Policy Gradient as a Proxy for Dynamic Oracles in Constituency Parsing

Daniel Fried and **Dan Klein**
Computer Science Division
University of California, Berkeley
{dfried,klein}@cs.berkeley.edu

Abstract

Dynamic oracles provide strong supervision for training constituency parsers with exploration, but must be custom defined for a given parser's transition system. We explore using a policy gradient method as a parser-agnostic alternative. In addition to directly optimizing for a tree-level metric such as F1, policy gradient has the potential to reduce exposure bias by allowing exploration during training; moreover, it does not require a dynamic oracle for supervision. On four constituency parsers in three languages, the method substantially outperforms static oracle likelihood training in almost all settings. For parsers where a dynamic oracle is available (including a novel oracle which we define for the transition system of Dyer et al. (2016)), policy gradient typically recaptures a substantial fraction of the performance gain afforded by the dynamic oracle.

1 Introduction

Many recent state-of-the-art models for constituency parsing are transition based, decomposing production of each parse tree into a sequence of action decisions (Dyer et al., 2016; Cross and Huang, 2016; Liu and Zhang, 2017; Stern et al., 2017), building on a long line of work in transition-based parsing (Nivre, 2003; Yamada and Matsumoto, 2003; Henderson, 2004; Zhang and Clark, 2011; Chen and Manning, 2014; Andor et al., 2016; Kiperwasser and Goldberg, 2016).

However, models of this type, which decompose structure prediction into sequential decisions, can be prone to two issues (Ranzato et al., 2016; Wiseman and Rush, 2016). The first is *exposure bias*: if, at training time, the model only observes states resulting from correct past decisions, it will not be prepared to recover from its own mistakes during prediction. Second is the *loss mismatch* between the action-level loss used at training and any structure-level evaluation metric, for example F1.

A large family of techniques address the exposure bias problem by allowing the model to make mistakes and explore incorrect states during training, supervising actions at the resulting states using an expert policy (Daumé III et al., 2009; Ross et al., 2011; Choi and Palmer, 2011; Chang et al., 2015); these expert policies are typically referred to as *dynamic oracles* in parsing (Goldberg and Nivre, 2012; Ballesteros et al., 2016). While dynamic oracles have produced substantial improvements in constituency parsing performance (Coavoux and Crabbé, 2016; Cross and Huang, 2016; Stern et al., 2017; González and Gómez-Rodríguez, 2018), they must be custom designed for each transition system.

To address the loss mismatch problem, another line of work has directly optimized for structure-level cost functions (Goodman, 1996; Och, 2003). Recent methods applied to models that produce output sequentially commonly use policy gradient (Auli and Gao, 2014; Ranzato et al., 2016; Shen et al., 2016) or beam search (Xu et al., 2016; Wiseman and Rush, 2016; Edunov et al., 2017) at training time to minimize a structured cost. These methods also reduce exposure bias through exploration but do not require an expert policy for supervision.

In this work, we apply a simple policy gradient method to train four different state-of-the-art transition-based constituency parsers to maximize expected F1. We compare against training with a dynamic oracle (both to supervise exploration and provide loss-augmentation) where one is available, including a novel dynamic oracle that we define for the top-down transition system of

Dyer et al. (2016).

We find that while policy gradient usually outperforms standard likelihood training, it typically underperforms the dynamic oracle-based methods – which provide direct, model-aware supervision about which actions are best to take from arbitrary parser states. However, a substantial fraction of each dynamic oracle's performance gain is often recovered using the model-agnostic policy gradient method. In the process, we obtain new state-of-the-art results for single-model discriminative transition-based parsers trained on the English PTB (92.6 F1), French Treebank (83.5 F1), and Penn Chinese Treebank Version 5.1 (87.0 F1).

