Tsinghua Science and Technology

Volume 25 | Issue 2 Article 5

2020

Robust Unsupervised Discriminative Dependency Parsing

Yong Jiang
the School of Information Science and Technology, ShanghaiTech University, Shanghai 201210, China. Yong Jiang and Jiong Cai are also with Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences, Shanghai 200050, and with University of Chinese Academy of Sciences, Beijing 100049, China.

Jiong Cai
the School of Information Science and Technology, ShanghaiTech University, Shanghai 201210, China. Yong Jiang and Jiong Cai are also with Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences, Shanghai 200050, and with University of Chinese Academy of Sciences, Beijing 100049, China.

Kewei Tu
the School of Information Science and Technology, ShanghaiTech University, Shanghai 201210, China. Yong Jiang and Jiong Cai are also with Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences, Shanghai 200050, and with University of Chinese Academy of Sciences, Beijing 100049, China.

Follow this and additional works at: https://tsinghuauniversitypress.researchcommons.org/tsinghua-science-and-technology

 Part of the Computer Sciences Commons, and the Electrical and Computer Engineering Commons

Recommended Citation

Yong Jiang, Jiong Cai, Kewei Tu. Robust Unsupervised Discriminative Dependency Parsing. Tsinghua Science and Technology 2020, 25(02): 192-202.

This Research Article is brought to you for free and open access by Tsinghua University Press: Journals Publishing. It has been accepted for inclusion in Tsinghua Science and Technology by an authorized editor of Tsinghua University Press: Journals Publishing.

TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 03/12 pp192–202
DOI: 10.26599/TST.2018.9010145
Volume 25, Number 2, April 2020

Robust Unsupervised Discriminative Dependency Parsing

Yong Jiang*, Jiong Cai, and Kewei Tu

Abstract: Discriminative approaches have shown their effectiveness in unsupervised dependency parsing. However, due to their strong representational power, discriminative approaches tend to quickly converge to poor local optima during unsupervised training. In this paper, we tackle this problem by drawing inspiration from robust deep learning techniques. Specifically, we propose robust unsupervised discriminative dependency parsing, a framework that integrates the concepts of denoising autoencoders and conditional random field autoencoders. Within this framework, we propose two types of sentence corruption mechanisms as well as a posterior regularization method for robust training. We tested our methods on eight languages and the results show that our methods lead to significant improvements over previous work.

Key words: unsupervised learning; dependency parsing; autoencoders

1 Introduction

Dependency parsing is an important task in natural language processing. Given a sentence, a dependency parser produces a rooted dependency tree for the words or Part-Of-Speech (POS) tags. Using supervised learning to build an effective dependency parser requires the manual annotation of gold parses, which is difficult and labor-intensive. On the other hand, the unsupervised training of dependency parsers requires no annotated data and is therefore suitable for learning parsers for low-resource languages or new application domains.

Most existing approaches to unsupervised dependency parsing are based on generative models such as the Dependency Model with Valence (DMV)[1] and the Combinatory Categorial Grammar (CCG)[2] models. Typically, generative models make strong assumptions about the output structure and may have an inductive bias that favors a learning process towards the desired linguistic structure. On the other hand, it is not obvious how discriminative learning can be applied to unsupervised parsing because there are no labeled data. Existing discriminative unsupervised parsing approaches are based on the concepts of discriminative clustering[3] and autoencoding[4]. These discriminative approaches typically utilize the rich features of the input sentence and hence have stronger representational power than generative approaches. Consequently, they are more likely to overfit self-generated parses or distributions of parses during early iterations of unsupervised learning, which may lead to early convergence to poor local optima.

To address this problem, we propose a novel framework for the robust unsupervised learning of discriminative dependency parsers. Motivated by the recent success of robust deep learning techniques, such as the dropout mechanism and the denoising autoencoder[5,6], our framework extends the Conditional Random Field (CRF) autoencoder for unsupervised parsing[4] by training with randomly corrupted sentences. We propose two types of effective sentence corruption mechanisms. To constrain the

• Yong Jiang, Jiong Cai, and Kewei Tu are with the School of Information Science and Technology, ShanghaiTech University, Shanghai 201210, China. Yong Jiang and Jiong Cai are also with Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences, Shanghai 200050, and with University of Chinese Academy of Sciences, Beijing 100049, China. E-mail: {jiangyong, caijiong, tukw}@shanghaitech.edu.cn.

∗ To whom correspondence should be addressed.

Manuscript received: 2018-08-12; accepted: 2018-12-24

@ The author(s) 2020. The articles published in this open access journal are distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/).

potential negative effect of random corruption, we further propose a novel posterior regularization method that encourages the original and corrupted sentences to have similar parses. We conducted experiments on the datasets of eight languages and the results show that our approach significantly outperforms previous discriminative approaches. To the best of our knowledge, our work is the first attempt to incorporate robust learning into unsupervised structured prediction. We believe that our work can motivate similar research on many other unsupervised structured prediction tasks.

2 Background

2.1 Unsupervised dependency parsing

Unsupervised dependency parsing is a task in which the goal is to build a dependency parser without supervision from gold parse trees. The literature reports three types of approaches to unsupervised dependency parsing: generative, discriminative, and rule-based approaches.

For generative approaches, previous work has focused on building different generative processes for both the sentence and the corresponding parse tree. For example, the DMV[1] is a popular generative model that outperforms the best branching baseline on the English language. Many subsequent approaches have been proposed to improve the DMV[7–9]. Another type of generative models is based on CCGs[2].

With respect to discriminative approaches, existing work is based on graph-based dependency models[3,4]. Grave and Elhadad[3] proposed to learn the parameters of a graph-based dependency model based on the idea of discriminative clustering[10]. Cai et al.[4] proposed to enhance a graph-based dependency model using a generative decoder as a CRF autoencoder. In this paper, we develop our approach based on the CRF autoencoder model.

In rule-based approaches, the general idea is to predefine a set of linguistic rules by exploiting their universal dependency constraints[11,12].

2.2 Robust learning

Robust learning has been recognized as an effective method for training machine learning models, and two well-known robust learning techniques are the denoising autoencoder and dropout training.

An autoencoder is a three-layer feedforward neural network in which the input is the training data, the hidden states represent the important data features to be learned, and the output is the reconstructed data. To better learn useful features in the hidden layer, Vincent et al.[13] proposed the denoising autoencoder, which corrupts the input of the autoencoder with some random noise (usually sampled from a Gaussian distribution). Denoising autoencoders have been widely used to address various natural language processing problems. We apply the idea of data corruption in denoising autoencoders in our CRF autoencoder approach to unsupervised dependency parsing.

