Fast Unsupervised Dependency Parsing with Arc-Standard Transitions

Mohammad Sadegh Rasooli
Department of Computer Engineering
Iran University of Science and Technology
Narmak, Tehran, Iran
rasooli@comp.iust.ac.ir
rasooli.ms@gmail.com

Heshaam Faili
School of Electrical
and Computer Engineering
University of Tehran
Amir-Abaad, Tehran, Iran
hfaili@ut.ac.ir

Abstract

Unsupervised dependency parsing is one of the most challenging tasks in natural languages processing. The task involves finding the best possible dependency trees from raw sentences without getting any aid from annotated data. In this paper, we illustrate that by applying a supervised incremental parsing model to unsupervised parsing; parsing with a linear time complexity will be faster than the other methods. With only 15 training iterations with linear time complexity, we gain results comparable to those of other state of the art methods. By employing two simple universal linguistic rules inspired from the classical dependency grammar, we improve the results in some languages and get the state of the art results. We also test our model on a part of the ongoing Persian dependency treebank. This work is the first work done on the Persian language.

1 Introduction

Unsupervised learning of grammars has achieved considerable focus in recent years. The lack of sufficient manually tagged linguistic data and the considerable successes of unsupervised approaches on some languages have motivated researchers to test different models of unsupervised learning on different linguistic representations.

Since the introduction of the dependency model with valence (DMV) proposed by Klein and Manning (2004), dependency grammar induction has received great attention by researchers. DMV was the first model to outperform the right attachment accuracy in English. Since this achievement, the model has been used by many researchers (e.g. (Cohen and Smith, 2010); (Gillenwater et al., 2011); (Headden III et al., 2009); and (Spitkovsky et al., 2011b)).

The main task of unsupervised dependency parsing is to obtain the most likely dependency tree of a sentence without using any annotated training data. In dependency trees, each word has only one head and the head of the sentence is a dependent of an artificial root word. Problems such as data sparsity and a large search space that increases the ambiguity have made the task difficult. Even deciding the direction of the link between two words in a dependency relation has made the task more difficult than finding phrase structures themselves (Klein and Manning, 2004).

In this paper, we propose a model based on Arc-Standard Transition System of Nivre (2004), which is known as an incremental greedy projective parsing model that parses sentences in linear time. To the best of our knowledge, the only incremental unsupervised dependency parsing is the model of Daumé III (2009) with Shift-Reduce parsing model (Nivre, 2003).[1]

Our model is not lexicalized, has a simple feature space and converges in 15 iterations with a linear ($O(n)$) parsing and training time, while other methods based on DMV in the best case work in $O(n^3)$ time complexity with $O(n^3)$ memory use for sentences with of length n. We believe that the output of this model can also improve DMV.[2] In addition, we use punctuation clues (Spitkovsky et al., 2011c), tying feature similarity in the transition system configuration, and

[1] The other study is in Seginer (2007) that is for constituency parsing (phrase structure extraction).

[2] For the effect of model initialization in unsupervised dependency parsing, see Gimpel and Smith (2011).

Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics, pages 1–9,
Avignon, France, April 23 - 27 2012.

"baby steps" notion (Spitkovsky et al., 2009) to improve the model accuracy.

We test our model on 9 CoNLL 2006 and 2007 shared task data sets (Buchholz and Marsi, 2006; Nivre et al., 2007) and WSJ part of Penn treebank and show that in some languages our model is better than the recent models. We also test our model on a part of an ongoing first Persian dependency corpus (Rasooli et al., 2011). Our study may be the first work to test dependency parsing on the Persian language.

The remainder of this paper is organized as follows. In Section 2, related work on unsupervised dependency parsing is reviewed. In Section 3, we describe our dependency parsing model. In Section 4 and Section 5, after the reporting experimental results on several languages, the conclusion is made.