2 Models

The transition-based parsers we use all decompose production of a parse tree $\mathbf{y}$ for a sentence $\mathbf{x}$ into a sequence of actions $(a_1, \dots a_T)$ and resulting states $(s_1, \dots s_{T+1})$. Actions a_t are predicted sequentially, conditioned on a representation of the parser's current state s_t and parameters θ:

$$p(\mathbf{y}|\mathbf{x};\theta) = \prod_{t=1}^{T} p(a_t \mid s_t; \theta) \qquad (1)$$

We investigate four parsers with varying transition systems and methods of encoding the current state and sentence: (1) the discriminative Recurrent Neural Network Grammars (RNNG) parser of Dyer et al. (2016), (2) the In-Order parser of Liu and Zhang (2017), (3) the Span-Based parser of Cross and Huang (2016), and (4) the Top-Down parser of Stern et al. (2017).[1] We refer to the original papers for descriptions of the transition systems and model parameterizations.

3 Training Procedures

Likelihood training without exploration maximizes Eq. 1 for trees in the training corpus, but may be prone to exposure bias and loss mismatch (Section 1). Dynamic oracle methods are known to improve on this training procedure for a variety of parsers (Coavoux and Crabbé, 2016; Cross and Huang, 2016; Stern et al., 2017; González and Gómez-Rodríguez, 2018), supervising exploration during training by providing the parser with the best action to take at each explored state. We describe how policy gradient can be applied as an oracle-free alternative. We then compare to several variants of dynamic oracle training which focus on addressing exposure bias, loss mismatch, or both.

3.1 Policy Gradient

Given an arbitrary cost function Δ comparing structured outputs (e.g. negative labeled F1, for trees), we use the *risk objective*:

$$\mathcal{R}(\theta) = \sum_{i=1}^{N} \sum_{\mathbf{y}} p(\mathbf{y} \mid \mathbf{x}^{(i)}; \theta) \Delta(\mathbf{y}, \mathbf{y}^{(i)})$$

which measures the model's expected cost over possible outputs $\mathbf{y}$ for each of the training examples $(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), \dots, (\mathbf{x}^{(N)}, \mathbf{y}^{(N)})$.

Minimizing a risk objective has a long history in structured prediction (Povey and Woodland, 2002; Smith and Eisner, 2006; Li and Eisner, 2009; Gimpel and Smith, 2010) but often relies on the cost function decomposing according to the output structure. However, we can avoid any restrictions on the cost using reinforcement learning-style approaches (Xu et al., 2016; Shen et al., 2016; Edunov et al., 2017) where cost is ascribed to the entire output structure – albeit at the expense of introducing a potentially difficult credit assignment problem.

The policy gradient method we apply is a simple variant of REINFORCE (Williams, 1992). We perform mini-batch gradient descent on the gradient of the risk objective:

$$\begin{aligned}\nabla\mathcal{R}(\theta) &= \sum_{i=1}^{N} \sum_{\mathbf{y}} p(\mathbf{y}|\mathbf{x}^{(i)}) \Delta(\mathbf{y}, \mathbf{y}^{(i)}) \nabla \log p(\mathbf{y}|\mathbf{x}^{(i)}; \theta) \\ &\approx \sum_{i=1}^{N} \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}^{(i)})} \Delta(\mathbf{y}, \mathbf{y}^{(i)}) \nabla \log p(\mathbf{y}|\mathbf{x}^{(i)}; \theta)\end{aligned}$$

where $\mathcal{Y}(\mathbf{x}^{(i)})$ is a set of k candidate trees obtained by sampling from the model's distribution for sentence $\mathbf{x}^{(i)}$. We use negative labeled F1 for Δ.

To reduce the variance of the gradient estimates, we standardize Δ using its running mean and standard deviation across all candidates used so far throughout training. Following Shen et al. (2016), we also found better performance when including the gold tree $\mathbf{y}^{(i)}$ in the set of k candidates $\mathcal{Y}(\mathbf{x}^{(i)})$, and do so for all experiments reported here.[2]

[1] Stern et al. (2017) trained their model using a non-probabilistic, max-margin objective. For comparison to the other models and to allow training with policy gradient, we create a locally-normalized probabilistic variant of their model by applying a softmax function to the predicted scores for each action.

