Randomized Greedy Inference for Joint Segmentation, POS Tagging and Dependency Parsing

Yuan Zhang, Chengtao Li, Regina Barzilay
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
{yuanzh, ctli, regina}@csail.mit.edu

Kareem Darwish
ALT Research Group
Qatar Computing Research Institute
kdarwish@qf.org.qa

Abstract

In this paper, we introduce a new approach for joint segmentation, POS tagging and dependency parsing. While joint modeling of these tasks addresses the issue of error propagation inherent in traditional pipeline architectures, it also complicates the inference task. Past research has addressed this challenge by placing constraints on the scoring function. In contrast, we propose an approach that can handle arbitrarily complex scoring functions. Specifically, we employ a randomized greedy algorithm that jointly predicts segmentations, POS tags and dependency trees. Moreover, this architecture readily handles different segmentation tasks, such as morphological segmentation for Arabic and word segmentation for Chinese. The joint model outperforms the state-of-the-art systems on three datasets, obtaining 2.1% TedEval absolute gain against the best published results in the 2013 SPMRL shared task.[1]

1 Introduction

Parsing accuracy is greatly impacted by the quality of preprocessing steps such as tagging and word segmentation. Li et al. (2011) report that the difference between using the gold POS tags and using the automatic counterparts reaches about 6% in dependency accuracy. Prior research has demonstrated that joint prediction alleviates error propagation inherent in pipeline architectures, where mistakes cascade from one task to the next (Bohnet et al., 2013; Tratz, 2013; Hatori et al., 2012; Zhang et al., 2014a). However, jointly modeling all the processing tasks inevitably increases inference complexity. Prior work addressed this challenge by introducing constraints on scoring functions to keep inference tractable (Qian and Liu, 2012).

In this paper, we propose a method for joint prediction that imposes no constraints on the scoring function. The method is able to handle high-order and global features for each individual task (e.g., parsing), as well as features that capture interactions between tasks. The algorithm achieves this flexibility by operating over full assignments that specify segmentation, POS tags and dependency tree, moving from one complete configuration to another.

Our approach is based on the randomized greedy algorithm from our earlier dependency parsing system (Zhang et al., 2014b). We extend this algorithm to jointly predict the segmentation and the POS tags in addition to the dependency parse. The search space for the algorithm is a combination of parse trees and lattices that encode alternative morphological and POS analyses. The inference algorithm greedily searches over this space, iteratively making local modifications to POS tags and dependency trees. To overcome local optima, we employ multiple restarts.

This simple, yet powerful approach can be easily applied to a range of joint prediction tasks. In prior work, joint models have been designed for a specific language. For instance, joint models for Chinese are designed with word segmentation in mind (Hatori et al., 2012), while algorithms for processing Semitic languages are tailored for morpho-

[1]The source code is available at https://github.com/yuanzh/SegParser.

Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL, pages 42–52,
Denver, Colorado, May 31 – June 5, 2015.

logical analysis (Tratz, 2013; Goldberg and Elhadad, 2011). In contrast, we show that our algorithm can be effortlessly applied to all these distinct languages. Language-specific characteristics drive the lattice construction and the feature selection, while the learning and inference methods are language-agnostic.

We evaluate our model on three datasets: SPMRL (Modern Standard Arabic), classical Arabic and CTB5 (Chinese). Our model consistently outperforms state-of-the-art systems designed for these languages. We obtain a 2.1% TedEval gain against the best published results in the 2013 SPMRL shared task (Seddah et al., 2013). The joint model results in significant gains against its pipeline counterpart, yielding 2.4% absolute F-score increase in dependency parsing on the same dataset. Our analysis reveals that most of this gain comes from the improved prediction on OOV words.

2 Related Work

Joint Segmentation, POS tagging and Syntactic Parsing It has been widely recognized that joint prediction is an appealing alternative for pipeline architectures (Goldberg and Tsarfaty, 2008; Hatori et al., 2012; Habash and Rambow, 2005; Gahbiche-Braham et al., 2012; Zhang and Clark, 2008; Bohnet and Nivre, 2012). These approaches have been particularly prominent for languages with difficult preprocessing, such as morphologically rich languages (e.g., Arabic and Hebrew) and languages that require word segmentation (e.g., Chinese). For the former, joint prediction models typically rely on a lattice structure to represent alternative morphological analyses (Goldberg and Tsarfaty, 2008; Tratz, 2013; Cohen and Smith, 2007). For instance, transition-based models intertwine operations on the lattice with operations on a dependency tree. Other joint architectures are more decoupled: in Goldberg and Tsarfaty (2008), a lattice is used to derive the best morphological analysis for each part-of-speech alternative, which is in turn provided to the parsing algorithm. In both cases, tractable inference is achieved by limiting the representation power of the scoring function. Our model also uses a lattice to encode alternative analyses. However, we employ this structure in a different way. The model samples the full path from the lattice, which corresponds to a valid segmentation and POS tagging assignment. Then the model improves the path and the corresponding tree via a hill-climbing strategy. This architecture allows us to incorporate arbitrary features for segmentation, POS tagging and parsing.