2 Related Work

The first considerable work on unsupervised dependency parsing which outperforms the baseline (right attachment) accuracy in English was proposed by Klein and Manning (2004). The model is called dependency model with valence (DMV). In the DMV, each word can be the head of the sentence with the probability of $P(root|X)$. Each word X, decides to get a child Y from a direction (right or left), with the probability $P_{CHOOSE}(X|Y, dir, adj)$, where adj is a Boolean value indicating whether the word has gotten a child in the direction dir or not. The other probability used in the DMV is $P_{STOP}(X|Y, dir, adj)$ that means whether to stop getting dependents from the direction with adjacency value or not. All the probabilities in the model are assumed to be independent and the dependency tree likelihood is a product of all probabilities. Only part of speech (POS) tags are used as features and the probabilities are multinomial. The model uses the inside-outside algorithm to find all possible subtrees efficiently in Expectation Maximization (EM) algorithm.

Several researchers have tried to improve and modify the DMV. In Headden III et al. (2009), by using the lexical values with the frequency more than 100 and defining tied probabilistic context free grammar (PCFG) and Dirichlet priors, the accuracy is improved. In Smith and Eisner (2005), by producing artificial neighbors of the feature space via actions such as deletion of one word, substitution of adjacent words and adding a word, the likelihood of the true feature space in all neighbors is calculated. That method is known as contrastive estimation (CE).

In Spitkovsky et al. (2009), the idea of learning the initial parameters of the model from shorter sentences leads to a method named "baby steps". In "baby steps", the model prior of each training set with the sentence length less than or equal to N, is achieved by training DMV on the training set with the sentence length less than or equal to $N - 1$. The other method used in the mentioned work, is "less is more" which hypothesize that training on a subset of all data (with the length of less than or equal to 15) in batch mode is more useful than training on all data. In Spitkovsky et al. (2010a), a combination of "baby steps" and "less is more", named "leapfrog" is applied to the DMV. In Spitkovsky et al. (2011b), a mixture of EMs is used to improve the DMV by trying to escape from local maxima; i.e., changing the EM policy in some iterations in order to escape from local maxima. The model is termed "lateen" EM. In Spitkovsky et al. (2010b), HTML hypertext tags are used as indicators of phrases in order to localize the search space of the dependency model. In Spitkovsky et al. (2011c), punctuation marks are used as indicators of local dependencies of the words in the sentence.

In Cohen and Smith (2010), shared logistic normal distribution is used to tie grammatical roles that are not assumed to be independent from each other. In the study, the bilingual similarity of each POS tag probability in the dependency model is applied to the probability model. In Blunsom and Cohn (2010), Pitman-Yor priors (PYP) are applied to the DMV. Furthermore, tree substitution grammar (TSG) is used as an intermediate representation of the tree. In Gillenwater et al. (2011), a mathematical model is employed to overcome the posterior sparsity in the DMV, by defining constraints on the probability model.

There are also some models different from DMV. In Daumé III (2009), based on a stochastic search method, Shift-Reduce transition parsing model of Nivre (2003) is applied. The model is greedy and selects an action stochastically according to each action probability at the time. The advantage of the model lies on its parsing and training speed. In Naseem and Barzilay (2011), sparse semantic annotations in texts are used as

Initialization	$\langle nil, W, \Phi \rangle$
Termination	$\langle S, nil, A \rangle$
Left-Reduce	$\langle w_i w_j \vert S, I, A \rangle \rightarrow \langle w_j \vert S, I, A \cup \langle w_j, w_i \rangle \rangle$
Right-Reduce	$\langle w_i w_j \vert S, I, A \rangle \rightarrow \langle w_i \vert S, I, A \cup \langle w_i, w_j \rangle \rangle$
Shift	$\langle S, w_i \vert I, A \rangle \rightarrow \langle w_i \vert S, I, A \rangle$

Figure 1: Actions in Arc-Standard Transition System (Nivre, 2004)

clues to unsupervised parsing. In Mareček and Žabokrtský (2011), by applying Gibbs sampling method to count the data occurrences, a simple probability model (the fraction of each dependency relation divided by the number of head POS tags) is used. In that model, non-projective dependency trees are allowed and all noun-root dependency probabilities are multiplied by a small number, to decrease the chance of choosing a noun-root dependency. There are also some studies in which labeled data in one language is employed to guide unsupervised parsing in the others (Cohen et al., 2011).