[2] Including the gold tree in the set of candidates does bias

3.2 Dynamic Oracle Supervision

For a given parser state s_t, a dynamic oracle defines an action $a^*(s_t)$ which should be taken to incrementally produce the best tree still reachable from that state.[3]

Dynamic oracles provide strong supervision for training with exploration, but require custom design for a given transition system. Cross and Huang (2016) and Stern et al. (2017) defined optimal (with respect to F1) dynamic oracles for their respective transition systems, and below we define a novel dynamic oracle for the top-down system of RNNG.

In RNNG, tree production occurs in a stack-based, top-down traversal which produces a left-to-right linearized representation of the tree using three actions: OPEN a labeled constituent (which fixes the constituent's span to begin at the next word in the sentence which has not been shifted), SHIFT the next word in the sentence to add it to the current constituent, or CLOSE the current constituent (which fixes its span to end after the last word that has been shifted). The parser stores opened constituents on the stack, and must therefore close them in the reverse of the order that they were opened.

At a given parser state, our oracle does the following:

1. If there are any open constituents on the stack which can be closed (i.e. have had a word shifted since being opened), check the topmost of these (the one that has been opened most recently). If closing it would produce a constituent from the the gold tree that has not yet been produced (which is determined by the constituent's label, span beginning position, and the number of words currently shifted), or if the constituent could not be closed at a later position in the sentence to produce a constituent in the gold tree, return CLOSE.
2. Otherwise, if there are constituents in the gold tree which have not yet been opened in the parser state, with span beginning at the next unshifted word, OPEN the outermost of these.
3. Otherwise, SHIFT the next word.

While we do not claim that this dynamic oracle is optimal with respect to F1, we find that it still helps substantially in supervising exploration (Section 5).

Likelihood Training with Exploration Past work has differed on how to use dynamic oracles to guide exploration during oracle training (Ballesteros et al., 2016; Cross and Huang, 2016; Stern et al., 2017). We use the same sample-based method of generating candidate sets $\mathcal{Y}$ as for policy gradient, which allows us to control the dynamic oracle and policy gradient methods to perform an equal amount of exploration. Likelihood training with exploration then maximizes the sum of the log probabilities for the oracle actions for all states composing the candidate trees:

$$\mathcal{L}_E(\theta) = \sum_{i=1}^{N} \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}^{(i)})} \sum_{s \in \mathbf{y}} \log p(a^*(s) \mid s)$$

where $a^*(s)$ is the dynamic oracle's action for state s.

Softmax Margin Softmax margin loss (Gimpel and Smith, 2010; Auli and Lopez, 2011) addresses loss mismatch by incorporating task cost into the training loss. Since trees are decomposed into a sequence of local action predictions, we cannot use a global cost, such as F1, directly. As a proxy, we rely on the dynamic oracles' action-level supervision.

In all models we consider, action probabilities (Eq. 1) are parameterized by a softmax function

$$p_{ML}(a \mid s_t; \theta) \propto \exp(z(a, s_t, \theta))$$

for some state–action scoring function z. The softmax-margin objective replaces this by

$$p_{SMM}(a \mid s_t; \theta) \propto \exp(z(a, s_t, \theta) + \Delta(a, a_t^*)) \tag{2}$$

We use $\Delta(a, a_t^*) = 0$ if $a = a_t^*$ and 1 otherwise. This can be viewed as a "soft" version of the max-margin objective used by Stern et al. (2017) for training without exploration, but retains a locally-normalized model that we can use for sampling-based exploration.

the estimate of the risk objective's gradient; however since in the parsing tasks we consider, the gold tree has constant and minimal cost, augmenting with the gold is equivalent to jointly optimizing the standard likelihood and risk objectives, using an adaptive scaling factor for each objective that is dependent on the cost for the trees that have been sampled from the model. We found that including the gold candidate in this manner outperformed initial experiments that first trained a model using likelihood training and then fine-tuned using unbiased policy gradient.

