Morphological Features for Parsing Morphologically-rich Languages: A Case of Arabic

Jon Dehdari
Department of Linguistics
The Ohio State University
jonsafari@ling.osu.edu

Lamia Tounsi
NCLT, School of Computing
Dublin City University
ltounsi@computing.dcu.ie

Josef van Genabith
CNGL, School of Computing
Dublin City University
josef@computing.dcu.ie

Abstract

We investigate how morphological features in the form of part-of-speech tags impact parsing performance, using Arabic as our test case. The large, fine-grained tagset of the Penn Arabic Treebank (498 tags) is difficult to handle by parsers, ultimately due to data sparsity. However, *ad-hoc* conflations of treebank tags runs the risk of discarding potentially useful parsing information.

The main contribution of this paper is to describe several automated, language-independent methods that search for the optimal feature combination to help parsing. We first identify 15 individual features from the Penn Arabic Treebank tagset. Either including or excluding these features results in 32,768 combinations, so we then apply heuristic techniques to identify the combination achieving the highest parsing performance.

Our results show a statistically significant improvement of 2.86% for vocalized text and 1.88% for unvocalized text, compared with the baseline provided by the Bikel-Bies Arabic POS mapping (and an improvement of 2.14% using product models for vocalized text, 1.65% for unvocalized text), giving state-of-the-art results for Arabic constituency parsing.

1 Introduction

Parsing Arabic is challenging due to its morphological richness and syntactic complexity. In particular, the number of distinct word-forms, the relative freedom with respect to word order, and the information expressed at the level of words make parsing Arabic a difficult task. Previous research established that adapting constituency parsing models developed from English to Arabic (and other languages) is a non-trivial task. Significant effort has been deployed to parse Chinese using the unlexicalized parser of Klein and Manning (2003a,b) with modest performance gains over previous approaches. Due to the specification of head rules, lexicalized parsing models also turned out to be difficult to generalize to other languages: Kulick et al. (2006) describe Arabic parsing results far below English or even Chinese using the Collins parsing model as implemented in the Bikel parser (Bikel, 2004).

In order to side-step the surface representations involved in constituency parsing, several studies have focused on Arabic dependency parsing. The general assumption is that dependency structures are better suited for representing syntactic information for morphologically rich and free-word order languages. However, the results of CoNLL shared tasks on 18 different languages, including Arabic (Nivre et al., 2007a) using either the Malt-Parser (Nivre et al., 2007b) or the MSTParser (McDonald and Crammer, 2005) suggests that Arabic is nonetheless quite a difficult language to parse[1], leaving open the question as to the effectiveness of dependency parsing for Arabic.

One reason for this ineffectiveness is that many parsers do not make much, if any, use of morphological information (Tsarfaty and Sima'an, 2008; Bengoetxea and Gojenola, 2010; Marton et al., 2010). In fact, many established parsing models do not capture visible morphological information provided by wordforms and thus fail to make important distributional distinctions.

In this paper, using a re-implementation of the Berkeley latent-variable PCFG parser we study how morphological features, as encoded in POS

[1]The quality and size of the treebanks certainly are important issues.

tags, can be learned automatically by modifying the distributional restriction of initial grammar symbols, and how they impact Arabic constituency parsing. We have selected PCFG-LA parsing models because they have been shown to be relatively language-independent with state-of-the-art performance for several languages (Petrov, 2009).

Kulick et al. (2006) reported that extending POS tags with definiteness information helps Arabic PCFG parsing[2], Diab (2007) enriched the POS tagset with gender, number and definiteness to improve Arabic base phrase chunking, and Marton et al. (2010) reported that definiteness, person, number and gender were most helpful for Arabic dependency parsing on predicted tag input. Our method is comparable to this work in terms of the investigation of the morphological features. However, the results are not comparable, as we use a different parsing paradigm, a different form of the treebank, and most importantly, we extend the investigation to use several automated feature selection methods.

Increasing the tagset size can lead to data sparsity and generally exacerbates the problem of unknown words (that is, `word:POS` pairs not attested during training). To overcome this problem, we have experimented with the technique presented in Attia et al. (2010) to handle unknown words (out of vocabulary words – OOV) within a generative parsing model. This method employs a list of heuristics to extract morphological clues from word forms and builds a set of word-classes. Our results show that enriching basic POS tags with morphological information, accompanied by a method for handling unknown words, jointly and statistically significantly improve upon the parsing baseline, which uses the Bikel-Bies collapsed POS tagset (Maamouri et al., 2009) and does not employ morphological information to handle unknown words.

This paper is organized as follows: In Section 2, we review the PCFG-LA parsing model and the dataset for our experiments. Section 3 addresses the different techniques we have applied to explore the morphological features space over the possible combinations, including a comparison with the best morphological features of previous works. In Section 4, we describe the results of applying the parser.