Dropout training has been shown to be very effective in preventing overfitting during the training of deep neural networks and can be regarded as a type of regularization[14,15]. At each training iteration, dropout training randomly omits a subset of nodes in a neural network. Because of its success in many real applications, dropout training has been a standard component included in many deep learning toolkits. Besides its application to deep neural networks, dropout training can also be used in the corruption of sparse features in many machine learning methods, such as Support Vector Machines (SVMs), logistic regression, and linear-chain conditional random fields. Bishop[16] showed that training with explicit noise from additive Gaussian noise can be regarded as L2-type regularizations. Burges and Schölkopf[17] proposed a virtual SVM method in which training data is explicitly augmented with support vectors from previous iterations. van der Maaten et al.[18] proposed a method that an implicit noise can be marginalized during training. Chen et al.[19] proposed a dropout training procedure for linear and nonlinear SVM predictors that marginalizes out corrupted noise variables. To the best of our knowledge, our work is the first to utilize dropout training in unsupervised learning.

2.3 Posterior regularization

In many learning problems, there is access to external task-specific information. Posterior regularization[20] is a framework of probabilistic latent variable models that uses external information to constrain the distribution of the latent variable. The basic idea of posterior regularization is to incorporate a regularization term into a learning objective that constrains the posterior moments of the latent variable. Given that each word is only associated with a few possible tags, Graça et al.[20] applied posterior regularization to POS induction and achieved better results than other Expectation

Maximization (EM)-based approaches. Gillenwater et al.[21] presented a similar approach to unsupervised dependency parsing. Tu and Honavar[22] observed that the ambiguity of natural language sentences is typically low and applied posterior regularization to incorporate this information in unsupervised parsing. Naseem et al.[23] proposed the use of linguistic rules to guide unsupervised dependency parsing based on posterior regularization and achieved promising results in many languages. In addition to unsupervised structured prediction, posterior regularization can also be used in supervised structured prediction by constraining the output space with human-designed rules. Yang and Cardie[24] incorporated discourse and lexical knowledge as soft constraints into sentence level sentiment analysis using posterior regularization. Zhang et al.[25] used posterior regularization to integrate parallel rules, such as the bilingual dictionary, phrase table, and length ratio, as a log linear model to guide the learning of neural machine translation models.

Our approach is motivated by these approaches and uses posterior regularization to encourage the similarity of the parse trees of real and corrupted sentences.

2.4 Graph-based dependency parser

Given an input sentence $x = (\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_n)$, we regard its parse tree as a latent structure represented by a sequence $y = (y_1, y_2, \ldots, y_n)$ where y_i is a pair $\langle t_i, h_i \rangle$, t_i is the head token of the dependency connected to token $\bar{x}_i$ in the parse tree, and $h_i \in \{0, 1, 2, \ldots, n\}$ is the index of this head token in the sentence where 0 denotes the dummy root. In a valid dependency tree y, the n dependencies should form a directed tree rooted at index 0. For languages such as English and Chinese, the dependency edges of most of the syntactic dependency trees do not cross, which results in so-called projective dependency trees. On the other hand, dependency trees characterized by edge crossings are called non-projective dependency trees. Given an input sentence x, the set of all possible valid dependency trees is denoted as $\mathcal{Y}(x)$.

By treating the dependency parser as a structured linear model[26], dependency parsing can be considered to search for the highest scoring tree, as follows:

$$y^* = \arg \max_{y \in \mathcal{Y}(x)} \boldsymbol{w}^{\mathrm{T}} F(x, y) \tag{1}$$

where $\boldsymbol{w}$ is the parameter vector to be learned and $F(x, y)$ is the feature representation of sentence x and dependency tree y.

As the number of dependency trees is exponential with respect to sentence length, discovering the best tree for sentence x is generally intractable. To make it tractable, one must assume factorization of the score function, such as the following first-order factorization

$$F(x, y) = \sum_{i=1}^{n} F(x, h_i, i) \tag{2}$$

where a feature vector $F(x, h_i, i)$ is specified for the dependency edge from the head index h_i to the child index i. The score of a dependency is then defined as the inner product of the feature vector and the weight vector $\boldsymbol{w}$, as follows:

$$\phi_{\boldsymbol{w}}(x, h_i, i) = \boldsymbol{w}^{\mathrm{T}} F(x, h_i, i) \tag{3}$$

The score of a dependency tree y of sentence x is as follows:

$$\phi_{\boldsymbol{w}}(x, y) = \sum_{i=1}^{n} \phi_{\boldsymbol{w}}(x, h_i, i) \tag{4}$$

We define the probability of parse tree y given sentence x as follows:

$$P_{\boldsymbol{w}}(y|x) = \frac{\exp(\phi_{\boldsymbol{w}}(x, y))}{Z_{\boldsymbol{w}}(x)} \tag{5}$$

where $Z_{\boldsymbol{w}}(x)$ is the partition function, which is defined as the sum of the scores for all valid dependency trees:

$$Z_{\boldsymbol{w}}(x) = \sum_{y' \in \mathcal{Y}(x)} \exp(\phi_{\boldsymbol{w}}(x, y')) \tag{6}$$

The partition function can be efficiently computed in $O(n^3)$ time using the inside algorithm for projective tree structures[27] and the matrix-tree theorem for non-projective tree structures[28]. Parsing can be reduced to searching for the maximum spanning tree, which can be done efficiently using Eisner's algorithm for projective dependency parsing and the Chu-Liu-Edmonds algorithm for non-projective dependency parsing.

The first-order assumption is oversimplified. To improve parsing performance, McDonald and Pereira[29] proposed second-order Maximum Spanning Tree (MST) parsing in which the dependency tree score is factorized into the sum of adjacent-edge-pair scores, so the parser can utilize more information to make parsing decisions. However, tractable exact second-order dependency parsing is possible only in projective parsing. Koo and Collins[30] proposed an efficient third-order dependency parser that makes use of sibling-style and grandchild-style interactions for projective dependency parsing.

3 CRF Autoencoder

The CRF autoencoder is proposed as a general framework for unsupervised structured prediction[31]. Cai et al.[4] extended this model for unsupervised dependency parsing. Their model contains an encoder and a decoder. The encoder is the same first-order graph-based dependency parser described in Section 2.4, and the decoder is a token-by-token generative model, in which each token $\hat{x}_i$ is generated independently given its head token t_i. So we have

$$P_\theta(\hat{x}|y) = \prod_{i=1}^{n} \theta_{\hat{x}_i|t_i} \tag{7}$$

where θ is the parameter of the decoder.