[3]More generally, an oracle can return a set of such actions that could be taken from the current state, but the oracles we use select a single canonical action.

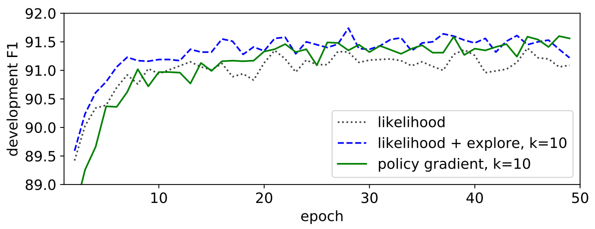

Figure 1: English development set F1 by training epoch, comparing likelihood training with two exploration variants for the Top-Down parser.

Softmax Margin with Exploration Finally, we train using a combination of softmax margin loss augmentation and exploration. We perform the same sample-based candidate generation as for policy gradient and likelihood training with exploration, but use Eq. 2 to compute the training loss for candidate states. For those parsers that have a dynamic oracle, this provides a means of training that more directly provides both exploration and cost-aware losses.

4 Experiments

We compare the constituency parsers listed in Section 2 using the above training methods. Our experiments use the English PTB (Marcus et al., 1993), French Treebank (Abeillé et al., 2003), and Penn Chinese Treebank (CTB) Version 5.1 (Xue et al., 2005).

Training To compare the training procedures as closely as possible, we train all models for a given parser in a given language from the same randomly-initialized parameter values.

We train two different versions of the RNNG model: one model using size 128 for the LSTMs and hidden states (following the original work), and a larger model with size 256. We perform evaluation using greedy search in the Span-Based and Top-Down parsers, and beam search with beam size 10 for the RNNG and In-Order parsers. We found that beam search improved performance for these two parsers by around 0.1-0.3 F1 on the development sets, and use it at inference time in every setting for these two parsers.

In our experiments, policy gradient typically requires more epochs of training to reach performance comparable to either of the dynamic oracle-based exploration methods. Figure 1 gives a typical learning curve, for the Top-Down parser on English. We found that policy gradient is also more sensitive to the number of candidates sampled per sentence than either of the other exploration methods, with best performance on the development set usually obtained with $k = 10$ for $k \in \{2, 5, 10\}$ (where k also counts the sentence's gold tree, included in the candidate set). See Appendix A in the supplemental material for the values of k used.

Tags, Embeddings, and Morphology We largely follow previous work for each parser in our use of predicted part-of-speech tags, pretrained word embeddings, and morphological features.

All parsers use predicted part-of-speech tags as part of their sentence representations. For English and Chinese, we follow the setup of Cross and Huang (2016): training the Stanford tagger (Toutanova et al., 2003) on the training set of each parsing corpus to predict development and test set tags, and using 10-way jackknifing to predict tags for the training set.

For French, we use the predicted tags and morphological features provided with the SPMRL dataset (Seddah et al., 2014). We modified the publicly released code for all parsers to use predicted morphological features for French. We follow the approach outlined by Cross and Huang (2016) and Stern et al. (2017) for representing morphological features as learned embeddings, and use the same dimensions for these embeddings as in their papers. For RNNG and In-Order, we similarly use 10-dimensional learned embeddings for each morphological feature, feeding them as LSTM inputs for each word alongside the word and part-of-speech tag embeddings.

For RNNG and the In-Order parser, we use the same word embeddings as the original papers for English and Chinese, and train 100-dimensional word embeddings for French using the structured skip-gram method of Ling et al. (2015) on French Wikipedia.