3 Fast Unsupervised Parsing

In this section, after a brief description of the Arc-Standard parsing model, our probability model, and the unsupervised search-based structure prediction (Daumé III, 2009) are reviewed. After these descriptions, we go through "baby steps," the use of curricula in unsupervised learning (Tu and Honavar, 2011), and the use of punctuation in unsupervised parsing. Finally, we describe our tied feature model that tries to overcome the data sparsity. In this paper, a mixture of "baby steps" and punctuation clues along with search-based structure prediction is applied to the Arc-Standard model.

3.1 Arc-Standard Transition Model

The parser in this model has a configuration represented by $\langle S, I, A \rangle$, where S is a stack of words, I is a buffer of input words which are not processed yet and A is the list of all arcs that are made until now. The parser initializes with $\langle nil, W, \phi \rangle$ in which W is a string of all words in the sentence, nil shows a stack with a root word and Φ shows an empty set. The termination configuration is shown as $\langle S, nil, A \rangle$, where S shows an empty stack with only root word, nil shows an empty buffer and A is the full arc set. An arc in which w_j is the head of w_i is shown by $w_j \rightarrow w_i$ or (w_j, w_i).

As shown in Figure 1, there are three actions in this model. In the shift action, the top-most input word goes to the top of the stack. In the left-reduce action, the top-most stack word becomes the head of the second item in the stack and the second item is removed from the stack. On the other hand, in the right-reduce action, the second word in the stack becomes the head of the top item in the stack and the top item is removed from the stack.

3.2 Feature Space and Probability Model

The feature space that we use in this model is a tuple of three POS tags; i.e., the first item in the buffer, the top-most and the second item in the stack. The probability of each action is inspired from Chelba and Jelinek (2000) as in equation (1). In each step in the configuration, the parser chooses an action based on the probability in equation (1), where $feat$ is the feature value and act is an action.

$$P(act, feat) = P(act) \cdot P(feat|act) \qquad (1)$$

The action selection in the training phase is done stochastically. In other words, in every step there is a maximum of 3 actions and a minimum of one action.[3] After calculating all probabilities, a roulette wheel is made to do multinomial sampling. The sampling is done with stochastic EM (Celeux and Diebolt, 1985) in a roulette wheel selection model.

The probabilities are initialized equally (except that $P(shift) = 0.5$ and $P(right - reduce) = P(left - reduce) = 0.25$). After sampling from the data, we update the model as in equations 2–4, where σ is a smoothing variable and N_f is the number of all possible unique features in the data set. $C(\cdot)$ is a function that counts the data from

[3]For example, in the first state only shift is possible and in the last state only right-reduce is possible.

samples. In equations 3 and 4, sh, $r - r$ and $l-r$ are shift, right-arc and left-arc actions respectively. $C(Action, Feature)$ is obtained from the samples drawn in the training phase. For example, if the right-reduce action is selected, we add its probability to $C(right - reduce, feature)$.

$$P(feat|act) = \frac{C(act, feat) + \sigma}{C(act) + N_f \sigma} \quad (2)$$

$$P(sh) = 0.5 \quad (3)$$

$$P(act) = \frac{C(act) + \sigma}{C(r - r) + C(l - r) + 2\sigma}; \quad act \neq Shift \quad (4)$$

3.3 Unsupervised Search-Based Structure Prediction

Since there are 3^{2n+1} possible actions for a sentence with the length of n it seems impractical to track the search space for even middle-length sentences. Accordingly, in Daumé III (2009) a stochastic search model is designed to improve the model accuracy based on random actions. In that work, with each configuration step, the trainer selects one of the actions according to the probability of each action stochastically. By choosing actions stochastically, a set of samples is drawn from the data and the model parameters are updated based on the pseudo-code in Figure 2. In Figure 2, π is known as the policy of the probability model and β is a constant number which changes in each iteration based on the iteration number ($\beta = \frac{1}{iteration\#^3}$). We employ this model in our work to learn probability values in equation 1. The learning from samples is done via equations (2–4).