The joint probability of $\hat{x}$ and y given the input token sequence x is as follows:

$$P_{\boldsymbol{w},\theta}(\hat{x}, y|x) = P_\theta(\hat{x}|y)P_{\boldsymbol{w}}(y|x) = \frac{\mathrm{e}^{\phi_{\boldsymbol{w}}(x,y)+\sum_{i=1}^{n}\log\theta_{\hat{x}_i|t_i}}}{\sum_{y'\in\mathcal{Y}(x)}\mathrm{e}^{\phi_{\boldsymbol{w}}(x,y')}} = \frac{\mathrm{e}^{\phi'_{\boldsymbol{w},\theta}(x,y,\hat{x})}}{\sum_{y'\in\mathcal{Y}(x)}\mathrm{e}^{\phi_{\boldsymbol{w}}(x,y')}} \tag{8}$$

where $\phi'_{\boldsymbol{w},\theta}(x, y, \hat{x}) = \sum_{i=1}^{n}(\log\theta_{\hat{x}_i|t_i} + \phi_{\boldsymbol{w}}(x, h_i, i))$.

At test time, given an input token sequence x, the best parse y^* can be found via the following,

$$y^* = \arg\max_{y\in\mathcal{Y}(x)} P_{\boldsymbol{w},\theta}(\hat{x}, y|x) = \arg\max_{y\in\mathcal{Y}(x)} \phi'_{\boldsymbol{w},\theta}(x, y, \hat{x}) \tag{9}$$

where we set $\hat{x} = x$. This has the same form as first-order graph-based parsing.

At training time, given a set of unannotated sentences $x_1, x_2, \ldots, x_N$, Cai et al.[4] proposed to optimize the regularized Viterbi conditional log-likelihood,

$$-\frac{1}{N}\sum_{i=1}^{N}\log\left(\max_{y\in\mathcal{Y}(x_i)} P_{\boldsymbol{w},\theta}(\hat{x}_i, y|x_i)\right) + \lambda\Omega(\boldsymbol{w}) \tag{10}$$

in which $\hat{x} = x$, $\Omega(\boldsymbol{w})$ is an L1 regularization term of the parameter $\boldsymbol{w}$ of the encoder, and λ is a hyper-parameter.

4 Robust Learning of CRF Autoencoder

4.1 Our framework

Motivated by the denoising autoencoder, we propose an extension of the CRF autoencoder, called Denoising CRF AutoEncoder (DCRFAE). DCRFAE injects random noise into the original input sentence $\boldsymbol{x}$ and takes in the corrupted input $\tilde{\boldsymbol{x}}$. The model is called Denoising CRF AutoEncoder (DCRFAE). So in our model the hidden variable is the dependency parse tree of the corrupted input sentence, and the target output can be either the original sentence or the corrupted sentence. Note that if we set both the input and the output to the same corrupted sentence, then our approach becomes data augmentation. The difference between the denoising autoencoder, the CRF autoencoder and our model is illustrated in Fig. 1. Our model differs from the denoising autoencoder in that the encoder of our model is a probabilistic conditional random field model ($\tilde{x} \to y$ is a stochastic mapping (Stochastic mapping refers to mapping by sampling from a distribution, whereas deterministic mapping refers to mapping by a set of step functions), while $x \to y$ is a deterministic mapping). Our model differs from the CRF autoencoder in that the input variable is corrupted.

Similar to the original CRF AutoEncoder (CRFAE) model, we aim to maximize the conditional log-likelihood $\log P_{w,\theta}(\hat{x}_i|\tilde{x}_i)$ for each sample i. As such, we have the following learning objective function:

$$J(\boldsymbol{w}, \theta) = -\frac{1}{N}\sum_{i=1}^{N}\log\left(P_{\boldsymbol{w},\theta}(\hat{x}_i|\tilde{x}_i)\right) + \lambda\Omega(\boldsymbol{w}) = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\sum_{y\in\mathcal{Y}(x_i)} P_{\boldsymbol{w},\theta}(\hat{x}_i, y|\tilde{x}_i)\right) + \lambda\Omega(\boldsymbol{w}) \tag{11}$$

This objective function can be optimized using the classic EM algorithm. Neal and Hinton[32] suggested that the EM algorithm could be seen as a coordinate descent on a new objective function augumented by a set of auxiliary distributions $q = \{q_i(y), i = 1, 2, \ldots, N\}$, where $q_i(y)$ is a distribution of variable y for the i-th sample:

$$J(w, \theta, q) = \underbrace{\frac{1}{N}\sum_{i=1}^{N}\sum_{y\in\mathcal{Y}(x_i)} q_i(y)\log q_i(y)}_{\Phi_1} - \underbrace{\frac{1}{N}\sum_{i=1}^{N}\log P_{\boldsymbol{w},\theta}(\hat{x}_i|\tilde{x}_i)}_{\Phi_2} - \underbrace{\frac{1}{N}\sum_{i=1}^{N}\sum_{y\in\mathcal{Y}(x_i)} q_i(y)\log P_{\boldsymbol{w},\theta}(y|\tilde{x}_i, \hat{x}_i)}_{\Phi_3} + \lambda\Omega(\boldsymbol{w}) \tag{12}$$

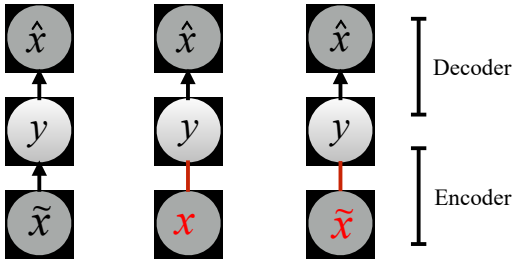


Fig. 1 Left: denoising autoencoder, where $\tilde{x} \to y$ is a deterministic layer. Middle: CRF autoencoder. Right: DCRFAE. In the (denoising) CRF autoencoders, $x \to y$, $\tilde{x} \to y$ are two stochastic mappings.

where $P_{\boldsymbol{w},\theta}(y|\tilde{x}_i, \hat{x}_i)$ is computed using Bayes' theorem, as follows,

$$P_{\boldsymbol{w},\theta}(y|\tilde{x}_i, \hat{x}_i) \propto P_\theta(\hat{x}_i|y_i) P_{\boldsymbol{w}}(y_i|x_i) \tag{13}$$

In EM algorithm, the E-step can be considered to be the optimization of $q(y)$ with $\boldsymbol{w}$ and θ fixed, whearas the M-step can be considered to be the optimization of $\boldsymbol{w}$ and θ with $q(y)$ fixed.

If we require that q is a delta distribution, the objective function becomes as follows:

$$-\min_{\boldsymbol{w},\theta} \frac{1}{N} \sum_{i=1}^{N} \log \left(\max_{y \in \mathcal{Y}(x_i)} P_{\boldsymbol{w},\theta}(\hat{x}_i, y|\tilde{x}_i) \right) + \lambda \Omega(\boldsymbol{w}) \tag{14}$$

which is the same objective function reported in Ref. [4] except for the corrupted input.