In joint prediction models for Chinese, lattice structures are not typically used. Commonly these models are formulated in a transition-based framework at the character level (Zhang and Clark, 2008; Zhang et al., 2014a; Wang and Xue, 2014). While this formulation can handle a large space of possible word segmentations, it can only capture features that are instantiated based on the stack and queue status. Our approach offers two advantages over prior work: (1) we can incorporate arbitrary features for word segmentation and parsing; (2) we demonstrate that a lattice-based approach commonly used for other languages can be effectively utilized for Chinese.

Randomized Greedy Inference Our prior work has demonstrated that a simple randomized greedy approach delivers near optimal dependency parsing (Zhang et al., 2014b). Our analysis explains this performance with the particular properties of the search space in dependency parsing. We show how to apply this strategy to a more challenging inference task and demonstrate that a randomized greedy algorithm achieves excellent performance in a significantly larger search space.

3 Randomized Greedy System for Joint Prediction

In this section, we introduce our model for joint morphological segmentation, tagging and parsing. Our description will first assume that word boundaries are provided (e.g., the case of Arabic). Later, we will describe how this model can be applied to a joint prediction task that involves word segmentation (e.g., Chinese).

3.1 Notation

Let $x = \{x_i\}_{i=1}^{|x|}$ be a sentence of length $|x|$ that consists of tokens x_i. We use $s = \{s_i\}_{i=1}^{|x|}$ to denote a segmentation of all the tokens in sentence x, and $s_i = \{s_{i,j}\}_{j=1}^{|s_i|}$ to denote a segmentation of the token x_i, where $s_{i,j}$ is the jth morpheme of the token x_i. Similarly, we use t, t_i and $t_{i,j}$ for the POS

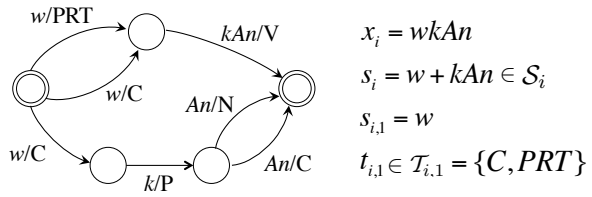


Figure 1: Example lattice structures for the Arabic token "*wkAn*". It has two candidate segmentations: *w+kAn* or *w+k+An*. The first segmentation consists of two morphemes. The first morpheme *w* has two candidate POS.

tags for each sentence, token and morpheme. We use y to denote a dependency tree over morphemes, and $y_{i,j}$ to denote the head of morpheme $s_{i,j}$. During training, the algorithm is provided with tuples that specify ground truth values for all the variables $\mathcal{D} = \{(x, \hat{s}, \hat{t}, \hat{y})\}$.

We also assume access to a morphological analyzer and a POS tagger that provide candidate analyses. Specifically, for each token x_i, the algorithm is provided with candidate segmentations $\mathcal{S}_i$, and candidate POS tags $\mathcal{T}_i$ and $\mathcal{T}_{i,j}$. These alternative analyses are captured in the **lattice structure** (see Figure 1 for an example). Finally, we use $\mathcal{Y}$ to denote the set of all valid dependency trees defined over morphemes.

3.2 Decoding

We parameterize the scoring function as

$$score(x, s, t, y) = \theta \cdot f(x, s, t, y) \qquad (1)$$

where θ is the parameter vector and $f(x, s, t, y)$ is the feature vector associated with the sentence and all variables.

The goal of decoding is to find a set of valid values for $(s, t, y) \in \mathcal{S} \times \mathcal{T} \times \mathcal{Y}$ that maximizes the score defined in Eq. 1. Our randomized greedy algorithm finds a high scoring assignment for (s, t, y) via a hill-climbing process with multiple random restarts. (Section 3.3 describes how the parameters θ are learned.)

Figure 2 shows the framework of our randomized greedy algorithm. First, we draw a full path from the lattice structure in two steps: (1) sampling a morphological segmentation s from $\mathcal{S}$; (2) sampling POS tags t for each morpheme. Next, we sample a dependency tree y from the parse space. Based on this random starting point, we iteratively hill-climb t and y in a bottom-up order.[2] In our earlier work (Zhang et al., 2014b), we showed this strategy guarantees that we can climb to any target tree in a finite number of steps. We repeat the sampling and the hill-climbing processes above until we do not find a better solution for K iterations. We introduce the details of this process below.

SampleSeg and SamplePOS: Given a sentence x, we first draw segmentations s and POS tags $t^{(0)}$ from the first-order distribution using the current learned parameter values. For segmentation, first-order features only depend on each token x_i and its morphemes $s_{i,j}$. Similarly, for POS, first-order features are defined based on $s_{i,j}$ and $t_{i,j}$. The sampling process is straightforward due to the fact that the candidate sets $|\mathcal{S}_i|$ and $|\mathcal{T}_{i,j}|$ are both small. We can enumerate and compute the probabilities proportional to the exponential of the first-order scores as follows.[3]

$$\begin{aligned} p(s_i) &\propto \exp\{\theta \cdot f(x, s_i)\} \\ p(t_{i,j}) &\propto \exp\{\theta \cdot f(x, s_i, t_{i,j})\} \end{aligned} \qquad (2)$$

SampleTree: Given a random sample of the segmentations s and the POS tags $t^{(0)}$, we draw a random tree $y^{(0)}$ from the first-order distribution using Wilson's algorithm (Wilson, 1996).[4]

HillClimbPOS: After sampling the initial values s, $t^{(0)}$ and $y^{(0)}$, the hill-climbing algorithm improves the solution via locally greedy changes. The hill-climbing algorithm iterates between improving the POS tags and the dependency tree. For POS tagging, it updates each $t_{i,j}$ in a bottom-up order as follows

$$t_{i,j} \leftarrow \underset{t_{i,j} \in \mathcal{T}_{i,j}}{\arg\max}\, score(x, s, t_{i,j}, t_{-(i,j)}, y) \qquad (3)$$

where $t_{-(i,j)}$ are the rest of the POS tags when we update $t_{i,j}$.