3.4 "Baby Steps" Incremental Parsing

In Spitkovsky et al (2009), the idea that shorter sentences are less ambiguous, hence more informative, is applied to the task. Spitkovsky et al. (2009) emphasize that starting from sentences with a length of 1 and iterating on sentences with the $length \leq N$ from the probabilities gained from the sentences with the $length \leq N - 1$, leads to better results.

```
Initialize π = π*
while not converge
  Take samples stochastically
  h ← learn from samples
  π = βπ + (1 − β)h
end while
return π
```

Figure 2: Pseudo-code of the search-based structure prediction model in Daumé III (2009)

We also use "baby steps" on our incremental model. For the sentences having length 1 through 5, we only iterate once in each sentence length. At those sentence lengths, the full search space is explored (all trees are made by doing all possible actions in each state of all possible configurations), while for sentence length 6 towards 15, we iterate at each step 3 times, only choosing one action stochastically at each state. The procedure is done similarly for all languages with the same parameters. In fact, the greedy nature of the model encourages us to bail out of each sentence length quickly. In other words, we want to jump out of early local maxima, as in early-terminating lateen EM (Spitkovsky et al., 2011b).

In curricula (Tu and Honavar, 2011), smoothing variable for shorter sentences is larger than smoothing variable for longer sentences. With regards to the idea, we start with smoothing variable equal to 1 and multiply it on each sentence length by a constant value equal to e^{-1}.

3.5 Punctuation Clues

Spitkovsky et al. (2011c) show that about 74.0% of words in English texts occurring between two punctuation marks have only one word linking with other words of the sentence. This characteristic is known as "loose". We apply this restriction on our model to improve the parsing accuracy and decrease the total search space. We show that this clue not only does not improve the dependency parsing accuracy, but also decreases it in some occasions.

3.6 Tying Probabilities with Feature Similarity Measure

We assume that the most important features in the feature set for right-reduce and left-reduce actions are the two top words in the stack. On the other hand, for the shift action, the most impor-

tant words are first buffer and top stack words. In order to solve the sparsity problem, we modify the probability of each action based on equation (5). In this equation, $neigh(act, feat)$ is gained via searching over all features with the same top and second stack item for left-reduce and right-reduce, and all features with the same top stack and first buffer item for the shift action.

$$P'(feat|act) = \frac{P(feat|act) + \frac{\sum_{f' \in neigh(act,feat)} P(f'|act)}{C(neigh(act,feat))}}{2} \quad (5)$$

3.7 Universal Linguistic Heuristics to Improve Parsing Accuracy

Based on the nature of dependency grammar, we apply two heuristics. In the first heuristic, we multiply the probability of the last verb reduction by 10^{-10} in order to keep verbocentricity of the dependency grammar. The last verb reduction occurs when there is neither a verb in the buffer nor in the stack except the one that is going to be reduced by one of the right-arc or left-arc actions. In other words, the last verb remaining on the stack should be less likely to be removed than the other actions in the current configuration.[4] In the second heuristic, in addition to the first heuristic, we multiply each $noun \rightarrow verb$, $adjective \rightarrow verb$, and $adjective \rightarrow noun$ by 0.1 in order to keep the nature of dependency grammar in which nouns and adjective in most cases are not able to be the head of a verb and an adjective is not able to be the head of a noun.[5] We show in the experiments that, in most languages, considering this nature will help improve the parsing accuracy.