5 Results and Discussion

Table 1 compares parser F1 by training procedure for each language. Policy gradient improves upon likelihood training in 14 out of 15 cases, with improvements of up to 1.5 F1. One of the three dynamic oracle-based training methods – either likelihood with exploration, softmax margin (SMM), or softmax margin with exploration – obtains better performance than policy gradient in 10 out of 12 cases. This is perhaps unsurprising given the strong supervision provided by the dynamic oracles and the credit assignment problem faced by

training	English	French	Chinese
Span-Based (Cross and Huang, 2016)			
C&H*	91.3	83.3	—
likelihood	91.0	81.5	83.3
policy gradient	91.4 (+0.4)	81.4 (-0.1)	83.5 (+0.2)
likelihood+explore*	91.3 (+0.3)	81.2 (-0.3)	83.5 (+0.2)
SMM*	91.3 (+0.3)	81.5 (+0.0)	83.7 (+0.4)
SMM+explore*	**91.5 (+0.5)**	**81.7 (+0.2)**	**84.0 (+0.7)**
Top-Down (Stern et al., 2017)			
Stern et al.*†	91.8	82.2	—
likelihood	91.2	80.7	83.9
policy gradient	**91.4 (+0.2)**	81.4 (+0.7)	84.7 (+0.8)
likelihood+explore*	91.3 (+0.1)	81.5 (+0.8)	**85.3 (+1.4)**
SMM*	91.1 (-0.1)	81.2 (+0.5)	84.5 (+0.6)
SMM+explore*	**91.4 (+0.2)**	**81.9 (+1.2)**	84.8 (+0.9)
RNNG Discriminative, Size 128 (Dyer et al., 2016)			
Dyer et al.	91.7	—	84.6
likelihood	91.4	83.2	84.5
policy gradient	91.6 (+0.2)	83.3 (+0.1)	84.7 (+0.2)
likelihood+explore*	**92.1 (+0.7)**	83.0 (-0.2)	**85.5 (+1.0)**
SMM*	91.5 (+0.1)	82.8 (-0.4)	83.6 (-0.9)
SMM+explore*	**92.1 (+0.7)**	**83.5 (+0.3)**	85.0 (+0.5)
RNNG Discriminative, Size 256			
likelihood	91.7	83.1	84.5
policy gradient	92.3 (+0.7)	**83.2 (+0.1)**	85.6 (+1.1)
likelihood+explore	**92.6 (+0.9)**	82.9 (-0.2)	**86.0 (+1.5)**
In-Order (Liu and Zhang, 2017)			
L&Z	91.8	—	86.1
likelihood	91.6	82.7	85.5
policy gradient	**92.2 (+0.6)**	**83.3 (+0.6)**	**87.0 (+1.5)**

Table 1: Test set F1 by training procedure, and in comparison to past work using the same models. Improvements over likelihood training are indicated in parentheses, with the highest results among the training procedures compared here in bold. *: training uses a dynamic oracle; †: past work using a global scoring model (all models we train here are locally-normalized).

policy gradient. However, a substantial fraction of this performance gain is recaptured by policy gradient in most cases.

While likelihood training with exploration using a dynamic oracle more directly addresses exploration bias, and softmax margin training more directly addresses loss mismatch, these two phenomena are still entangled, and the best dynamic oracle-based method to use varies. The effectiveness of the oracle method is also likely to be influenced by the nature of the dynamic oracle available for the parser. For example, the oracle for RNNG lacks F1 optimality guarantees, and softmax margin without exploration often underperforms likelihood for this parser. However, exploration improves softmax margin training across all parsers and conditions.

Although results from likelihood training are mostly comparable between RNNG-128 and the larger model RNNG-256 across languages, policy gradient and likelihood training with exploration both typically yield larger improvements in the larger models, obtaining 92.6 F1 for English and 86.0 for Chinese (using likelihood training with exploration), although results are slightly higher for the policy gradient and dynamic oracle-based methods for the smaller model on French (including 83.5 with softmax margin with exploration). Finally, we observe that policy gradient also provides large improvements for the In-Order parser, where a dynamic oracle has not been defined.