2 General Background

2.1 Parsing Models

Johnson (1998) enriched treebank categories with context information to improve the performance of PCFG parsing, then Klein and Manning (2003b) explored manual and automatic extensions of syntactic categories into a richer category set to enhance PCFG parsing results. Later, Matsuzaki et al. (2005) used unsupervised techniques, known as PCFG-Latent Annotation (PCFG-LA), to learn more fine-grained categories from the treebank. This method involves *splitting* categories of the grammar, leading to a better estimation of their distributional properties. It uses a small amount of random noise to break symmetries. Petrov et al. (2006) proposed *merging* categories that produce a loss in likelihood, and *smoothing* to prevent overfitting.

Petrov (2009) discussed efficient methods for learning and parsing with these latent variable grammars, and demonstrates that this formalism can be adapted to a variety of languages and applications.

An important aspect of this approach is the use of random seeds in the EM initialization points, which was only recently treated in Petrov (2010). Using 16 different seeds (1–16), he saw a range of approximately 0.65% in the F-scores of the development set, and a range of about 0.55% in the F-scores of the test set for English. He also found a Pearson correlation coefficient of 0.34 between the accuracies of the development set of the Peen Treebank and the test set, and therefore suggested less of a reliance on any particular random seed to yield better grammars. This led him to propose combining grammars from several seeds to produce a higher-quality product model.

In this work, we use a re-implementation of the Berkeley parser, which trains a PCFG-LA using the split-merge-smooth procedure and parses using the max-rule algorithm (Petrov et al., 2006; Petrov and Klein, 2007). For our experiments, we apply the split-merge-smooth cycle five times[3]

[2]By comparison, the case feature improved parsing for Czech (Collins et al., 1999) and the combination of the number feature for adjectives and mode feature for verbs improved results for Spanish (Cowan and Collins, 2005).

[3]Petrov et al. (2006) reports that the best grammars for English using the Penn Treebank are obtained by repeating this cycle 5 or 6 times, depending on the test set. We opted for five cycles due to the vast difference in training times: 2

and we parse on sentences of less than or equal to 40 words in length. For English, the Berkeley parser explores n-gram suffixes to distinguish between unknown words ending with *-ing*, *-ed*, *-s*, etc. to assign POS tags. The Berkeley parser does not provide the same methodology to Arabic. For Arabic, we apply the technique used by Attia et al. (2010) for the purpose of classification of unknown words. The methodology uses several heuristics based on the exploration of Arabic prefixes, suffixes and templates, and then maps unknown words onto 26 classes.

We present our final experiment on the test set and all intermediate experiments on the development set.

2.2 Corpus: Arabic Penn Treebank

We use the Penn Arabic Treebank (ATB Part3v3.2: Maamouri et al., 2009) and apply the usual treebank split (80% training, 10% development, 10% test; Kulick et al. (2006)).[4] The ATB uses written Modern Standard Arabic newswire and follows the style of the English Penn-II treebank (Marcus et al., 1994). The ATB consists of 12,628 parsed sentences which provides gold segmented, gold vocalized and gold annotated text. More precisely, a tokenized Arabic word is separated from its prefixes and suffixes and it is also segmented to morphemes. For example, the Arabic word الْيَوْمُ is vocalized, segmented and transliterated using Buckwalter transliteration to `Al+yawom+u` (the + day + NOM).

2.3 Part-Of-Speech Tagset

The POS tagset in the ATB covers nouns, pronouns, adjectives, verbs, adverbs, prepositions, interjections, particles, conjunctions, and subordinating conjunctions. This tagset uses 78 atomic components, such as: `ABBREV` for abbreviation, `ACC` for accusative case and `ADJ` for adjective. While there are theoretically hundreds of thousands of full ATB-style POS tags (Habash and Rambow, 2005), only 498 full tags occur in the above-mentioned version of the ATB. For example, `Al+dawol+atayoni` ("the states") receives the following tag `DET+NOUN+NSUFF_FEM_DU_GEN`, annotating a definite, feminine, dual noun in the genitive case.

From this tagset we derive all our experiments. We have identified 15 morphological features from those present in the treebank[5], and our method, based on these features, searches for the optimal POS tagset for parsing the ATB. Table 1 lists these features used in our research. No capitalization is used in Arabic orthography and named entities are often tagged as regular nouns. For this reason we consider "proper noun" as a morphological feature (feature 11) .