[2] We do not hill-climb segmentation, or else we have to jointly find the optimal t and y, and the resulting computational cost is too high.

[3] We notice that the distribution becomes significantly sharper after training for several epochs. Therefore, we also smooth the distribution by multiplying the score with a scaling factor.

[4] We also smooth the distribution in the same way as in segmentation and POS tagging.

Input: parameter θ, sentence x
Output: segmentations s, POS tags t and dependency tree y

1: $s \leftarrow SampleSeg(x)$
2: $t^{(0)} \leftarrow SamplePos(x, s)$
3: $y^{(0)} \leftarrow SampleTree(x, s, t^{(0)})$
4: $k = 0$
5: **repeat**
6: $\quad t^{(k+1)} \leftarrow HillClimbPOS(x, s, t^{(k)}, y^{(k)})$
7: $\quad y^{(k+1)} \leftarrow HillClimbTree(x, s, t^{(k+1)}, y^{(k)})$
8: $\quad k \leftarrow k + 1$
9: **until** no change in this iteration
10: **return** $(s, t^{(k)}, y^{(k)})$

Figure 2: The hill-climbing algorithm with random initializations. Details of the sampling and hill-climbing functions in Line 1-3 and 6-7 are provided in Section 3.2.

HillClimbTree: We improve the dependency tree y via a similar hill-climbing process. Specifically, we greedily update the head $y_{i,j}$ of each morpheme in a bottom-up order as follows

$$y_{i,j} \leftarrow \underset{y_{i,j} \in \mathcal{Y}_{i,j}}{\arg\max}\, score(x, s, t, y_{i,j}, y_{-(i,j)}) \quad (4)$$

where $\mathcal{Y}_{i,j}$ is the set of candidate heads such that changing $y_{i,j}$ to any candidate does not violate the tree constraint.

3.3 Training

We learn the parameters θ in a max-margin framework, using on-line updates. For each update, we need to compute the segmentations, POS tags and the tree that maximize the cost-augmented score:

$$(\tilde{s}, \tilde{t}, \tilde{y}) = \underset{s \in \mathcal{S}, t \in \mathcal{T}, y \in \mathcal{Y}}{\arg\max} \{\theta \cdot f(x, s, t, y) + Err(s, t, y)\} \quad (5)$$

where $Err(s, t, y)$ is the number of errors of (s, t, y) against the ground truth $(\hat{s}, \hat{t}, \hat{y})$. The parameters are then updated to guide the selection against the violation. This is done via standard passive-aggressive updates (Crammer et al., 2006).

3.4 Adapting to Chinese Joint Prediction

In this section we describe how the proposed model can be adapted to languages that do not delineate words with spaces, and thus require word segmentation. The main difference lies in the construction of the lattice structure. We employ a state-of-the-art word segmenter to produce candidate word boundaries. We consider boundaries common across all the top-k candidates as true word boundaries. The remaining tokens (i.e., strings between these boundaries) are treated as words to be further segmented and labeled with POS tags. Figure 3 shows an example of the Chinese word lattice structure we construct. Once the lattice is constructed, the joint prediction model is applied as described above.

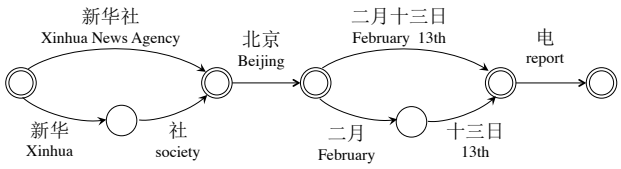


Figure 3: Example lattice structures for the Chinese sentence "新华社北京二月十三日电" (Xinhua Press at Beijing reports on February 13th). The token 新华社 has two candidate segmentations: 新华社 or 新华 + 社.

4 Features

Segmentation Features For both Arabic and Chinese, each segmentation is represented by its score from the preprocessing system, and by the corresponding morphemes (or words in Chinese). Following previous work (Zhang and Clark, 2010), we also add character-based features for Chinese word segmentation, including the first and the last characters in the word, and the length of the word.

POS Tag Features Table 1 summarizes the POS tag features employed by the model. First, we use the feature templates proposed in our previous work on Arabic joint parsing and POS correction (Zhang et al., 2014c). In addition, we incorporate character-based features specifically designed for Chinese. These features are mainly inspired by previous transition-based models on Chinese joint POS tagging and word segmentation (Zhang and Clark, 2010).