We note that although some of these results (92.6 for English, 83.5 for French, 87.0 for Chinese) are state-of-the-art for single model, discriminative transition-based parsers, other work on constituency parsing achieves better performance through other methods. Techniques that combine multiple models or add semi-supervised data (Vinyals et al., 2015; Dyer et al., 2016; Choe and Charniak, 2016; Kuncoro et al., 2017; Liu and Zhang, 2017; Fried et al., 2017) are orthogonal to, and could be combined with, the single-model, fixed training data methods we explore. Other recent work (Gaddy et al., 2018; Kitaev and Klein, 2018) obtains comparable or stronger performance with global chart decoders, where training uses loss augmentation provided by an oracle. By performing model-optimal global inference, these parsers likely avoid the exposure bias problem of the sequential transition-based parsers we investigate, at the cost of requiring a chart decoding procedure for inference.

Overall, we find that although optimizing for F1 in a model-agnostic fashion with policy gradient typically underperforms the model-aware expert supervision given by the dynamic oracle training methods, it provides a simple method for consistently improving upon static oracle likelihood training, at the expense of increased training costs.

Acknowledgments

DF is supported by a Huawei / Berkeley AI fellowship. This research used the Savio computational cluster provided by the Berkeley Research Computing program at the University of California, Berkeley.

References

Anne Abeillé, Lionel Clément, and François Toussenel. 2003. *Building a Treebank for French*. Springer Netherlands, Dordrecht.

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally normalized transition-based neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2442–2452, Berlin, Germany. Association for Computational Linguistics.

Michael Auli and Jianfeng Gao. 2014. Decoder integration and expected BLEU training for recurrent neural network language models. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 136–142.

Michael Auli and Adam Lopez. 2011. Training a log-linear parser with loss functions via softmax-margin. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 333–343, Stroudsburg, PA, USA. Association for Computational Linguistics.

Miguel Ballesteros, Yoav Goldberg, Chris Dyer, and Noah A. Smith. 2016. Training with exploration improves a greedy stack lstm parser. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2005–2010, Austin, Texas. Association for Computational Linguistics.

Kai-Wei Chang, Akshay Krishnamurthy, Alekh Agarwal, Hal Daumé III, and John Langford. 2015. Learning to search better than your teacher. In *International Conference on Machine Learning*.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 740–750.

Do Kook Choe and Eugene Charniak. 2016. Parsing as language modeling. In *Empirical Methods in Natural Language Processing*.

Jinho D. Choi and Martha Palmer. 2011. Getting the most out of transition-based dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11, pages 687–692, Stroudsburg, PA, USA. Association for Computational Linguistics.

Maximin Coavoux and Benoit Crabbé. 2016. Neural greedy constituent parsing with dynamic oracles. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 172–182, Berlin, Germany. Association for Computational Linguistics.

James Cross and Liang Huang. 2016. Span-based constituency parsing with a structure-label system and provably optimal dynamic oracles. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1–11, Austin, Texas. Association for Computational Linguistics.

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

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. 2016. Recurrent neural network grammars. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 199–209, San Diego, California. Association for Computational Linguistics.

Sergey Edunov, Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2017. Classical structured prediction losses for sequence to sequence learning. *arXiv preprint arXiv:1711.04956*.

Daniel Fried, Mitchell Stern, and Dan Klein. 2017. Improving neural parsing by disentangling model combination and reranking effects. In *Proceedings of ACL*.

David Gaddy, Mitchell Stern, and Dan Klein. 2018. What's going on in neural constituency parsers? an analysis. In *Proceedings of NAACL*.