By adding a small amount of noise into a sentence, one would expect that the parse tree would not be dramatically changed. Based on this intuition, we add a novel posterior regularization[33] term into the objective function to encourage the original and corrupted sentences to have similar parses, as follows:

$$J(w, q) = \Phi_1 + \Phi_2 + \Phi_3 + \lambda \Omega(\boldsymbol{w}) + \underbrace{\mu \frac{1}{N} \sum_{i=1}^{N} E_{q_i(y)} |\phi_{\boldsymbol{w}}(x_i, y) - \phi_{\boldsymbol{w}}(\tilde{x}_i, y)|}_{\Phi_4} \tag{15}$$

where μ is a hyper-parameter, and $\phi_{\boldsymbol{w}}$ is the score function of the encoder.

The posterior regularization term in our objective function is a new type of inductive bias that can be applied to many other induction problems.

We again optimize the new objective function with coordinate decent. In the E-step, we fix $\boldsymbol{w}$ and θ and optimize the following objective function of q,

$$\Phi_1 + \Phi_3 + \Phi_4 = -\frac{1}{N} \sum_{i=1}^{N} E_{q_i(y)} \Big(-\log q_i(y) + \mu |\phi_{\boldsymbol{w}}(x_i, y) - \phi_{\boldsymbol{w}}(\tilde{x}_i, y)| + \log P_{\boldsymbol{w},\theta}(y|\tilde{x}_i, \hat{x}_i) \Big) \tag{16}$$

If we assume q is a delta distribution, the function is maximized when $q_i(y)$ is centered at

$$\arg \max_{y \in \mathcal{Y}(x)} P_{\boldsymbol{w},\theta}(y|\tilde{x}_i, \hat{x}_i) \mathrm{e}^{\mu(|\phi_{\boldsymbol{w}}(x_i,y) - \phi_{\boldsymbol{w}}(\tilde{x}_i,y)|)} \tag{17}$$

To solve this argmax, we first solve the following two decoding problems, which can be formulated as first-order dependency parsing, as follows:

$$y_1^* = \arg \max_{y \in \mathcal{Y}(x)} \phi'(\tilde{x}_i, y, \hat{x}) + \mu \big(\phi_{\boldsymbol{w}}(x_i, y) - \phi_{\boldsymbol{w}}(\tilde{x}_i, y) \big) \tag{18}$$

$$y_2^* = \arg \max_{y \in \mathcal{Y}(x)} \phi'(\tilde{x}_i, y, \hat{x}) - \mu \big(\phi_{\boldsymbol{w}}(x_i, y) - \phi_{\boldsymbol{w}}(\tilde{x}_i, y) \big) \tag{19}$$

Then the solution to the argmax is the parse with the higher score.

In the M-step, we fix q and optimize the following objective functions of $\boldsymbol{w}$ and θ,

$$\Phi_2 + \Phi_3 + \Phi_4 + \lambda \Omega(\boldsymbol{w}) = -\frac{1}{N} \sum_{i=1}^{N} \Big(\log P_{\boldsymbol{w},\theta}(\hat{x}_i, y_i^*|\tilde{x}_i) + \mu |\phi_{\boldsymbol{w}}(x_i, y_i^*) - \phi_{\boldsymbol{w}}(\tilde{x}_i, y_i^*)| + \lambda \Omega(\boldsymbol{w}) \Big) \tag{20}$$

where y_i^* is the center of the delta function $q_i(y)$, found in the E-step. As the second term in the bracket is not related to θ, we can derive optimal value of θ using Lagrange multipliers, which is the same as the method used in Ref. [4]. To optimize $\boldsymbol{w}$, we use mini-batch sub-gradient descent.

4.2 Corruption mechanisms

We designed two types of corruption, i.e., feature-level corruption and word-level corruption.

Feature-level corruption adds noise into the feature vectors of dependency edges when computing the edge scores $\phi_{\boldsymbol{w}}(x, h_i, i)$. This is similar to dropout training. We corrupt each feature by setting it to zero with probability p_1. Specifically, we have

$$\phi_{\boldsymbol{w}}(\tilde{x}, y) = \sum_{i=1}^{n} \phi_{\boldsymbol{w}}(\tilde{x}, h_i, i) = \sum_{i=1}^{n} \boldsymbol{w}^{\mathrm{T}} (F(x, h_i, i) * I) \tag{21}$$

where I is a vector with the same dimension as the feature vector. * is an element-wise multiplication operator over two vectors. Each element I_i of I is sampled independently from a Bernoulli distribution p, which is defined as follows:

$$p(I_i = 0) = p_1, \ p(I_i = 1) = 1 - p_1 \tag{22}$$

In addition to feature-level corruption, we propose a word-level corruption method. There are often several different noun tags (e.g., plural nouns and proper nouns) and verb tags (e.g., past tense and third person singular present tense) in a treebank corpus. Utilizing the similarity of different sub-types of nouns and verbs is an important technique that is well documented in the unsupervised dependency parsing literature[8, 9, 34]. For example, Berg-Kirkpatrick et al.[8] used the same binary sparse features for different sub-types of nouns and verbs (Section 6.2). Jiang et al.[9] learned POS tag embeddings to model the similarity of different POS tags and found that the embeddings of different sub-types of nouns and verbs are very similar (Section 6.1). Our word-level corruption can be seen as a new approach to the utilization of this type of similarity. Specifically, we replace each noun tag or verb tag in a training sentence with another noun tag or verb tag with probability p_2.

5 Experiments

5.1 Setup

We conducted experiments on the datasets of eight languages that have been widely used for evaluating unsupervised dependency parsing. Seven datasets are drawn from the PASCAL Challenge on Grammar Induction[35]. The English dataset is the Wall Street Journal corpus. Following previous work, we used training sentences of length $\leqslant 10$, tuned all the hyper-parameters on validation sentences of length $\leqslant 10$, and reported the accuracy of the directed dependency on both the test sentences of length $\leqslant 10$ and all the test sentences. Table 1 shows the statistics of our datasets.

5.2 Systems

We compare our approach (Our code is based on CRFAE (https://github.com/caijiong/CRFAE-Dep-Parser). Our code and hyper-parameters will be available at https://github.com/caijiong/CRFAE-Dep-Parser.) with the original CRFAE along with three additional baselines: the DMV[1], the neural DMV[9], and the convex MST[3], which are strong baselines published in the literature.

Table 1 Data statistics of multiple languages.