Dependency Parsing Features The feature templates for dependency parsing are mainly drawn from our previous work (Zhang et al., 2014b). Fig-

1-gram	$\langle t_0, w_{-2}\rangle, \langle t_0, w_{-1}\rangle, \langle t_0, w_0\rangle, \langle t_0, w_1\rangle, \langle t_0, w_2\rangle,$ $\langle t_0, w_{-1}, w_0\rangle, \langle t_0, w_0, w_1\rangle, \langle s(t_0)\rangle, \langle t_0, s(t_0)\rangle$
2-gram	$\langle t_{-1}, t_0\rangle, \langle t_{-2}, t_0\rangle, \langle t_{-1}, t_0, w_{-1}\rangle, \langle t_{-1}, t_0, w_0\rangle$
3-gram	$\langle t_{-1}, t_0, t_1\rangle, \langle t_{-2}, t_0, t_1, \rangle, \langle t_{-1}, t_0, t_2\rangle,$ $\langle t_{-2}, t_0, t_2\rangle$
4-gram	$\langle t_{-2}, t_{-1}, t_0, t_{+1}\rangle, \langle t_{-2}, t_{-1}, t_0, t_2\rangle,$ $\langle t_{-2}, t_0, t_1, t_2\rangle$
5-gram	$\langle t_{-2}, t_{-1}, t_0, t_1, t_2\rangle$
Character	$\langle t_0, pre_1(w_0)\rangle, \langle t_0, pre_2(w_0)\rangle, \langle t_0, suf_1(w_0)\rangle,$ $\langle t_0, suf_2(w_0)\rangle, \langle t_0, c_n(w_0)\rangle, \langle t_0, len(w_0)\rangle$

Table 1: POS tag feature templates. t_0 and w_0 denotes the POS tag and the word at the current position. t_{-x} and t_x denote left and right context tags, and similarly for words. $s(\cdot)$ denotes the score of the POS tag produced by the preprocessing tagger. The last row shows the "Character"-based features for Chinese. $pre_1(\cdot)$ and $pre_2(\cdot)$ denote the word prefixes with one and two characters respectively. $suf_1(\cdot)$ and $suf_2(\cdot)$ denote the word suffixes similarly. $c_n(\cdot)$ denotes the n-th character in the word. $len(\cdot)$ denotes the length of the word, capped at 5 if longer.

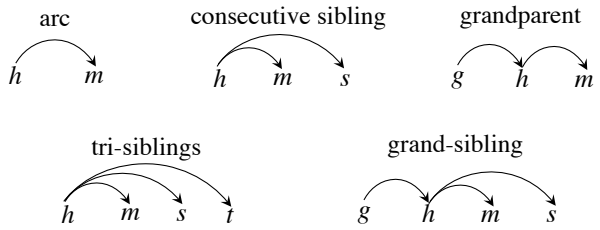


Figure 4: First- to third-order dependency parsing features.

ure 4 shows the first- to third-order feature templates that we use in our model. We also use global features to capture the adjacent conjuncts agreement in a coordination structure, and the valency patterns for each POS category. Note that most dependency features are implicitly cross-task in that they include POS tag and segmentation information. For example, the standard feature involves the POS tags of the words on both ends of the arc.

5 Experimental Setup

5.1 Datasets

We evaluate our model on two Arabic datasets and one Chinese dataset. For the first Arabic dataset, we use the dataset used in the Statistical Parsing of Morphologically Rich Languages (SPMRL) Shared Task 2013 (Seddah et al., 2013).[5] We follow the official split for training, development and testing set. We use the core set of 12 POS categories provided by Marton et al. (2013). In the second Arabic dataset, the training set is a dependency conversion of the Arabic Treebank, which primarily includes Modern Standard Arabic (MSA) text. However, we test on a new corpus, which consists of classical Arabic text obtained from the Comprehensive Islamic Library (CIS).[6] A native Arabic speaker with background in computational linguistics annotated the morphological segmentation and POS tags. This corpus is an excellent testbed for a joint model because classical Arabic may use rather different vocabulary from MSA, while their syntactic grammars are very similar to each other. Therefore incorporating syntactic information should be particularly beneficial to morphological segmentation and POS tagging. For Chinese, we use the Chinese Penn Treebank 5.0 (CTB5) and follow the split in previous work (Zhang and Clark, 2010).

Dataset		SPMRL	Classical	CTB5
Language		Arabic	Arabic	Chinese
Train	#sent	14.4k	15.4k	17.5k
	#token	451k	573k	442k
Dev.	#sent	1.8k	–	348
	#token	56.9k	–	6.6k
Test.	#sent	1.8k	163	348
	#token	55.6k	7.9k	8.0k

Table 2: Statistics of datasets.

Table 2 summarizes the statistics of the datasets. For the SPMRL test set, we follow the common practice which limits the sentence lengths up to 70 (Seddah et al., 2013). For classical Arabic and Chinese, we evaluate on all the test sentences.

5.2 Generating Lattice Structures

In this section we introduce the methodology for constructing candidate sets for segmentation and

[5]This dataset is originally provided by the LDC (Maamouri et al., 2004), specifically its SPMRL 2013 dependency instance, derived from the Columbia Catib Treebank (Habash and Roth, 2009; Habash et al., 2009) and extended according to the SPMRL 2013 extension scheme (Seddah et al., 2013).