Kevin Gimpel and Noah A. Smith. 2010. Softmax-margin crfs: Training log-linear models with cost functions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 733–736, Los Angeles, California. Association for Computational Linguistics.

Yoav Goldberg and Joakim Nivre. 2012. A dynamic oracle for arc-eager dependency parsing. In *Proceedings of COLING 2012*, pages 959–976, Mumbai, India. The COLING 2012 Organizing Committee.

Daniel Fernández González and Carlos Gómez-Rodríguez. 2018. Faster shift-reduce constituent parsing with a non-binary, bottom-up strategy. *arXiv preprint*, 1804.07961.

Joshua Goodman. 1996. Parsing algorithms and metrics. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pages 177–183, Santa Cruz, California, USA. Association for Computational Linguistics.

James Henderson. 2004. Discriminative training of a neural network statistical parser. In *Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional lstm feature representations. *Transactions* *of the Association for Computational Linguistics*, 4:313–327.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. *arXiv preprint arXiv:1805.01052*.

Adhiguna Kuncoro, Miguel Ballesteros, Lingpeng Kong, Chris Dyer, Graham Neubig, and Noah A. Smith. 2017. What do recurrent neural network grammars learn about syntax? In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1249–1258, Valencia, Spain. Association for Computational Linguistics.

Zhifei Li and Jason Eisner. 2009. First- and second-order expectation semirings with applications to minimum-risk training on translation forests. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 40–51, Stroudsburg, PA, USA. Association for Computational Linguistics.

Wang Ling, Chris Dyer, Alan W Black, and Isabel Trancoso. 2015. Two/too simple adaptations of word2vec for syntax problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1299–1304. Association for Computational Linguistics.

Jiangming Liu and Yue Zhang. 2017. In-order transition-based constituent parsing. *Transactions of the Association for Computational Linguistics*, 5:413–424.

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19(2):313–330.

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan. Association for Computational Linguistics.

Daniel Povey and Philip C Woodland. 2002. Minimum phone error and i-smoothing for improved discriminative training. In *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, volume 1, pages I–105. IEEE.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. *International Conference on Learning Representations*.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635.

Djamé Seddah, Sandra Kübler, and Reut Tsarfaty. 2014. Introducing the spmrl 2014 shared task on parsing morphologically-rich languages. In *Proceedings of the First Joint Workshop on Statistical Parsing of Morphologically Rich Languages and Syntactic Analysis of Non-Canonical Languages*, pages 103–109, Dublin, Ireland. Dublin City University.

Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2016. Minimum risk training for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1683–1692, Berlin, Germany. Association for Computational Linguistics.

David A. Smith and Jason Eisner. 2006. Minimum risk annealing for training log-linear models. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 787–794, Sydney, Australia. Association for Computational Linguistics.

Mitchell Stern, Jacob Andreas, and Dan Klein. 2017. A minimal span-based neural constituency parser. In *Proceedings of ACL*.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 173–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Oriol Vinyals, Łukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *Proceedings of NIPS*, pages 2773–2781.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Reinforcement Learning*, pages 5–32. Springer.

Sam Wiseman and Alexander M. Rush. 2016. Sequence-to-sequence learning as beam-search optimization. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1296–1306, Austin, Texas. Association for Computational Linguistics.

Wenduan Xu, Michael Auli, and Stephen Christopher Clark. 2016. Expected f-measure training for shift-reduce parsing with recurrent neural networks. In *Proceedings of the 2016 Conference of the North*

American Chapter of the Association for Computational Linguistics on Human Language Technology. Association for Computational Linguistics.

Naiwen Xue, Fei Xia, Fu-dong Chiou, and Marta Palmer. 2005. The penn chinese treebank: Phrase structure annotation of a large corpus. *Nat. Lang. Eng.*, 11(2):207–238.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*.

Yue Zhang and Stephen Clark. 2011. Syntactic processing using the generalized perceptron and beam search. *Computational linguistics*, 37(1):105–151.