Language	Number of trainings (length $\leqslant 10$)	Number of developments	Number of tests	Number of POSs
Basque	4478	1010	1121	40
Czech	25 774	9270	10148	57
Danish	1222	1000	1000	25
Dutch	6337	386	386	12
English	5779	1700	2416	35
Portuguese	16 695	400	288	21
Slovene	2821	1000	1000	31
Swedish	3498	389	389	30

Because in our approach the inputs are corrupted differently at each training epoch, the objective function is always changing and convergence is not guaranteed. Therefore, rather than using a stop-criteria based on convergence, we terminate the training algorithm after 20 epochs and use the model produced at the last iteration for tuning and testing. This is why the reported accuracies of the CRFAE differ from those reported by Cai et al.[4], who use convergence-based stop criteria based on the changes of the loss function of the validation dataset.

We tested three variants of our approach: Data Augmentation (DA) which sets $\mu = 0$ and sets both the input and output to the same word-level corrupted sentence; Word-Level (WL) corruption on the input only; and Feature-Level (FL) corruption on the input only. We report the mean and standard deviation for 15 runs of the test data.

5.3 Results with the basic setup

Table 2 shows our experimental results. Note that in the Dutch dataset, there is only one type of verb tags and one type of noun tags, so the DA and WL performances are the same as the that of the baseline CRFAE.

We can see that our approaches perform significantly better overall than the other approaches. Our approach with WL corruption outperforms the CRFAE baseline in four languages, and our approach with FL corruption outperforms the CRFAE baseline by a large margin in seven languages. Comparing of the WL corruption and FL corruption, we can see from the table that on average, FL corruption outperforms WL corruption by 0.76% for sentences no longer than 10 and by 2.4% for sentences of all lengths. The two exceptions are the Danish and Slovene languages, for which WL corruption outperforms FL corruptions. One possible reason for this is that the sizes of the training datasets of the Danish and Slovene languages are small compared with those of the other languages. For large datasets, the similar behaviors of different noun and verb sub-types may be reflected in the training data, whereas small datasets may not contain enough data to capture such

Table 2 Parsing accuracy/standard deviations of our approaches on datasets of eight languages in the basic setup. For our approach, we report the average and standard deviation for 15 runs with different random seeds. Results of previous approaches are taken from Ref. [4].

(%)

	Basque	Czech	Danish	Dutch	English	Portuguese	Slovene	Swedish	Avg.
Length $\leqslant$ 10									
DMV	47.1	27.1	39.1	37.1	58.3	42.6	32.3	23.7	38.4
Neural DMV	48.1	28.6	39.8	37.2	65.9	47.9	36.5	39.9	43.0
Convex MST	29.4	36.5	49.3	31.3	34.4	46.4	33.7	35.5	37.1
CRFAE[4]	49.0	33.9	28.8	39.3	51.4	47.6	34.7	51.3	42.0
CRFAE	49.86	56.31	34.66	29.67	59.25	44.60	41.52	56.77	46.58
DCRFAE-DA	53.69/0.73	45.52/2.46	35.71/2.37	29.67/0.00	53.82/4.90	52.15/5.80	37.66/6.27	53.76/1.33	45.25/2.98
DCRFAE-WL	52.65/0.82	51.26/7.98	**60.69/6.91**	29.67/0.00	55.08/3.78	53.02/12.96	**56.37/0.89**	56.46/2.77	51.90/4.51
DCRFAE-FL	**54.04/1.96**	**55.82/3.41**	49.65/3.82	**47.57/2.28**	**55.72/2.34**	**54.69/7.90**	43.05/1.98	**60.70/3.01**	**52.66/3.34**
Length: All									
DMV	40.9	20.4	32.6	33.0	39.4	36.2	26.9	16.5	30.7
Neural DMV	41.8	23.8	34.2	33.6	47.0	40.2	29.4	30.8	35.1
Convex MST	30.5	33.4	44.2	29.3	28.5	38.3	32.2	28.3	33.1
CRFAE[4]	39.8	25.4	24.2	35.2	37.4	52.2	26.4	40.0	35.1
CRFAE	41.90	44.40	27.57	23.67	47.28	49.35	34.64	42.21	38.88
DCRFAE-DA	44.56/0.78	35.51/1.00	28.97/2.15	23.67/0.00	39.09/5.03	52.73/2.22	32.70/4.87	41.06/0.99	37.29/2.13
DCRFAE-WL	44.35/1.01	39.93/5.67	**52.79/5.00**	23.67/0.00	38.27/4.71	45.87/14.73	**45.30/0.64**	41.74/2.02	41.49/4.22
DCRFAE-FL	**44.82/1.53**	**43.14/2.18**	41.64/3.85	**38.83/6.34**	**45.04/2.70**	**53.26/7.19**	36.62/1.05	**47.75/2.17 /**	**43.89/3.38**

similarity, hence WL corruption can help inducing such similarity.

5.4 Results with linguistic prior

Naseem et al.[23] proposed a way to bias grammar induction using a set of pre-defined universal linguistic rules. This technique has been widely utilized in many grammar induction models[3,4,23]. We enhanced our approach by the use of a universal linguistic prior in the same way reported in Ref. [4], and then repeated all the experiments.

The results are as shown in Table 3, again, our approaches outperformed the other approaches in most cases. Our approach with WL corruption outperformed the CRFAE baseline on six languages, and our approach with FL corruption outperformed the CRFAE baseline on seven languages. FL corruption again performed better than WL corruption in most cases. In addition,

Table 3 Parsing accuracy/standard deviations for our approaches on eight languages with models enriched with linguistic prior (p denotes these models). For our approaches, the average and standard deviations across 15 runs with different random seeds are reported. Results of previous approaches are from Ref. [4].

(%)