[6]This classical Arabic dataset is publicly available at `http://farasa.qcri.org/`

MADA analysis

Word *Emlyp*
Emly/NOUN+*p*/NSUFF, gen:f/num:s/per:na
Emly/ADJ+*p*/NSUFF, gen:f/num:s/per:na
Eml/NOUN+*y*/NSUFF+*p*/PRON, gen:m/num:d/per:na

Lattice structure

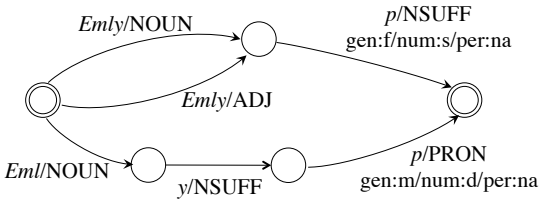


Figure 5: Example MADA analysis for the word *Emlyp* and the corresponding lattice structure.

POS tagging. Table 3 provides statistics on the generated candidate sets.

SPMRL 2013 Following Marton et al. (2013), we use the MADA system to generate candidate morphological analyses and POS tags. For each token in the sentence, MADA provides a list of possible morphological analyses and POS tags, each associated with a score. The score of each segmentation or POS tag equals the highest score of the MADA analysis in which it appears. In addition, we associate each segmentation with MADA analyses on gender, number and person. Figure 5 shows an example of MADA output for the token *Emlyp* and the corresponding lattice structure.

Classical Arabic We construct the lattice for this corpus in a similar fashion to the SPMRL dataset with two main departures. First, we use the Arabic morphological analyzer developed by Darwish et al. (2014) because MADA is primarily trained for MSA and performs poorly on classical Arabic. Second, we implement a CRF-based morpheme-level POS tagger and generate the POS tag candidates for each morpheme based on their marginal probabilities, truncated by a probability threshold.

CTB5 We first re-train the Stanford Chinese word segmenter on CTB5 and generate a top-10 list for each sentence.[7] We treat the word boundaries shared across all the 10 candidates as the confident ones, and construct the lattice as described in Section 3.4. Our model then focuses on disambiguating the rest of the word boundaries in the candidates. To generate POS candidates, we apply a CRF-based tagger with Chinese-specific features used in previous work (Hatori et al., 2011).

[7] We use 10-fold cross validation to avoid overfitting on the training set.

Dataset	Seg			POS	
	F1	Oracle	Avg. $\lvert\mathcal{S}_i\rvert$	F1	Avg. $\lvert\mathcal{T}_{i,j}\rvert$
SPMRL	99.4	99.8	1.23	96.9	1.71
Classical	92.4	97.0	1.16	82.4	3.01
CTB5	95.3	99.0	1.22	91.4	2.02

Table 3: Quality of the lattice structures on each dataset. For SPMRL and CTB5, we show the statistics on the development sets. For classical Arabic, we directly show the statistics on the testing set because the development set is not available.

5.3 Evaluation Measures

Following standard practice in previous work (Hatori et al., 2012; Zhang et al., 2014a), we use F-score as the evaluation metric for segmentation, POS tagging and dependency parsing. We report the morpheme-level F-score for Arabic and the word-level F-score for Chinese. In addition, we use TedEval (Tsarfaty et al., 2012) to evaluate the joint prediction on the SPMRL dataset, because TedEval score is the only evaluation metric used in the official report. We directly use the evaluation tools provided on the SPMRL official website.[8]

5.4 Baselines

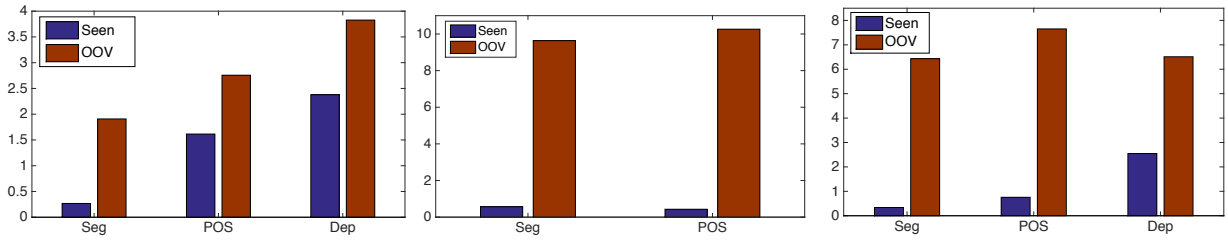
State-of-the-Art Systems For the SPMRL dataset, we directly compare with Björkelund et al. (2013). This system achieves the best TedEval score in the track of dependency parsing with predicted information and we directly republish the official result. We also compute the F-score of this system on each task using our own evaluation script.[9] For the CTB5 dataset, we directly compare to the arc-eager system by Zhang et al. (2014a), which slightly outperforms the arc-standard system by Hatori et al. (2012).

[8] http://www.spmrl.org/spmrl2013-sharedtask.html

[9] F-score evaluation for Arabic is not straightforward due to the stem changes in the morphological analysis. Therefore, the comparison of F-scores is only approximate.