We have tested our model on 9 CoNLL data sets (Buchholz and Marsi, 2006; Nivre et al., 2007). The data sets include Arabic, Czech, Bulgarian, Danish, Dutch, Portuguese, Slovenian, Spanish, and Swedish. We have also tested our model on a part of the ongoing project of Persian dependency treebank. The data set includes 2,113 train and 235 test sentences.[6]

As shown in Figure 3, we use the same procedure as in Daumé III (2009), except that we restrict β to not be less than 0.005 in order to increase the chance of finding new search spaces stochastically. As in previous works, e.g., Smith and Eisner (2005), punctuation is removed for evaluation.

$iteration\# = 0$
for i=1 to 15 do
 Train-set=all sentences-length≤i
 max-iter=3
 if(i≤ 5)
  max-iter=1
 end-if
 for j=1 to max-iter do
  $\beta = max(\frac{1}{iteration\#^3}, 0.005)$
  $iteration\# \leftarrow iteration\# + 1$
  if($i \leq 5$)
   samples← find all subtrees
  end-if
  else
   samples← sample instances stochastically
  end-else
 h ← learn from samples
 $\pi = \beta\pi + (1 - \beta)h$
end-for
$\sigma = \sigma \times e^{-1}$
end-for

Figure 3: Pseudo-code of the unsupervised Arc-Standard training model

4 Evaluation Results

Although training is done on sentences of length less than 16, the test was done on all sentences in the test data without dropping any sentences form the test data. Results are shown in Table 1 on 9 languages. In Table 1, "h1" and "h2" refer to the two linguistic heuristics that are used in this paper. We also compare our work with Spitkovsky et al. (2011b) and Mareček and Žabokrtský (2011)

[4]It is important to note that the only reason that we choose a very small number is to decrease the chance of verb-reduction among three possible actions. Using other values≤ 0.01 does not change results significantly.

[5]The are some exceptions too. For example, in the sentence: "I am certain your work is good." Because of that, we do not choose a very small number.

[6]This dataset is obtained via contacting with the project team at http://www.dadegan.ir/en/. Recently an official pre-version of the dataset is released, consisting more than 12,000 annotated sentences (Dadegan Research Group, 2012). We wish to report results on the dataset in our future publications.

	Baselines			Using Heuristic 1 and 2			Using Heuristic 1		Without any heuristic	
Language	**Rand**	**LA**	**RA**	**fs+punc**	**fs**	**punc**	**fs+punc**	**punc**	**punc+fs**	**simp.**
Arabic'07	3.90	**59.00**	06.00	52.05	52.05	52.05	54.55	54.55	**55.64**	**55.64**
Bulgarian	8.00	38.80	17.90	52.48	**53.86**	46.36	42.75	37.35	35.99	35.99
Czech'07	7.40	29.60	24.20	42.37	**42.40**	39.31	30.21	27.94	25.17	25.17
Danish	6.70	47.80	13.10	52.14	**53.11**	52.14	51.10	51.70	46.01	46.01
Dutch	7.50	24.50	28.00	48.14	**48.80**	48.20	28.30	28.36	23.47	23.45
Persian	9.50	03.90	29.16	**51.65**	51.37	50.99	49.78	50.99	26.87	26.87
Portuguese	5.80	31.20	25.80	54.86	**55.84**	46.82	33.84	33.62	28.83	28.83
Slovenian	7.90	**26.60**	24.30	**22.44**	**22.44**	22.43	21.31	21.30	19.47	19.45
Spanish	4.30	29.80	24.70	30.88	31.16	30.88	**32.33**	**32.33**	29.63	29.69
Swedish	7.80	27.80	25.90	32.74	**34.33**	33.52	28.48	28.48	25.74	25.74
Turkish	6.40	01.50	**65.40**	33.83	27.39	38.13	**61.27**	47.92	30.56	34.52
Average	6.84	29.14	25.86	**43.05**	42.98	41.89	39.45	37.69	31.58	31.94

Table 1: Results tested on CoNLL data sets and the Persian data set. "Rand", "LA" and "RA" stand for random, left-attach and right-attach, respectively; "punc" refers to punctuation clues and fs refers to feature similarity cue; "all" refers to using both heuristics h1 and h2; and "simp." refers to the simple model.

in Table 2. As shown in Table 2, our model outperforms the accuracy in 7 out of 9 languages.