	Basque	Czech	Danish	Dutch	English	Portuguese	Slovene	Swedish	Avg.
Length $\leqslant$ 10									
Convex MST (p)	30.0	46.1	51.6	35.3	60.8	55.4	63.7	50.9	49.2
CRFAE[4] (p)	49.9	48.1	53.4	43.9	71.7	68.0	52.5	64.7	56.5
CRFAE (p)	49.98	**53.44**	60.61	41.99	68.85	63.02	50.00	64.34	56.53
DCRFAE-DA (p)	52.46/1.34	52.24/4.40	57.75/0.99	41.99/0.00	67.42/0.67	63.20/1.23	46.50/2.36	64.77/0.88	55.79/1.48
DCRFAE-WL (p)	52.93/0.50	46.17/0.93	**61.88/3.92**	42.65/0.00	**68.39/0.47**	66.73/0.73	**55.29/4.64**	65.96/0.65	57.50/1.48
DCRFAE-FL (p)	**54.97/0.66**	44.79/0.65	60.26/1.76	**46.66/1.50**	67.93/0.81	**72.65/0.27**	50.15/2.24	**68.45/0.79**	**58.23/1.09**
Length: All									
Convex MST (p)	30.6	40.0	45.8	35.6	48.6	46.3	51.8	40.5	42.4
CRFAE[4] (p)	41.4	36.8	40.5	38.6	55.7	58.9	43.3	48.5	45.5
CRFAE (p)	42.02	**42.04**	47.68	40.82	53.86	56.00	39.41	52.34	46.77
DCRFAE-DA (p)	43.60/1.37	39.66/4.52	44.94/1.31	40.82/0.00	52.79/0.70	52.09/1.46	38.89/1.14	49.56/0.80	45.29/1.41
DCRFAE-WL (p)	44.22/0.53	36.60/1.50	**53.15/4.01**	40.92/0.00	54.07/0.71	61.06/0.35	**43.81/5.37**	48.26/0.82	47.76/1.66
DCRFAE-FL (p)	**45.50/0.52**	37.73/0.99	50.22/1.69	**43.94/1.81**	**56.87/0.44**	**67.42/0.17**	41.21/1.70	**55.33/0.84**	**49.78/1.02**

when enhanced with a universal linguistic prior, the variance of our approaches is significantly reduced, which suggests that a universal linguistic prior helps constrain the parameter space and stabilized our approaches. For the Danish and Slovene languages, WL corruption again performs better than FL corruption.

6 Analysis

In this section, we answer several questions regarding our approach.

6.1 Impact of the regularization term

We investigated the utility of the posterior regularization term ϕ_4 by comparing the learning results with $\mu = 0$ and $\mu > 0$ (tuned for each language). We fixed the noise levels at $p_1 = p_2 = 0.3$ and tuned the other hyper-parameters on the validation datasets. Figures 2 and 3 show the results for the test datasets of eight languages. We can see that for all the languages, the posterior regularization term is indeed helpful with both FL corruption and WL corruption.

For FL corruption, we further plotted the change in accuracy with respect to different μ values, with $p_1 = p_2 = 0.3$ and $\alpha = 0.1$, as shown in Fig. 4. For six of the eight languages, the accuracy first increased and then decreased with the increase of μ. We observe similar trends for word level corruption.

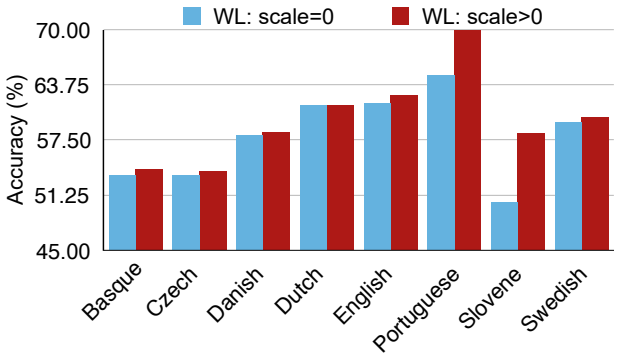


Fig. 2 Accuracies of WL corruption on the validation set with and without our regularization term.

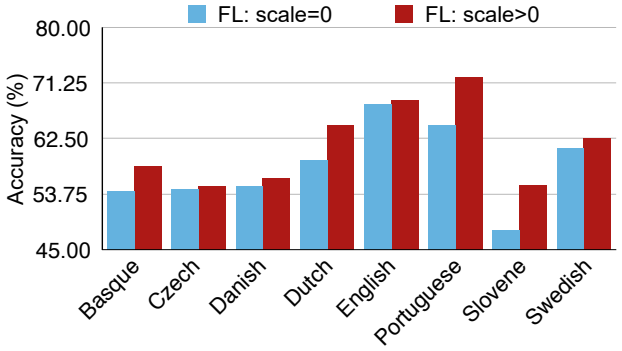


Fig. 3 Accuracies of FL corruption on the validation set with and without our regularization term.

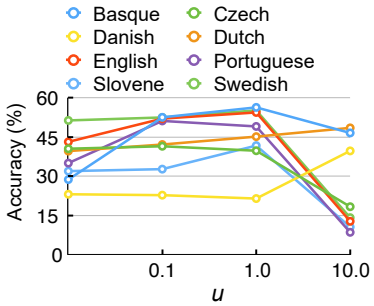


Fig. 4 Change of accuracy of the validation set with different μ values with FL corruption.

6.2 Impact of noise level

Next, we investigated the impact of the noise level on our approach. For FL corruption, we set $\alpha = 0.1, \mu = 1$ and changed the corruption probability p_2. Figure 5 shows the results for three languages in which we can see that in all the three cases the noise level $p_2 = 0.3$ achieved the best performance and higher noise levels reduced the accuracy. Again, we observed similar trends for WL corruption.

6.3 What is learned

Our goal was to discover the kind of syntactic information that can be better learned using our approach as compared with the CRFAE baseline. We computed the F1 score of dependencies headed by NOUN or VERB, and the results are shown in

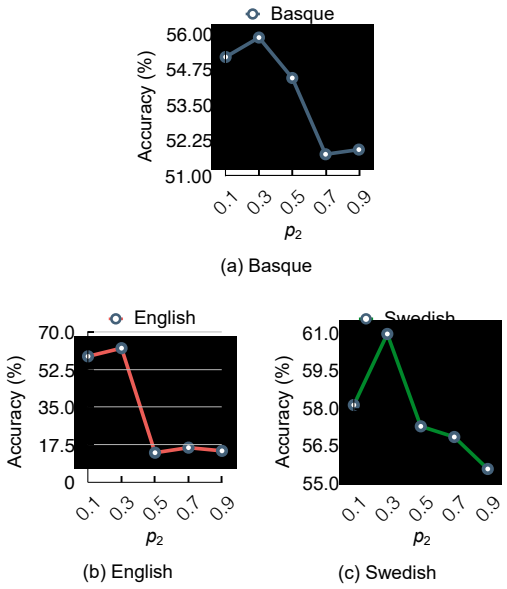


Fig. 5 Change of accuracy of the validation set with different noise level p_2 with FL corruption.

Table 4. Overall, the ranking of the F1 score is FL > BASE > WL. The FL scores of our approach are higher than that of the CRFAE baseline, which showed that our FL corruption can bias the original model to learn better dependencies. However, the WL scores of our approaches are worse than those of the CRFAE baseline, which surprised us. One possible reason is that the model is forced to learn better syntactic relations involving other POS tags if nouns and verbs are corrupted. For the POS tags other than NOUN and VERB, when using WL corruptions, the F1 scores increased compared with those of the CRFAE baseline. It would be interesting to combine the benefits of our two corruption mechanisms, which we leave for future work.