Model	SPMRL				Classical Arabic		CTB5		
	Seg	POS	Dep	TedEval	Seg	POS	Seg	POS	Dep
Pipeline	99.18	95.76	84.79	92.86	92.37	82.40	97.45	93.42	79.46
Joint	**99.52**	**97.43**	**87.23**	**93.87**	**94.35**	**84.44**	**98.04**	**94.47**	**82.01**
Best Published	96.42	91.66	82.41	91.74	–	–	97.76	94.36	81.70

Table 4: Segmentation, POS tagging and unlabeled attachment dependency F-scores (%) and TedEval score (%) on different datasets. The first line denotes the performance by the pipeline variation of our model. The second row shows the results by our joint model. "Best Published" includes the best reported results: Björkelund et al. (2013) for the SPMRL 2013 shared task and Zhang et al. (2014a) for the CTB5 dataset. Note that the POS F-scores are not directly comparable because Björkelund et al. (2013) use a different POS tagset from us.



(a) SPMRL (b) Classical Arabic (c) CTB5

Figure 6: Absolute F-score (%) improvement of the joint model over the pipeline counterpart on seen and out-of-vocabulary (OOV) words.

System Variants We also compare against a pipeline variation of our model. In our pipeline model, we predict segmentations and POS tags by the same system that we use to generate candidates. The subsequent standard parsing model then operates on the predicted segmentations and POS tags.

5.5 Experimental Details

Following our earlier work (Zhang et al., 2014b), we train a first-order classifier to prune the dependency tree space.[10] Following common practice, we average parameters over all iterations after training with passive-aggressive online learning algorithm (Crammer et al., 2006; Collins, 2002). We use the same adaptive random restart strategy as in our earlier work (Zhang et al., 2014b) and set $K = 300$. In addition, we also apply an aggressive early-stop strategy during training for efficiency. If we have found a violation against the ground truth during the first 50 iterations, we immediately stop and update the parameters based on the current violation. The reasoning behind this early-stop strategy is that weaker violations for some training sentences are already sufficient for separable training sets (Huang et al., 2012).

6 Results

Comparison to State-of-the-art Systems Table 4 summarizes the performance of our model and the best published results for the SPMRL and the CTB5 datasets.[11] On both datasets, our system outperforms the baselines. On the SPMRL 2013 shared task, our approach yields a 2.1% TedEval score gain over the top performing system (Björkelund et al., 2013). We also improve the segmentation and dependency F-scores by 3.1% and 4.8% respectively. Note that the POS F-scores are not directly comparable because Björkelund et al. (2013) use a different POS tagset from us. On the CTB5 dataset, we outperform the state-of-the-art with respect to all

[10] We set the probability threshold to 0.05 and limit the number of candidate heads up to 20, which gives a 99.5% pruning recall on both the SPMRL and the CTB5 development sets.

[11] We are not aware of any published results on the Classical Arabic Dataset.

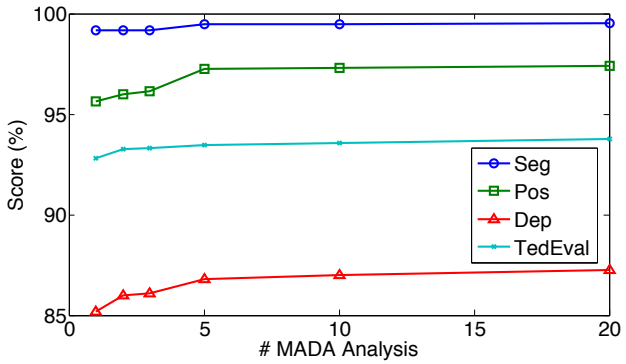


Figure 7: Performance with different sizes of the candidate sets on the SPMRL dataset. The graph shows the TedEval and F-scores when considering the best k analyses by MADA, and the variation is achieved by changing k.

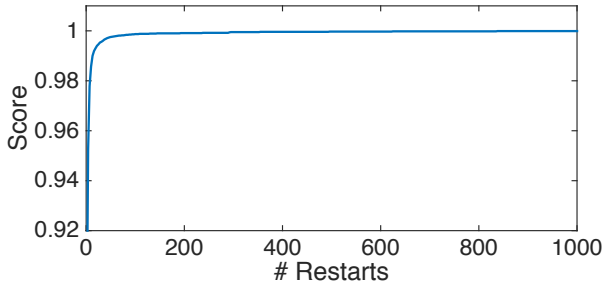


Figure 8: The normalized score of the output tree as the function of the number of restarts. We normalize scores of each sentence by the highest score among 3,000 restarts for this sentence. We show the curve up to 1,000 restarts because it reaches convergence after 500 restarts.

tasks: segmentation (0.3%), tagging (0.1%), and dependency parsing (0.3%).[12]

Impact of the Joint Prediction As Table 4 shows, our joint prediction model consistently outperforms the corresponding pipeline model in all three tasks. This observation is consistent with findings in previous work (Hatori et al., 2012; Tratz, 2013). We also observe that gains are higher (2%) on the classical Arabic dataset, which demonstrates that joint prediction is particularly helpful in bridging the gap between MSA and classical Arabic.

[12]Zhang et al. (2014a) improve the dependency F-score to 82.14% by adding manually annotated intra-word dependency information. Even without such gold word structure annotations, our model still achieves a comparable result.

Dataset	Seg		POS		Dep	
	Seen	OOV	Seen	OOV	Seen	OOV
SPMRL	48.4	27.8	44.7	15.0	15.9	17.5
Classical	13.8	34.8	4.2	17.2	–	–
CTB5	20.3	25.7	14.2	19.9	13.0	15.6

Table 5: F-score error reductions (%) of the joint model over the pipeline counterpart on seen and OOV words.