1	Determiner	Presence of the prefix *al-* for nouns
2	Person	First person, second person, third person
3	Number	Singular, dual, and plural
4	Aspect	Perfective, imperfective and imperative
5	Voice	Active and passive
6	Mood	Indicative, subjunctive and jussive
7	Gender	Masculine and feminine
8	State	Construct state (*iḍāfa*)
9	Case	Nominative, accusative and genitive. The ATB suffers from accusative-genitive case under-specification for feminine and masculine plurals
10	Definiteness	Noun can be definite even if it does not start with *al-*, e.g. proper nouns are assumed to be inherently definite
11	Proper noun	Name of people, places, and organizations
12	Genitive clitics	Possessive pronouns
13	Negation	Negative markers
14	Particles	Emphatic, restrictive, negative, interrogative, focus, connective, verbal, response conditional
15	Future	Presence of the future prefix *sa-* for verbs

Table 1: Morphological features in Arabic

We investigate two directions to find optimal tagsets, as follows:

Direction 1: starts from ATB top tagset (|498|) and reduces the size of the tagset by excluding the features that hurt parsing performance. In our initial set of experiments, we trained and parsed 15 different configurations, each configuration starting from the top tagset, then excluding one feature. For instance, training and parsing after excluding feature 9, case, produces the best results from the first level, $A_{\top-1}$ (see Table 1).

hours vs. 8 hours, with F-scores being almost the same.

[4]Specifically for every 10 sentences, the first 8 go into the training set, the 9th sentence go into the test set, and the 10th sentence into the dev set.

[5]Annotation guidelines at `http://projects.ldc.upenn.edu/ArabicTreebank`

Direction 2: In order to produce the bottom ATB tagset (|27|), we exclude all 15 features. Then we include one feature at a time, giving us another level $A_{\perp+1}$. For instance, training and parsing after including feature 12, genitive clitics, gives the best results compared with other individual features (see Table 1).

These approaches are further developed in Section 3.

3 Part-Of-Speech and Morphological Features

Since Arabic words are highly inflected, there is a complex interaction between the morphology and the syntax. Thus parsers that have morphological information on hand can make better-informed decisions regarding parse trees. The ATB represents morphological features in the form of complex part-of-speech tags. Features representing agglutinated morphemes are likewise agglutinated in the POS tags, separated by a "+" symbol. Features representing fused morphemes are likewise fused in the POS tags, making feature extraction or conflation particularly challenging.

Due to the large size of the original ATB tagset (|498|), many Arabic parsing systems use the Bikel-Bies POS mapping (Bikel, 2004), which maps the original ATB tagset onto a small, PTB-style tagset of 37 tags, discarding almost all morphological features that are not also present in the English PTB.

But are some of the morphological features helpful in parsing Arabic? Kulick et al. (2006) showed this to be the case. They preserved two morphological features (Determiner and Demonstrative) that would otherwise have been lost in the existing Bikel-Bies POS mapping, and achieved a higher F-score. However, Arabic has many morphological features. We have identified fifteen sets of morphological features in the ATB, such as voice, mood, and grammatical gender (see Table 1).

Including or excluding these features allows for 32,768 combinations (2^{15}). Training and parsing all of these combinations is prohibitively expensive, so an alternative is to use various heuristics on the powerset of the features, some of which are described below. Many of these methods rely on an initial heuristic function $h_{\perp+1}(x)$, which is derived by ranking the results of using individual features. The antichain $A_{\perp+1}$ in Figure 1 gives

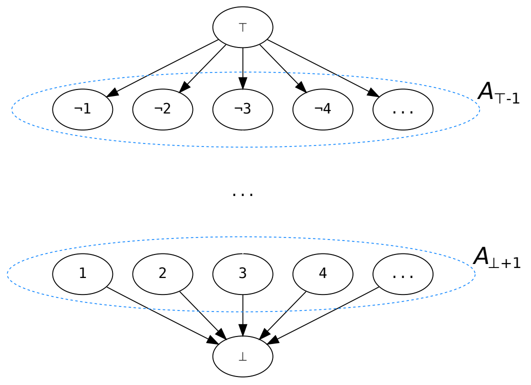


Figure 1: The top node $\top$ represents the tagset containing all ATB morphological features, and the bottom node $\perp$ represents the tagset excluding all morphological features. Each node in the antichain $A_{\top-1}$ excludes one morphological feature, and each node in the antichain $A_{\perp+1}$ includes one morphological feature.

$h_{\perp+1}(x)$. Its dual, $h_{\top-1}(x)$, is derived by ranking the results of not using individual features. The antichain $A_{\top-1}$ in Figure 1 gives $h_{\top-1}(x)$.

We investigated the following automated methods to find the tagset giving the best parse results:

Non-iterative: A non-iterative best-first algorithm that successively folds-in the next-best feature. Thus if there are 5 features ranked individually from best to worst as $\langle 4, 1, 5, 3, 2 \rangle$, then there are 4 tagsets to explore: $\{4, 1\}$ $\{4, 1, 5\}$ $\{4, 1, 5, 3\}$ $\{4, 1, 5, 3, 2\}$. After $h_{\perp+1}(x)$ is defined, the algorithm does not require results from previous steps, and thus can be run in parallel. At most f−1 parses are run, where f is the number of features. The dual of this uses $h_{\top-1}(x)$ to rank the results of excluding exactly one feature.