The Effect of Feature Similarity

As shown in Table 1, feature similarity cannot have any effect on the simple model. When we add linguistic information to the model, this feature similarity measure keeps the trainer from diverging. In other words, the greedy nature of the model becomes endangered when incomplete knowledge (as in our linguistic heuristics) is used. Incomplete knowledge may cause early divergence. In other words, the greedy algorithm tracks the knowledge which it has and does not consider other probable search areas. This phenomenon may cause early divergence in the model. By using feature similarity, we try to escape from this event.

The Effect of Punctuation Clues

As shown in Table 1, in most languages punctuation clues do not improve the accuracy. This maybe arises out of the fact that "loose" is not a good clue for incremental parsing. The other clue is "sprawl" in which the external link restriction is lifted. This restriction is in 92.9% of fragments in English texts (Spitkovsky et al., 2011c), but it is not implemented and tested in this paper.

4.1 Evaluation on English

We also test our data on Penn Treebank but we do not gain better results than state of the art methods. We use the same train and test set as in Spitkovsky et al. (2009). We convert the Penn treebank data via automatic "head-percolation" rules (Collins, 1999). We have also tested our model via simple stochastic EM (without using unsupervised structure prediction) and show that the main problem with this method in English is its fast divergence when jumping from sentence length 5 to 6. In the model settings tested for English, the model with heuristic 1 with the feature similarity is the best setting that we find. By testing with a smoothing variable ten times bigger

Model	WSJ'10	$WSJ'\infty$
h1+fs	45.16	31.97
h1+fs+punc	44.17	30.17
Stoch. EM(1-5)	40.86	33.65
Stoch. EM(1-5)+h1	52.70	42.85
Stoch. EM(1-5)+h1+h2	50.30	41.37
A1+fs+h1	49.9	43.3
Klein and Manning (2004)	43.2	-
Daumé III (2009)	45.4	-
Blunsom and Cohn (2010)	67.7	55.7
Spitkovsky et al. (2011a)	-	**59.1**

Table 3: Results of our model on WSJ, compared to its counterpart Daumé III (2009) and other DMV-based models. Since in Blunsom and Cohn (2010) and Spitkovsky et al. (2011b), other results are reported, we only limit our report to some of the results on WSJ. In the Table, "h1" shows heuristic 1 and "fs" shows the use of feature similarity. Stochastic EM(1-5) is one test that have done only by applying baby steps on sentences with the length 1 to 5 without using unsupervised search-based model. A1 refers to a change in the model in which smoothing variable in steps 1 to 5 is multiplied by 10.

Language \ Method Name	MZ-NR	MZ	Spi5	Spi6	Our Best
Arabic'07	24.8	25.0	22.0	49.5	**55.64**
Bulgarian	51.4	25.4	44.3	43.9	**53.86**
Czech'07	33.3	24.3	31.4	28.4	**42.40**
Danish	38.6	30.2	44.0	38.3	**53.11**
Dutch	43.4	32.2	32.5	27.8	**48.80**
Persian	-	-	-	-	**51.65**
Portuguese	41.8	43.2	34.4	36.7	**55.84**
Slovenian	**34.6**	25.4	33.6	32.2	22.44
Spanish	**54.6**	53.0	33.3	50.6	32.33
Swedish	26.9	23.3	42.5	**50.0**	34.33
Turkish	32.1	32.2	33.4	35.9	**61.27**
Average (except Persian)	38.1	31.4	35.1	39.3	**46.00**

Table 2: Comparison of our work to those of Table 5 ("Spi5") and Table 6 ("Spi6") in Spitkovsky et al. (2011b) and Mareček and Žabokrtský (2011) with Noun-Root constraint ("MZ-NR") and no constraint ("MZ"). The comparison results are from Table 4 in Mareček and Žabokrtský (2011). "Our Best" refers to the bold scores in Table 1.

in the first 5 steps, we have seen that the results change significantly. The results are shown in Table 3.