Figure 6 shows the break of the improvement based on seen and out-of-vocabulary (OOV) words. As expected, across all languages OOV words benefit more from the joint prediction, as they constitute a common source of error propagation in a pipeline model. The extent of improvement depends on the underlying accuracy of the preprocessing for segmentation and POS tagging on OOV words. For instance, we observe a higher gain (7%) on Chinese OOV words which have a 61.5% accuracy when processed by the original stand-along POS tagger. On the SPMRL dataset, the gain on OOV words is lower (3%), while preprocessing accuracy is higher (82%). Their error reductions on OOV words are nevertheless close to each other. Table 5 summarizes the results on F-score error reduction.

We also observe that the error reductions of OOV words/morphemes on the Chinese and the Classical Arabic dataset are larger than that of the in-vocabulary counterparts (e.g. 26% vs. 20% on Chinese word segmentation). However, we have the opposite observation on the segmentation and POS tagging on the SPMRL dataset (28% vs. 48%). This can be explained by analyzing the oracle performance in which we select the best solution from possible candidates. The oracle error reduction of OOV morphemes in the SPMRL dataset is relatively low (44%), compared to the 61% oracle error reduction of OOV morphemes in the Classical Arabic dataset.

Impact of the Number of Alternative Analyses In Figure 7, we plot the performance on the SPMRL dataset as a function of the number k of MADA analyses that we use to construct the candidate sets. For low k, increasing the number of analyses improves performance across all evaluation metrics. However, the performance converges at around $k = 15$.

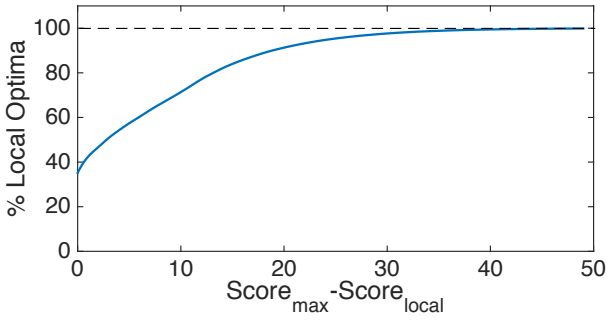


Figure 9: Cumulative distribution function (CDF) for the number of local optima versus the score of these local optima obtained from each restart, on the SPMRL dataset. The score captures the difference between a local optimum and the best one among 3,000 restarts.

Convergence Properties To assess the quality of the approximation obtained by the randomized greedy inference, we would like to compare it against the optimal solution. Following our earlier work (Zhang et al., 2014b), we use the highest score among 3,000 restarts for each sentence as a proxy for the optimal solution. Figure 8 shows the normalized score of the retrieved solution as a function of the number of restarts. We observe that most sentences converge quickly.[13] Specifically, more than 97% of the sentences converge within first 300 restarts. Since for the vast majority of cases our system converges fast, we achieve a comparable speed to that of other state-of-the-art joint systems. For example, our model achieves high performance on Chinese at about 0.5 sentences per second. The speed is about the same as that of the transition-based system (Hatori et al., 2012) with beam size 64, the setting that achieved best accuracy in their work.

Quality of Local Optima Figure 9 shows the cumulative distribution function (CDF) for the number of local optima versus the score of these local optima obtained from each restart. More specifically, the score captures the difference between a local optimum and the maximal score among 3,000 restarts. We can see that most of the local optima reached by hill-climbing have scores close to the maximum. For instance, about 30% of the local optima are identical to the best solution, namely $score_{max} - score_{local} = 0$.

[13] As expected, we also observe that convergence is slower when comparing to standard dependency parsing with a similar randomized greedy algorithm (Zhang et al., 2014b), because joint prediction results in a harder inference problem.

7 Conclusions

In this paper, we propose a general randomized greedy algorithm for joint segmentation, POS tagging and dependency parsing. On both Arabic and Chinese, our model achieves improvement on the three tasks over state-of-the-art systems and pipeline variants of our system. In particular, we demonstrate that OOV words benefits more from the power of joint prediction. Finally, our experimental results show that increasing candidate sizes improves performance across all evaluation metrics.

Acknowledgments

This research is developed in a collaboration of MIT with the Arabic Language Technologies (ALT) group at Qatar Computing Research Institute (QCRI) within the Interactive sYstems for Answer Search (IYAS) project. The authors acknowledge the support of the U.S. Army Research Office under grant number W911NF-10-1-0533, and of the DARPA BOLT program. We thank Meishan Zhang and Anders Björkelund for answering questions and sharing the outputs of their systems. We also thank the MIT NLP group and the ACL reviewers for their comments. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors, and do not necessarily reflect the views of the funding organizations.

References

Anders Björkelund, Ozlem Cetinoglu, Richárd Farkas, Thomas Mueller, and Wolfgang Seeker. 2013. (re)ranking meets morphosyntax: State-of-the-art results from the SPMRL 2013 shared task. In *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 135–145, Seattle, Washington, USA, October. Association for Computational Linguistics.

Bernd Bohnet and Joakim Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computa-*

tional Natural Language Learning, pages 1455–1465. Association for Computational Linguistics.