Greedy-iterative: An iterative, greedy best-first search algorithm with a single heuristic function throughout. We use the heuristic function $h_{\perp+1}(x)$ to rank the results of including exactly one feature, then iteratively fold-in the next feature and retain it only if it achieves a higher score. This requires f−1 iterations after $h_{\perp+1}(x)$ is obtained. This method was used by Marton et al. (2010) and Marton et al. (2011). The dual of this uses $h_{\top-1}(x)$ to rank the results of excluding exactly one feature.

Merged: The previous two methods can instead use a different heuristic function by merging the results of $A_{\perp+1}$ with the results of $A_{\top-1}$. This potentially provides more robust results than either one individually, without requiring any additional iterations. The merge function $M(f)$ is defined as:

$$M(f) = (S_\top - S_{\neg f}) + (S_f - S_\perp)$$

where S_f is the F-score of including a given feature f. The results of $M(f)$ are ordered to provide the merged heuristic function $h_m(x)$.

Backtrack: An iterative, best-first backtracking algorithm that updates its heuristic function at each iteration. Whereas the previous algorithm uses a single heuristic function throughout (such as $h_{\perp+1}(x)$), for this algorithm the next feature chosen is decided by $h_i(x)$, which is based upon the results of the current iteration i. The most helpful feature at each iteration is chosen. After exhausting this path, it will backtrack, discarding the most-recently added feature and instead revisit $h_{i-1}(x)$. The dual of this uses $h_{\top-1}(x)$ to rank the results of excluding exactly one feature, and starts from the top.

Like beam-stack search (Zhou and Hansen, 2005), this algorithm has the advantage of being an anytime search algorithm, which quickly finds good solutions while still finding a globally optimal solution. However, this algorithm is much simpler conceptually than beam-stack search, it proceeds best-first, and the beam width is determined by the number of unused features.

4 Experimental Results

We have experimented with the strategies presented in section 3. All our feature selection experiments are based on the results of vocalized no-tag parsing on the development set where the parser assigns the tags learned during the training phase. However we also provide the results of gold tag and unvocalized no-tag parsing in our final experiments. We measure quality and coverage of the output trees using the standard EVALB (Satoshi and Collins, 1997).

The initial results on parsing are presented in Table 2. These results describe the first stage of traversing the powerset of features. Only one feature is included (in $A_{\perp+1}$) or excluded (in $A_{\top-1}$) at a time. The Bikel-Bies POS tagset included in the Penn Arabic Treebank part 3 v3.2 represents our baseline.[6] At this stage, we use the obtained F-scores to derive our initial $h(x)$ for all methods, since we are interested in measuring relative improvements when changing POS tagsets.

Bikel-Bies POS	81.33	
$\top$	81.99	
$\perp$	81.98	
Features	$\mathbf{A}_{\top-1}$	$\mathbf{A}_{\perp+1}$
1 Determiner	81.52	82.33
2 Person	81.80	81.46
3 Number	81.64	81.56
4 Aspect	82.00	81.81
5 Voice	81.99	82.57
6 Mood	82.21	82.27
7 Gender	82.06	82.15
8 State	81.60	82.07
9 Case	**83.00**	81.68
10 Definiteness	82.10	82.41
11 Proper noun	82.13	82.27
12 Genitive clitics	82.26	**82.69**
13 Negation	81.71	82.42
14 Particles	81.20	82.61
15 Future	81.70	82.25

Table 2: Parsing results (F-score) for including features 1–15 in $A_{\perp+1}$ and excluding features 1–15 in $A_{\top-1}$.

The results presented in Table 3 correspond to the non-iterative method. Line 6 shows that the improvement for this configuration is mainly due to adding feature 1 (determiner) and lines 7–8 show that feature 11 (proper noun) is helpful only when it is associated with feature 6 (mood).

Table 4 presents the results of the greedy-iterative method using a single heuristic function (left) and a merged heuristic function (right). The method using single heuristic function was also employed by Marton et al. (2010). Grey rows indicate that the feature decreased the F-score, and thus was discarded. The number of iterations with this algorithm is fixed at f–1 after the initial heuristic function is obtained, where f is the number of morphological features. We can observe that the features 11,13 (proper noun & negation) work together to produce the highest jump in the score.

In contrast with the results of Marton et al.

[6]This tagset has 37 unique tags in the treebank.