One main problem of the converted dependencies in English is their conversion errors like multi-root trees.[7] There are many trees in the corpus that have wrong multi-root dependencies. Such problems lead us to believe that we should not rely too much on the results on WSJ part of the Penn treebank.

5 Analysis and Conclusion

One main aspect of incremental methods is their similarity to the way that humans read and learn sentences in language. The interesting characteristic of incremental parsing lies on its speed and low memory use. In this paper, we use one new incremental parsing model on unsupervised parsing for the first time. The simple mathematical model and its linear training order has made the model flexible to be used for bigger data sets. In addition to testing recently used heuristics in unsupervised parsing, by inspiring basic dependency theory from linguistics, parsing accuracy has been increased in some languages. We see that this model is capable of detecting many true dependencies in many languages.

Some observations show that choosing inappropriate parameters for the model may lead to unwanted divergence in the model. The divergence is mostly seen in English, where we see a significant accuracy decrease at the last step in comparison to step 5 instead of seeing an increase in the accuracy. With one setting in English, we reach the accuracy equal to 43% (13% more than the accuracy of the model reported in this paper).

In some languages like Slovenian, we see that even with a good undirected accuracy, the model does not succeed in finding the dependency direction with heuristics. While in Czech, Dutch, and Bulgarian the second heuristic works well, it does not change accuracy a lot in other languages (in languages like Turkish and English this heuristic decreases accuracy). We believe that choosing better linguistic knowledge like the ones in Naseem et al. (2010), tying grammatical rules from other languages similar to the work in Cohen and Smith (2010), and choosing better probability models that can be enriched with lexical features and broad context consideration (like the works in supervised incremental dependency parsing) will help the model perform better on different languages.

Despite the fact that our study is the first work done on Persian, we believe that the results that we achieve for Persian is very considerable, regarding the free-word order nature of the Persian language.

[7] We assume to find only trees that are projective and single-rooted.

Acknowledgments

The paper is funded by Computer Research Center of Islamic Sciences (CRCIS). We would like to appreciate Valentin Spitkovsky and Shay Cohen for their technical comments on the research and paper draft. We would also thank Maryam Faal-Hamedanchi, Manouchehr Kouhestani, Amirsaeid Moloodi, Hamid Reza Ghader, Maryam Aminian and anonymous reviewers for their comments on the draft version. We would also like to thank Jason Eisner, Mark Johnson, Noah Smith and Joakim Nivre for their help in answering our questions and Saleh Ziaeinejad and Younes Sangsefidi for their help on the project.

References

Phil Blunsom and Trevor Cohn. 2010. Unsupervised induction of tree substitution grammars for dependency parsing. In *2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, pages 1204–1213.

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceeding of the Tenth Conference on Computational Natural Language Learning (CoNLL)*.

Gilles Celeux and Jean Diebolt. 1985. The SEM algorithm: A probabilistic teacher algorithm derived from the em algorithm for the mixture problem. *Computational Statistics Quarterly*, 2:73–82.

Ciprian Chelba and Frederick Jelinek. 2000. Structured language modeling. *Computer Speech & Language*, 14(4):283–332.

Shay B. Cohen and Noah A. Smith. 2010. Covariance in unsupervised learning of probabilistic grammars. *Journal of Machine Learning Research (JMLR)*, 11:3117–3151.

Shay B. Cohen, Dipanjan Das, and Noah A. Smith. 2011. Unsupervised structure prediction with Non-Parallel multilingual guidance. In *Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*.

Michael Collins. 1999. *Head-driven statistical models for natural language parsing*. Ph.D. thesis, University of Pennsylvania.

Dadegan Research Group. 2012. *Persian Dependency Treebank Version 0.1, Annotation Manual and User Guide*. `http://dadegan.ir/en/`.