Bernd Bohnet, Joakim Nivre, Igor Boguslavsky, Richárd Farkas, Filip Ginter, and Jan Hajic. 2013. Joint morphological and syntactic analysis for richly inflected languages. *TACL*, 1:415–428.

Shay B Cohen and Noah A Smith. 2007. Joint morphological and syntactic disambiguation. In *Proceedings of EMNLP*.

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing - Volume 10*, EMNLP '02. Association for Computational Linguistics.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *The Journal of Machine Learning Research*.

Kareem Darwish, Ahmed Abdelali, and Hamdy Mubarak. 2014. Using stem-templates to improve arabic pos and gender/number tagging. In *International Conference on Language Resources and Evaluation (LREC-2014)*.

Souhir Gahbiche-Braham, Hélene Bonneau-Maynard, Thomas Lavergne, and François Yvon. 2012. Joint segmentation and pos tagging for arabic using a crf-based classifier. In *LREC*, pages 2107–2113.

Yoav Goldberg and Michael Elhadad. 2011. Joint hebrew segmentation and parsing using a pcfg-la lattice parser. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 704–709. Association for Computational Linguistics.

Yoav Goldberg and Reut Tsarfaty. 2008. A single generative model for joint morphological segmentation and syntactic parsing. In *ACL*, pages 371–379. Citeseer.

Nizar Habash and Owen Rambow. 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*, pages 573–580. Association for Computational Linguistics.

Nizar Habash and Ryan Roth. 2009. Catib: The columbia arabic treebank. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 221–224, Suntec, Singapore, August. Association for Computational Linguistics.

Nizar Habash, Reem Faraj, and Ryan Roth. 2009. Syntactic Annotation in the Columbia Arabic Treebank. In *Proceedings of MEDAR International Conference on Arabic Language Resources and Tools*, Cairo, Egypt.

Jun Hatori, Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2011. Incremental joint pos tagging and dependency parsing in chinese. In *IJCNLP*, pages 1216–1224. Citeseer.

Jun Hatori, Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2012. Incremental joint approach to word segmentation, pos tagging, and dependency parsing in chinese. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 1045–1053. Association for Computational Linguistics.

Liang Huang, Suphan Fayong, and Yang Guo. 2012. Structured perceptron with inexact search. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–151. Association for Computational Linguistics.

Zhenghua Li, Min Zhang, Wanxiang Che, Ting Liu, Wenliang Chen, and Haizhou Li. 2011. Joint models for chinese pos tagging and dependency parsing. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1180–1191. Association for Computational Linguistics, July.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, and Wigdan Mekki. 2004. The Penn Arabic Treebank: Building a Large-Scale Annotated Arabic Corpus. In *NEMLAR Conference on Arabic Language Resources and Tools*.

Yuval Marton, Nizar Habash, Owen Rambow, and Sarah Alkhulani. 2013. Spmrl'13 shared task system: The cadim arabic dependency parser. In *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 76–80.

Xian Qian and Yang Liu. 2012. Joint chinese word segmentation, pos tagging and parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 501–511. Association for Computational Linguistics.

Djamé Seddah, Reut Tsarfaty, Sandra Kübler, Marie Candito, Jinho D. Choi, Richárd Farkas, Jennifer Foster, Iakes Goenaga, Koldo Gojenola Galletebeitia, Yoav Goldberg, Spence Green, Nizar Habash, Marco Kuhlmann, Wolfgang Maier, Joakim Nivre, Adam Przepiórkowski, Ryan Roth, Wolfgang Seeker, Yannick Versley, Veronika Vincze, Marcin Woliński, Alina Wróblewska, and Eric Villemonte de la Clergerie. 2013. Overview of the SPMRL 2013 shared task: A cross-framework evaluation of parsing morphologically rich languages. In *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 146–182, Seattle, Washington, USA, October. Association for Computational Linguistics.

Stephen Tratz. 2013. A cross-task flexible transition model for arabic tokenization, affix detection, affix labeling, pos tagging, and dependency parsing. In *Fourth Workshop on Statistical Parsing of Morphologically Rich Languages*, page 34. Citeseer.

Reut Tsarfaty, Joakim Nivre, and Evelina Andersson. 2012. Joint evaluation of morphological segmentation and syntactic parsing. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 6–10. Association for Computational Linguistics.

Zhiguo Wang and Nianwen Xue. 2014. Joint pos tagging and transition-based constituent parsing in chinese with non-local features. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 733–742, Baltimore, Maryland, June. Association for Computational Linguistics.

David Wilson. 1996. Generating random spanning trees more quickly than the cover time. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 296–303. ACM.

Yue Zhang and Stephen Clark. 2008. Joint word segmentation and pos tagging using a single perceptron. In *ACL*, pages 888–896.

Yue Zhang and Stephen Clark. 2010. A fast decoder for joint word segmentation and pos-tagging using a single discriminative model. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 843–852. Association for Computational Linguistics.

Meishan Zhang, Yue Zhang, Wanxiang Che, and Ting Liu. 2014a. Character-level chinese dependency parsing. In *ACL*.

Yuan Zhang, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2014b. Greed is good if randomized: New inference for dependency parsing. In *EMNLP*.

Yuan Zhang, Tao Lei, Regina Barzilay, Tommi Jaakkola, and Amir Globerson. 2014c. Steps to excellence: Simple inference with refined scoring of dependency trees. In *ACL*.