$\perp$	81.98
$\{12\}$	82.69
$\{12, 14\}$	82.36
$\{12, 14, 5\}$	82.48
$\{12, 14, 5, 13\}$	82.20
$\{12, 14, 5, 13, 10\}$	82.49
$\{12, 14, 5, 13, 10, 1\}$	82.94
$\{12, 14, 5, 13, 10, 1, 11\}$	82.78
$\{\mathbf{12, 14, 5, 13, 10, 1, 11, 6}\}$	**83.02**
$\{12, 14, 5, 13, 10, 1, 11, 6, 15\}$	82.54
...	...

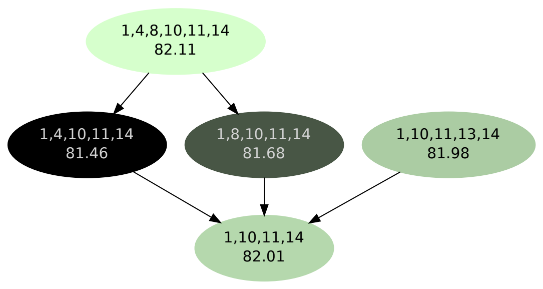
Table 3: Parsing results for the non-iterative method. The tagset with highest F-score contains genitive clitics, particles, voice, negation, definiteness, determiner, proper noun, and mood. The improvement over the baseline is statistically significant.

$\perp$	81.98	$\perp$	81.98
$\{12\}$	82.69	$\{10\}$	82.41
$\{12, 14\}$	82.36	$\{10, 13\}$	82.97
$\{12, 5\}$	82.44	$\{10, 13, 14\}$	82.38
$\{12, 13\}$	82.24	$\{10, 13, 1\}$	82.65
$\{12, 10\}$	81.79	$\{10, 13, 8\}$	82.44
$\{12, 1\}$	81.80	$\{10, 13, 3\}$	82.22
$\{\mathbf{12, 11}\}$	**82.87**	$\{10, 13, 7\}$	82.48
$\{12, 11, 6\}$	82.33	$\{10, 13, 5\}$	82.00
$\{12, 11, 15\}$	82.59	$\{\mathbf{10, 13, 11}\}$	**83.07**
$\{12, 11, 7\}$	81.83	$\{10, 13, 11, 12\}$	82.378
$\{12, 11, 8\}$	82.01	$\{10, 13, 11, 6\}$	82.38
$\{12, 11, 4\}$	81.98	$\{10, 13, 11.2\}$	82.45
$\{12, 11, 9\}$	82.19	$\{10, 13, 11, 4\}$	82.89
$\{12, 11, 3\}$	82.40	$\{10, 13, 11, 15\}$	82.30
$\{12, 11, 2\}$	81.92	$\{10, 13, 11, 9\}$	82.56

Table 4: Parsing results for the greedy-best-first method with a single heuristic, using $h_{\perp+1}(x)$ in the left table and $h_m(x)$ in the right table. The tagset with highest F-score on the left contains genitive clitics and proper noun markers. However, the improvement over the baseline is not statistically significant. The tagset with highest F-score on the right contains definiteness, negation and proper noun markers. In this case, the improvement over the baseline is statistically significant.

(2010), the so-called ϕ-features (person, number, gender) did not contribute to a higher score in our experiments. We believe this is in part because morphological features are represented as atomic tags in the Penn Arabic Treebank. Generative parsers that use this style of treebank typically do not analyze individual features and make use of them in determining agreement relationships. On the other hand, the CoNLL-X format[7] provides a field where morphological features are specified individually. This encourages dependency parsers to inspect individual components of a tag, and make use of specific features when helpful. The CoNLL-X format is used by most dependency parsers, including the MaltParser used in Marton et al. (2010).

The landscape of the search space contains many local maxima, which can prevent (non-backtracking) greedy algorithms from exploring paths that eventually lead to high scores. Consider the case below:



A greedy algorithm arriving at feature combination 1,10,11,14 moving upward will pursue an unfruitful 1,10,11,13,14, never arriving at the highest feature combination 1,4,8,10,11,14. Features 4 (aspect) and 8 (state) may work together to provide a useful distinction for parsing that individually would otherwise only increase sparsity.

Table 5 gives the results of using the best-first with backtracking algorithm. We join the highest-ranked individual feature (12: genitive clitics) with all other individual features (i.e. $\{12, 1\}$ $\{12, 2\}$ $\{12, 3\}$ $\ldots$), and select the combination achieving the highest F-score ($\{12, 11\}$). The initial heuristic function $h_{\perp+1}(x)$ is used only in the first iteration (to select feature 12 in our case). Afterward, we join the resulting set with all other remaining individual features (i.e. $\{12, 11, 1\}$ $\{12, 11, 2\}$ $\{12, 11, 3\}$ $\ldots$), and select the combination achieving the highest score. With each iteration, the number of remaining features is decremented by one. While this algorithm is exhaustive, we have not explored all possible combinations.