Hal Daumé III, John Langford, and Daniel Marcu. 2009. Search-based structured prediction. *Machine Learning*, 75(3):297–325.

Hal Daumé III. 2009. Unsupervised search-based structured prediction. In *26th International Conference on Machine Learning (ICML)*, pages 209–216. ACM.

Jennifer Gillenwater, Kuzman Ganchev, João Graça, Fernando Pereira, and Ben Taskar. 2011. Posterior sparsity in unsupervised dependency parsing. *Journal of Machine Learning Research (JMLR)*, 12:455–490.

Kevin Gimpel and Noah A. Smith. 2011. Concavity and initialization for unsupervised dependency grammar induction. Technical report.

William P. Headden III, Mark Johnson, and David McClosky. 2009. Improving unsupervised dependency parsing with richer contexts and smoothing. In *Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the ACL*, pages 101–109.

Dan Klein and Christopher D. Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Association for Computational Linguistics (ACL)*.

David Mareček and Zdeněk Žabokrtský. 2011. Gibbs sampling with treeness constraint in unsupervised dependency parsing. In *RANLP Workshop on Robust Unsupervised and Semisupervised Methods in Natural Language Processing*.

Tahira Naseem and Regina Barzilay. 2011. Using semantic cues to learn syntax. In *25th Conference on Artificial Intelligence (AAAI-11)*.

Tahira Naseem, Harr Chen, Regina Barzilay, and Mark Johnson. 2010. Using universal linguistic knowledge to guide grammar induction. In *2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceeding of CoNLL 2007*.

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *International Workshop on Parsing Technologies*, pages 149–160.

Joakim Nivre. 2004. Incrementality in deterministic dependency parsing. In *Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, pages 50–57.

Mohammad Sadegh Rasooli, Amirsaeid Moloodi, Manouchehr Kouhestani, and Behrouz Minaei-Bidgoli. 2011. A syntactic valency lexicon for Persian verbs: The first steps towards Persian dependency treebank. In *5th Language & Technology Conference (LTC): Human Language Technologies as a Challenge for Computer Science and Linguistics*, pages 227–231.

Yoav Seginer. 2007. Fast unsupervised incremental parsing. In *45th Annual Meeting of the Association of Computational Linguistics (ACL)*, pages 384–391.

Noah A. Smith and Jason Eisner. 2005. Guiding unsupervised grammar induction using contrastive estimation. In *IJCAI Workshop on Grammatical Inference Applications*, pages 73–82.

Valentin I. Spitkovsky, Hiyan Alshawi, and Daniel Jurafsky. 2009. Baby steps: How "Less is more" in unsupervised dependency parsing. In *NIPS 2009 Workshop on Grammar Induction, Representation of Language and Language Learning*.

Valentin I. Spitkovsky, Hiyan Alshawi, and Daniel Jurafsky. 2010a. From baby steps to leapfrog: How "Less is more" in unsupervised dependency parsing. In *Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010)*.

Valentin I. Spitkovsky, Daniel Jurafsky, and Hiyan Alshawi. 2010b. Profiting from Mark-Up: Hyper-Text annotations for guided parsing. In *48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*.

Valentin I. Spitkovsky, Hiyan Alshawi, Angel X Chang, and Daniel Jurafsky. 2011a. Unsupervised dependency parsing without gold Part-of-Speech tags. In *2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*.

Valentin I. Spitkovsky, Hiyan Alshawi, and Daniel Jurafsky. 2011b. Lateen EM: unsupervised training with multiple objectives, applied to dependency grammar induction. In *2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*.

Valentin I. Spitkovsky, Hiyan Alshawi, and Daniel Jurafsky. 2011c. Punctuation: Making a point in unsupervised dependency parsing. In *Fifteenth Conference on Computational Natural Language Learning (CoNLL-2011)*.

Kewei Tu and Vasant Honavar. 2011. On the utility of curricula in unsupervised learning of probabilistic grammars. In *22nd International Joint Conference on Artificial Intelligence (IJCAI 2011)*.