cohen and Smith approach this by introducing the α hyperparameter, which performs best when optimized independently for each sentence (cf. Oracle results).

In contrast, our morphological probabilities are based on a *unigram*, *lexeme*-based model, and all other (local and non-local) contextual considerations are delegated to the PCFG. This fully generative model caters for real interaction between the syntactic and morphological levels as a part of a single coherent process.

8 Discussion and Conclusion

Employing a PCFG-based generative framework to make both syntactic and morphological disambiguation decisions is not only theoretically clean and linguistically justified and but also probabilistically apropriate and empirically sound. The overall performance of our joint framework demonstrates that a probability distribution obtained over mere syntactic contexts using a Treebank grammar and a data-driven lexicon outperforms upper bounds proposed by previous joint disambiguation systems and achieves segmentation and parsing results on a par with state-of-the-art standalone applications results.

Better grammars are shown here to improve performance on both morphological and syntactic tasks, providing support for the advantage of a joint framework over pipelined or factorized ones. We conjecture that this trend may continue by incorporating additional information, e.g., three-dimensional models as proposed by Tsarfaty and Sima'an (2007). In the current work morphological analyses and lexical probabilities are derived from a small Treebank, which is by no means the best way to go. Using a wide-coverage morphological analyzer based on (Itai et al., 2006) should cater for a better coverage, and incorporating lexical probabilities learned from a big (unannotated) corpus (cf. (Levinger et al., 1995; Goldberg et al., ; Adler et al., 2008)) will make the parser more robust and suitable for use in more realistic scenarios.

Acknowledgments We thank Meni Adler and Michael Elhadad (BGU) for helpful comments and discussion. We further thank Khalil Simaan (ILLC-UvA) for his careful advise concerning the formal details of the proposal. The work of the first author was supported by the Lynn and William Frankel Center for Computer Sciences. The work of the second author as well as collaboration visits to Israel was financed by NWO, grant number 017.001.271.

References

Meni Adler and Michael Elhadad. 2006. An Unsupervised Morpheme-Based HMM for Hebrew Morphological Disambiguation. In *Proceeding of COLING-ACL-06*, Sydney, Australia.

Meni Adler, Yoav Goldberg, David Gabay, and Michael Elhadad. 2008. Unsupervised Lexicon-Based Resolution of Unknown Words for Full Morpholological Analysis. In *Proceedings of ACL-08*.

Meni Adler. 2001. Hidden Markov Model for Hebrew Part-of-Speech Tagging. Master's thesis, Ben-Gurion University of the Negev.

Meni Adler. 2007. *Hebrew Morphological Disambiguation: An Unsupervised Stochastic Word-based Approach*. Ph.D. thesis, Ben-Gurion University of the Negev, Beer-Sheva, Israel.

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*.

Roy Bar-Haim, Khalil Sima'an, and Yoad Winter. 2007. Part-of-speech tagging of Modern Hebrew text. *Natural Language Engineering*, 14(02):223–251.

J. Chappelier, M. Rajman, R. Aragues, and A. Rozenknop. 1999. Lattice Parsing for Speech Recognition.

Eugene Charniak, Glenn Carroll, John Adcock, Anthony R. Cassandra, Yoshihiko Gotoh, Jeremy Katz, Michael L. Littman, and John McCann. 1996. Taggers for Parsers. *AI*, 85(1-2):45–57.

David Chiang, Mona Diab, Nizar Habash, Owen Rambow, and Safiullah Shareef. 2006. Parsing Arabic Dialects. In *Proceedings of EACL-06*.

Shay B. Cohen and Noah A. Smith. 2007. Joint morphological and syntactic disambiguation. In *Proceedings of EMNLP-CoNLL-07*, pages 208–217.

Lewis Glinert. 1989. *The Grammar of Modern Hebrew*. Cambridge University Press.

Yoav Goldberg and Michael Elhadad. 2007. SVM Model Tampering and Anchored Learning: A Case Study in Hebrew NP Chunking. In *Proceeding of ACL-07*, Prague, Czech Republic.

Yoav Goldberg, Meni Adler, and Michael Elhadad. EM Can Find Pretty G]ood HMM POS-Taggers (When Given a Good Start), booktitle = Proceedings of ACL-08, year = 2008,.

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

Felix Hageloh. 2006. Parsing Using Transforms over Treebanks. Master's thesis, University of Amsterdam.

Nadav Har'el and Dan Kenigsberg. 2004. HSpell - the free Hebrew Spell Checker and Morphological Analyzer. *Israeli Seminar on Computational Linguistics*.

Alon Itai, Shuly Wintner, and Shlomo Yona. 2006. A Computational Lexicon of Contemporary Hebrew. In *Proceedings of LREC-06*.

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.

Helmut Schmid, 2000. *LoPar: Design and Implementation*. Institute for Computational Linguistics, University of Stuttgart.

Helmut Schmid. 2004. Efficient Parsing of Highly Ambiguous Context-Free Grammars with Bit Vector. In *Proceedings of COLING-04*.

Erel Segal. 2000. Hebrew Morphological Analyzer for Hebrew Undotted Texts. Master's thesis, Technion, Haifa, Israel.

Danny Shacham and Shuly Wintner. 2007. Morphological Disambiguation of Hebrew: A Case Study in Classifier Combination. In *Proceedings of EMNLP-CoNLL-07*, pages 439–447.

Khalil Sima'an, Alon Itai, Yoad Winter, Alon Altman, and Noa Nativ. 2001. Building a Tree-Bank for Modern Hebrew Text. In *Traitement Automatique des Langues*, volume 42.

Noah A. Smith, David A. Smith, and Roy W. Tromble. 2005. Context-based morphological disambiguation with random fields. In *Proceedings of HLT-05*, pages 475–482, Morristown, NJ, USA. Association for Computational Linguistics.

Reut Tsarfaty and Yoav Goldberg. 2008. Word-Based or Morpheme-Based? Annotation Strategies for Modern Hebrew Clitics. In *Proceedings of LREC-08*.

Reut Tsarfaty and Khalil Sima'an. 2004. An Integrated Model for Morphological and Syntactic Disambiguation in Modern Hebrew. MOZAIEK detailed proposal, NWO Mozaiek scheme.

Reut Tsarfaty and Khalil Sima'an. 2007. Three-Dimensional Parametrization for Parsing Morphologically Rich Languages. In *Proceedings of IWPT-07*.

Reut Tsarfaty. 2006. Integrated Morphological and Syntactic Disambiguation for Modern Hebrew. In *Proceedings of ACL-SRW-06*.

Shlomo Yona and Shuly Wintner. 2005. A Finite-state Morphological Grammar of Hebrew. In *Proceedings of the ACL-05 Workshop on Computational Approaches to Semitic Languages*.