We also investigated a probabilistic search algorithm to see how well it could overcome the challenges posed by many local maxima. Using simulated annealing (Kirkpatrick et al., 1983), we performed 50 experiments, with 50 cycles each. The mean of the final dev-set F-scores was 82.84,

[7] http://nextens.uvt.nl/~conll/#dataformat

$\perp$	81.98
$\{12\}$	82.69
$\dots$	
$\{12, 11\}$	82.87
$\dots$	
$\{12, 11, 13\}$	82.60
$\dots$	
$\mathbf{\{12, 11, 13, 1\}}$	**83.16**
$\dots$	

Table 5: Parsing results for the best-first with backtracking method. The improvement of the score over the baseline is statistically significant. The best feature combination includes the following features: genitive clitics, proper noun, negation, and determiner. The cardinality of the best-performing tagset is 41.

with a standard deviation of 0.21 . As is usually the case, the cooling schedule played an important role in the results. We evaluated three different cooling schedules, and found that the slowest one resulted in many low scores, given the same number of cycles. This was due to the higher probability of jumping to a higher energy state later in the experiment. This is often advantageous given a large number of cycles, however we are limited to fewer cycles due to the high cost of performing each training/parsing cycle.

We observed that simulated annealing usually required many more cycles to find a good score (e.g. higher than 82.40) than the previous search methods described. This was due to their differences in starting points and movement strategies. The previous methods started from the bottom and added potentially helpful features, in various ways. On the other hand, simulated annealing started at a random location in the powerset, and moved stochastically. The differences in the results indicate that many of the high scores lie relatively near the bottom, whereas there is much greater uncertainty in the middle of the feature powerset.

For comparative purposes, Table 6 presents the results obtained by using the most helpful features proposed in two previous works.[8] The Determiner feature in the first row was added by Kulick et al. (2006) to the Bikel-Bies POS tagset, using the Bikel parser. The features in the last two rows were determined by using the MaltParser with the Columbia Arabic Treebank (CATiB: Habash and Roth, 2009) and discussed in Marton et al. (2010).

[8]Using the same setup as in the other experiments in this paper (eg. PCFG-LA parser and ATB3v3.2).

While the case feature (9) helped their gold-tag parsing, it was not helpful for either vocalized or unvocalized parsing in our experiments. Case markings in Arabic exhibit ambiguity with certain noun forms—there are particular instances where both the genitive and accusative endings are the same, such as the duals and masculine plurals. Related to this is the imperfect alignment in Arabic between true grammatical function and morphological case markings (in vocalized and to a lesser extent unvocalized text).

As expected, these features do not achieve the highest overall F-score. Given the variability with different parsers, annotation schemes, evaluation metrics, etc., it should not be surprising that there is no "universally-best" tagset for a language, but rather a tagset optimized for a given task. For example, while the gender feature has not benefited PCFG-LA parsing in our experiments, it could be vitally important for MT applications. But these systems must be able to readily access these individual features, or they may not be utilized.

Kulick +Determiner	$\{1\}$	82.33
Marton best predicted tags	$\{1, 2, 3, 7\}$	79.23
Marton best gold tags	$\{8, 9\}$	82.40

Table 6: F-scores on tag combinations proposed by previous investigations, using the PCFG-LA parser.

Tables 7 and 8 present the final results on the vocalized development set and test set. We applied significance tests on all the results in these tables, and significantly-improved F-scores are indicated with asterisks. The no-tag column gives F-scores when the parser was not given any part-of-speech information in the test (or dev) set at all—just the text to be parsed. Tables 9 and 10 present the final results on the unvocalized development set and test set.

We have also investigated multiple, automatically-learned grammars that differ only in the random seed used to initialize the EM learning algorithm. We explored seeds 1–50, and found a statistically significant difference of 1.24% between the highest EM initialization seed and the lowest, and decided to pursue combining seeds into product models (Petrov, 2010). We explored different *seed combinations* (rather than feature combinations, as before) to form the product models of the baseline and the highest feature combination found in the development set. Using the non-iterative search method described in section 3, we took the highest-scoring 16 seeds from seeds 1–50 and successively folded-in the next-best seed. Figure 2 shows that the F-scores of the vocalized models tend to level off after incorporating the four highest-scoring seeds. The unvocalized counterparts see continued gradual improvements with larger product models, possibly due to less data sparsity.