6.4 Escaping local optima

In Section 1, we mentioned that our approach was motivated by the observation that discriminative approaches to unsupervised grammar learning tend to converge early to poor local optima.

Here we investigated whether our approach can alleviate this early convergence problem. We plotted the change in accuracy with the training epochs of our FL corruption approach and the CRFAE baseline for three languages as shown in Fig. 6. We can see that in two of the three cases, CRFAE converges after only a few epochs. In contrast, in all the three cases, our approach does not converge and eventually achieves higher accuracy than the CRFAE.

We also plotted the change in the objective function on the development dataset with the training epochs of DCRFAE and CRFAE. As the objective function of the DCRFAE is not deterministic due to the random corruption, we instead plotted the objective function (negative Viterbi log-likelihood) of the original CRFAE model. From Fig. 7, we see that while the CRFAE converges very quickly, the DCRFAE does not converge because its objective function is stochastic.

Table 4 F1 scores of dependencies headed by NOUN or VERB.

Language	Basic setup			Linguistic prior		
	BASE	WL	FL	BASE	WL	FL
Basque	42.1	40.2	47.5	43.1	42.2	46.7
English	40.3	43.5	50.1	52.4	46.7	54.1
Swedish	53.7	47.9	54.4	55.7	52.9	59.6

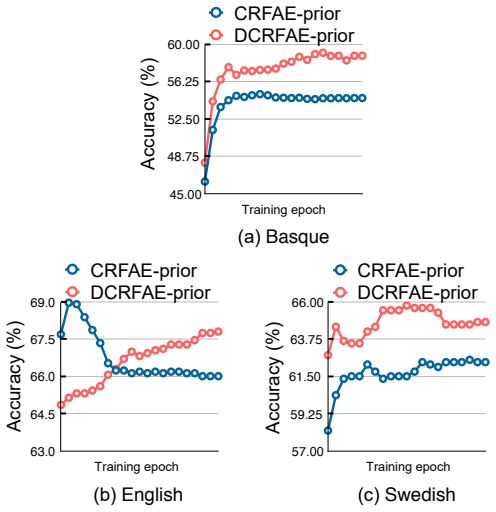


Fig. 6 Accuracy on the validation dataset vs. training epoch for CRFAE and DCRFAE-FL.

7 Conclusion

In this paper, we propose a novel framework for the robust learning of unsupervised dependency parsers. Our framework is based on the conditional random field autoencoder and extends its training approach by training sentence corruption. We presented two types of sentence corruption mechanisms as well as a posterior regularization method for robust training. Our experiments show that our approach can significantly boost the performance of discriminative approaches to unsupervised dependency parsing. Our framework is general, simple, and easily to be adapted to other unsupervised structured prediction problems.

In future work, we plan to consider marginalized noise rather than explicit noise, and hope to reduce the variance of our approach for cases in which no prior information is available. In addition, we plan to test our approach in learning generative models for unsupervised structured prediction problems.

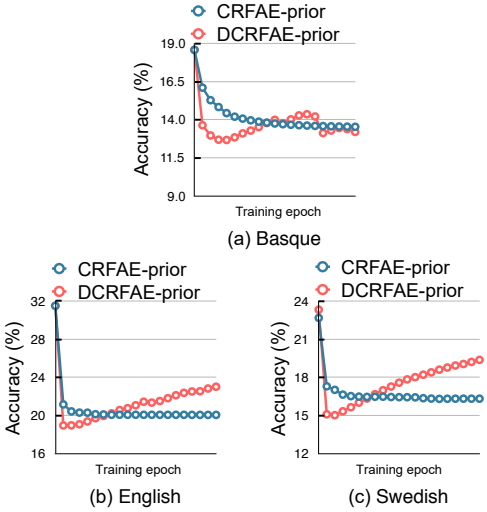


Fig. 7 Negative Viterbi log-likelihood on the validation dataset vs. training epoch for CRFAE and DCRFAE-FL.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (No. 61503248) and the Major Program of Science and Technology Commission Shanghai Municipal (No. 17JC1404102).

References

[1] D. Klein and C. D. Manning, Corpus-based induction of syntactic structure: Models of dependency and constituency, in *Proc. 42nd Annu. Meeting on Association for Computational Linguistics*, Barcelona, Spain, 2004, p. 478.

[2] Y. Bisk and J. Hockenmaier, Simple robust grammar induction with combinatory categorial grammars, in *Proc. 26th AAAI Conf. Artificial Intelligence*, Toronto, Canada, 2012.

[3] E. Grave and N. Elhadad, A convex and feature-rich discriminative approach to dependency grammar induction, in *Proc. 53rd Annu. Meeting of the Association for Computational Linguistics and the 7th Int. Joint Conf. Natural Language Processing*, Beijing, China, 2015, pp. 1375–1384.

[4] J. Cai, Y. Jiang, and K. W. Tu, CRF autoencoder for unsupervised dependency parsing, in *Proc. 2017 Conf. Empirical Methods in Natural Language Processing*, Copenhagen, Denmark, 2017, pp. 1638–1643.

[5] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. A. Manzagol, Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion, *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, 2010.

[6] I. J. Goodfellow, J. Shlens, and C. Szegedy, Explaining and harnessing adversarial examples, in *Proc. Int. Conf. Learning Representations*, San Diego, CA, USA, 2015.

[7] N. A. Smith and J. Eisner, Annealing structural bias in multilingual weighted grammar induction, in *Proc. 21st Int. Conf. Computational Linguistics and the 44th Annu. Meeting of the Association for Computational Linguistics*, Sydney, Australia, 2006, pp. 569–576.

[8] T. Berg-Kirkpatrick, A. Bouchard-Côté, J. DeNero, and D. Klein, Painless unsupervised learning with features, in *Human Language Technologies: The 2010 Annu. Conf. of the North American Chapter of the Association for Computational Linguistics*, Los Angeles, CA, USA, 2010, pp. 582–590.

[9] Y. Jiang, W. Han, and K. Tu, Unsupervised neural dependency parsing, in *Proc. 2016 Conf. Empirical Methods in Natural Language Processing*, Austin, TX, USA, 2016, pp. 763–771.

[10] L. L. Xu, J. Neufeld, B. Larson, and D. Schuurmans, Maximum margin clustering, in *Proc. 17th Int. Conf. Neural Information Processing Systems*, Vancouver, Canada, 2004, pp. 1537–1544.

[11] A. Søgaard, Unsupervised dependency parsing without training, *Nat. Lang. Eng.*, vol. 18, no. 2, pp. 187–203, 2012.

[12] H. Martínez Alonso, Ž. Agic, B. Plank, and A. Søgaard, Parsing universal dependencies without training, in *Proc. 15th Conf. European Chapter of the Association for Computational Linguistics*, East Stroudsburg, PA, USA, 2017, pp. 230–240.