Tagset	No-tag	Gold
Bikel-Bies Baseline	81.33	85.36
Bikel-Bies + OOV	82.23^{**}	85.59^{*}
$\{12, 11, 13, 1\}$ + OOV	83.16^{***}	85.94^{***}
Bikel-Bies Baseline + product models	83.70	87.02
$\{12, 11, 13, 1\}$+OOV+product models	**84.40*****	87.52^{***}

Table 7: Final F-scores on the *vocalized development set* for the best feature combination, and handling unknown words. The best feature combination included the following features: genitive clitics, proper noun, negation, and determiner. Statistically significant with *=$p < 0.05$, **=$p < 0.01$, ***= $p < 0.001$

Tagset	No-tag	Gold
Bikel-Bies Baseline	80.69	85.03
Bikel-Bies + OOV	82.45^{***}	85.29
$\{12, 11, 13, 1\}$ + OOV	83.55^{***}	85.89^{*}
Bikel-Bies Baseline + product models	82.89	87.10
$\{12, 11, 13, 1\}$+OOV+product models	**85.03*****	87.57^{***}

Table 8: Final F-scores on the *vocalized test set* for the best feature combination, and handling unknown words. Statistically significant with *=$p < 0.05$, **=$p < 0.01$, ***= $p < 0.001$

Tagset	No-tag	Gold
Bikel-Bies Baseline	80.08	85.12
Bikel-Bies + OOV	81.44^{***}	85.42
$\{12, 11, 13, 1\}$ + OOV	82.30^{***}	86.67^{**}
Bikel-Bies Baseline + product models	82.33	87.49
$\{12, 11, 13, 1\}$+OOV+product models	**84.10*****	87.89

Table 9: Final F-scores on the *unvocalized development set* for the best feature combination, and handling unknown words. The best feature combination included the following features: genitive clitics, proper noun, negation, and determiner. Statistically significant with *=$p < 0.05$, **=$p < 0.01$, ***= $p < 0.001$

Tagset	No-tag	Gold
Bikel-Bies Baseline	80.26	85.61
Bikel-Bies + OOV	80.72	85.83
$\{12, 11, 13, 1\}$ + OOV	82.14^{***}	86.20
Bikel-Bies Baseline + product models	81.69	87.23
$\{12, 11, 13, 1\}$+OOV+product models	**83.34*****	87.38

Table 10: Final F-scores on the *unvocalized test set* for the best feature combination, and handling unknown words. Statistically significant with *=$p < 0.05$, **=$p < 0.01$, ***= $p < 0.001$

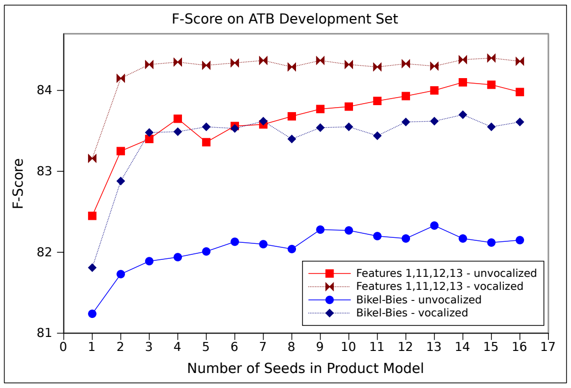


Figure 2: Development set F-scores using product models from multiple grammars.

5 Conclusion

This paper focuses on finding the best use of morphological features for PCFG-LA parsing. We identify 15 morphological features and use the original Penn Arabic Treebank POS tagset (|498|) and a shallow version that excludes all morphological features (|27|), and apply feature inclusion or exclusion to calculate the optimal feature combination for Arabic parsing. We show that using morphological information helps parsing even though it results in a larger tagset. In order to diminish the impact of the newly created POS tags on unknown words, we use a list of Arabic signatures to differentiate between these unknown words when assigning POS tags based on string examination.

We have applied several search methods to find the feature combination that improves grammar quality the most, using **i**) a non-iterative best-first search algorithm, **ii**) an iterative, greedy best-first search algorithm with a single heuristic function, **iii**) an iterative, best-first with backtracking search algorithm, and **iv**) simulated annealing.

The best-first with backtracking algorithm has provided the best results, achieving state-of-the-art

F-scores of 85.03 for vocalized ATB no-tag parsing and 83.34 for unvocalized ATB no-tag parsing, significant improvements of 2.17% and 1.88% over the baseline. These results, together with the scores of the other search algorithms, suggest that the optimal morphological feature combination for this task involves including just a few features. Three out of the four features from our optimal tagset occur in noun phrases. Since noun phrases are so common[9], features that can help parse just these phrases appear to have a great impact on overall F-scores.

We have also performed experiments using features highlighted in previous studies on different parsing models, and have shown that considering only one tagset for a language does not provide optimal scores.

Acknowledgements

This research is funded by Enterprise Ireland (PC/09/037). We would like to thank Chris Brew, Mike White, William Schuler, Jennifer Foster, Joseph Le Roux, Eric Fosler-Lussier, Dennis Mehay, Andy Plummer, and anonymous reviewers for all their very helpful suggestions.

References

Attia, M., Foster, J., Hogan, D., Le Roux, J., Tounsi, L., and van Genabith, J. (2010). Handling unknown words in statistical latent-variable parsing models for Arabic, English and French. In *NAACL/ HLT Workshop on Statistical Parsing of Morphologically Rich Languages*, pages 67–75.

Bengoetxea, K. and Gojenola, K. (2010). Application of different techniques to dependency parsing of Basque. In *NAACL/ HLT Workshop on Statistical Parsing of Morphologically Rich Languages*, pages 37–39.

Bikel, D. (2004). *On the Parameter Space of Generative Lexicalized Parsing Models*. PhD thesis, University of Pennslyvania.

Collins, M., Hajič, J., Ranshaw, L., and Tillman, C. (1999). A statistical parser for Czech. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Cowan, B. and Collins, M. (2005). Morphology and reranking for the statistical parsing of Spanish. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 795–802, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Diab, M. T. (2007). Improved Arabic base phrase chunking with a new enriched POS tag set. In *ACL Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources*, pages 89–96.

Habash, N. and Rambow, O. (2005). Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 573–580, Stroudsburg, PA, USA. Association for Computational Linguistics.

Habash, N. and Roth, R. (2009). CATiB: The Columbia Arabic Treebank. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 221–224, Suntec, Singapore. Association for Computational Linguistics.

Johnson, M. (1998). PCFG models of linguistic tree representations. *Computational Linguistics*, 24(4):613–632.

Kirkpatrick, S., Gelatt, C. D. J., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

Klein, D. and Manning, C. (2003a). A* parsing: fast exact viterbi parse selection. In *Proceedings of the North American Association for Association for Computational Linguistics*.

Klein, D. and Manning, C. (2003b). Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the ACL*, pages 423–430.

Kulick, S., Gabbard, R., and Marcus, M. (2006). Parsing the Arabic Treebank: Analysis and improvements. In *Proceedings of the Treebanks and Linguistic Theories Conference (TLT-2006)*, pages 31–42, Prague, Czech Republic.

Maamouri, M., Bies, A., and Kulic, S. (2009). Creating a methodology for large-scale correction of treebank annotation: The case of the Arabic treebank. In Choukri, K. and Maegaard,

[9]`NP`'s comprise 58% of all non-terminals in the treebank that we used. They are more than five times more frequent than the next most common non-terminal (`PP`).

B., editors, *MEDAR Second International Conference on Arabic Language Resources and Tools, Egypt*.

Marcus, M., Kim, G., Marcinkiewicz, M. A., MacIntyre, R., Bies, A., Ferguson, M., Katz, K., and Schasberger, B. (1994). The Penn Treebank: Annotating predicate argument structure. In *Proceedings of the 1994 ARPA Speech and Natural Language Workshop*, pages 114–119, Princeton, NJ, USA.

Marton, Y., Habash, N., and Rambow, O. (2010). Improving Arabic dependency parsing with lexical and inflectional morphological features. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 13–21, Los Angeles, CA, USA. Association for Computational Linguistics.

Marton, Y., Habash, N., and Rambow, O. (2011). Improving Arabic dependency parsing with form-based and functional morphological features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1586–1596, Portland, Oregon, USA. Association for Computational Linguistics.

Matsuzaki, T., Miyao, Y., and Tsujii, J. (2005). Probabilistic CFG with latent annotations. In *Proceedings of the 43rd Annual Meeting of the ACL*, pages 75–82, Ann Arbor, MI, USA.

McDonald, R. and Crammer, K. (2005). Online large-margin training of dependency parsers. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel, S., and Yuret, D. (2007a). The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague, Czech Republic. Association for Computational Linguistics.

Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., and Marsi, E. (2007b). Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Petrov, S. (2009). *Coarse-to-Fine Natural Language Processing*. PhD thesis, University of California at Berkeley, Berkeley, CA, USA.

Petrov, S. (2010). Products of random latent variable grammars. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 19–27, Los Angeles, CA, USA. Association for Computational Linguistics.

Petrov, S., Barrett, L., Thibaux, R., and Klein, D. (2006). Learning accurate, compact and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the ACL*, Sydney, Australia.

Petrov, S. and Klein, D. (2007). Improved inference for unlexicalized parsing. In *Proceedings of HLT-NAACL*, Rochester, NY, USA.

Satoshi, S. and Collins, M. (1997). Evalb bracket scoring program.

Tsarfaty, R. and Sima'an, K. (2008). Relational-realizational parsing. In *Proceedings of Conference on Computational Linguistics (CoLing)*, pages 18–22.

Zhou, R. and Hansen, E. A. (2005). Beam-stack search: Integrating backtracking with beam search. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, pages 90–98, Monterey, CA, USA.