[13] P. Vincent, H. Larochelle, Y. Bengio, and P. A. Manzagol, Extracting and composing robust features with denoising autoencoders, in *Proc. 25th Int. Conf. Machine Learning*, Helsinki, Finland, 2008, pp. 1096–1103.

[14] J. Sietsma and R. J. F. Dow, Creating artificial neural networks that generalize, *Neural Netw.*, vol. 4, no. 1, pp. 67–79, 1991.

[15] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors, arXiv preprint arXiv: 1207.0580, 2012.

[16] C. M. Bishop, Training with noise is equivalent to Tikhonov regularization, *Neural Comput.*, vol. 7, no. 1, pp. 108–116, 1995.

[17] C. J. C. Burges and B. Schölkopf, Improving the accuracy and speed of support vector machines, in *Proc. 9th Int. Conf. Neural Information Processing Systems*, Denver, CO, USA, 1996, pp. 375–381.

[18] L. van der Maaten, M. M. Chen, S. Tyree, and K. Q. Weinberger. Learning with marginalized corrupted features, in *Proc. 30th Int. Conf. Machine Learning*, Atlanta, GA, USA, 2013, pp. 410–418.

[19] N. Chen, J. Zhu, J. F. Chen, and B. Zhang, Dropout training for support vector machines, in *Proc. 28th AAAI Conf. Artificial Intelligence*, Québec City, Canada, 2014, pp. 1752–1759.

[20] J. V. Graça, K. Ganchev, B. Taskar, and F. Pereira, Posterior vs. parameter sparsity in latent variable models, in *Proc. 22nd Int. Conf. Neural Information Processing Systems*, Vancouver, Canada, 2009, pp. 664–672.

[21] J. Gillenwater, K. Ganchev, J. Graça, F. Pereira, and B. Taskar, Sparsity in dependency grammar induction, in *Proc. ACL 2010 Conf. Short Papers*, Uppsala, Sweden, 2010, pp. 194–199.

[22] K. W. Tu and V. Honavar, Unambiguity regularization for unsupervised learning of probabilistic grammars, in *Proc. 2012 Joint Conf. Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Jeju Island, Korea, 2012, pp. 1324–1334.

[23] T. Naseem, H. Chen, R. Barzilay, and M. Johnson, Using universal linguistic knowledge to guide grammar induction, in *Proc. 2010 Conf. Empirical Methods in Natural Language Processing*, Cambridge, MA, USA, 2010, pp. 1234–1244.

[24] B. S. Yang and C. Cardie, Context-aware learning for sentence-level sentiment analysis with posterior regularization, in *Proc. 52nd Annu. Meeting of the Association for Computational Linguistics*, Baltimore, MA, USA, 2014, pp. 325–335.

[25] J. C. Zhang, Y. Liu, H. B. Luan, J. F. Xu, and M. S. Sun, Prior knowledge integration for neural machine translation using posterior regularization, in *Proc. 55th Annu. Meeting of the Association for Computational Linguistics*,

Vancouver, Canada, 2017, pp. 1514–1523.

[26] R. McDonald, F. Pereira, K. Ribarov, and J. Hajic, Non-projective dependency parsing using spanning tree algorithms, in *Proc. Conf. Human Language Technology and Empirical Methods in Natural Language Processing*, Vancouver, Canada, 2005, pp. 523–530.

[27] M. A. Paskin, Cubic-time parsing and learning algorithms for grammatical bigram models, Report, Berkeley, California, University of California, 2001.

[28] T. Koo, A. Globerson, X. Carreras, and M. Collins, Structured prediction models via the matrix-tree theorem, in *Proc. 2007 Joint Conf. Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Prague, Czech Republic, 2007, pp. 141–150.

[29] R. McDonald and F. Pereira, Online learning of approximate dependency parsing algorithms, in *Proc. 11^{th} Conf. European Chapter of the Association for Computational Linguistics*, Trento, Italy, 2006, pp. 81–88.

[30] T. Koo and M. Collins, Efficient third-order dependency parsers, in *Proc. 48^{th} Annu. Meeting of the Association for Computational Linguistics*, Uppsala, Sweden, 2010, pp. 1–11.

[31] W. Ammar, C. Dyer, and N. A. Smith, Conditional random field autoencoders for unsupervised structured prediction, in *Proc. 27^{th} Int. Conf. Neural Information Processing Systems*, Montreal, Canada, 2014, pp. 3311–3319.

[32] R. M. Neal and G. E. Hinton, A view of the EM algorithm that justifies incremental, sparse, and other variants, in *Learning in Graphical Models*, M. I. Jordan, ed. Cambridge, MA, USA: MIT Press, 1998, pp. 355–368.

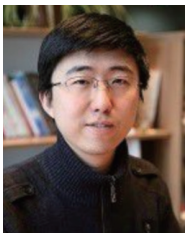
[33] K. Ganchev, J. Graça, J. Gillenwater, B. Taskar, Posterior regularization for structured latent variable models, *J. Mach. Learn. Res.*, vol. 11, pp. 2001–2049, 2010.

[34] S. B. Cohen, K. Gimpel, and N. A. Smith, Logistic normal priors for unsupervised probabilistic grammar induction, in *Proc. 21^{st} Int. Conf. Neural Information Processing Systems*, Vancouver, Canada, 2008, pp. 321–328.

[35] D. Gelling, T. Cohn, P. Blunsom, and J. Graça, The PASCAL challenge on grammar induction, in *Proc. NAACL-HLT Workshop on the Induction of Linguistic Structure*, Montreal, Canada, 2012, pp. 64–80.

Yong Jiang is currently a PhD student at ShanghaiTech University, and with Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences, and also with the University of Chinese Academy of Sciences. His current research mainly focuses on learning the latent structure of modern data, applications on structured prediction, text classification, and probabilistic modeling.

Jiong Cai is a PhD student at ShanghaiTech University, and with the Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences, and also with the University of Chinese Academy of Sciences. His research interests lie in structured prediction problems in NLP, such as dependency parsing and sequence labeling.



Kewei Tu received the BS and MS degrees from Shanghai Jiao Tong University, China, in 2002 and 2005, respectively, and received the PhD degree from Iowa State University, USA, in 2012. During 2012 to 2014, he worked as a postdoctoral researcher at the Vision, Cognition, Learning and Art Laboratory, Departments of Statistics and Computer Science of the University of California, Los Angeles, USA. He has been an assistant professor with the School of Information Science and Technology at ShanghaiTech University, Shanghai, China, since 2014. His research interests include machine learning, natural language processing, probabilistic knowledge representation, computer vision, and artificial intelligence in general.