Joint Bi-Affine Parsing and Semantic Role Labeling

Peng Shi
David R. Cheriton School of Computer Science
University of Waterloo
Canada
peng.shi@uwaterloo.ca

Yue Zhang
Information Systems Technology and Design Pillar
Singapore University of Technology and Design
Singapore
yue_zhang@sutd.edu.sg

***Abstract*—We propose a simple encoder-decoder model for joint learning of dependency parsing and semantic role labeling (SRL). Experiments on CoNLL-2009 datasets show that our model is competitive with the state-of-the-art ensemble model on SRL task and significantly outperforms state-of-the-art joint models on joint evaluation metrics. Results show that with the implicit encoding, the syntax information can further improve a state-of-the-art semantic role labeler.**

***Keywords*-Parsing, SRL, Joint Learning**

I. Introduction

Semantic role labeling (SRL) [1] defines shallow semantic dependencies between arguments and predicates. Traditional statistical models rely heavily on outputs of syntactic parsing [2]–[5], benefiting from syntactic features. Recently, neural models have become the dominate approach for SRL. With the representation learning ability of neural network, whether to use the syntactic feature becomes debatable. Both syntax-dependent semantic role labeler [6]–[9] and syntax-agnostic ones [10]–[12] achieve state-of-the-art accuracies.

Regardless of being statistical or neural, all SRL models above rely on explicit parser outputs. The neural models extract syntactic information by learning representations of parse trees, via methods including dependency path embedding [8] and tree-LSTMs [13]. However, such models can be negatively impacted by parser errors. One solution to this problem is to perform joint learning of syntax and semantic roles, which are intuitively related knowledge. However, joint parsing and semantic role labeling turns out a highly challenging task, with negative results being reported [14]–[16].

Neural network models shed new light on the joint task. [17] leveraged the shift-reduce algorithm of [18] and a stack LSTM structure [19], showing that SRL can benefit from joint decoding, without degrading parsing accuracies. [13] showed that both parsing and SRL can be improved by sharing word embeddings between two neural models, using a Chinese dataset. Their finding is in line with [20] and [21], who find that multi-task learning between SRL and non-parsing tasks lead to improvements. On the other hand, no work has done investigating deeper parameter sharing between syntactic and SRL tasks.

We fill this gap by empirically investigating a conceptually simple model that integrates the dependency parsing of [22] and SRL model of [12], sharing model parameters beyond word embeddings. These signle models rely on the LSTM encoder layers, with light output layers, which make it feasible to share information heavily using the encoder. For long sentences whose predicted syntax trees can be incorrect in a pipeline model, implicit syntactic information can be learned in our joint model and support for better semantic role label prediction. Compared to the method of [17], our model decouples syntactic and semantic outputs, performing only joint learning but not joint decoding. Compared with [13], our model performs deeper parameter sharing. On standard CoNLL09 datasets, our model achieves state-of-the-art performance on SRL task and joint learning task.

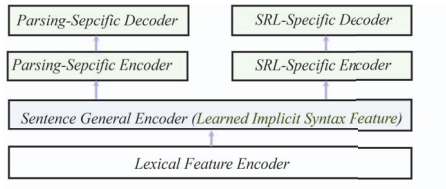

Figure 1. System Architecture

II. model

Shown in Figure 1, our model consists of two main components, namely the encoder and the decoder, respectively. The encoder component (Section II-A) is split into general encoder layers shared by syntax and semantic roles, and task-specific encoder layers. The decoder components (Section II-B) are task-specific, taken from [22] and [12], respectively.

A. Encoder

1) Lexical Feature Encoder: For each word in the given sentence, we create a word representation x_t, which has three components: a word embedding w_t, a pre-trained word embedding pre_t, and a part-of-speech (POS) tag embedding pos_t. All embeddings are randomly initialized and fine-tuned during the training, except for the fixed pre-trained embeddings pre_t. Following [22], the final lexical feature representation is vector $x_t = (w_t + pre_t) \circ pos_t$, where $\circ$ denotes the concatenation operation.

2) General Sentence Encoder: A three-layer bidirectional Long Short Term Memory network (LSTM) with h hidden units is used to model each sentence. We adopt the LSTM variation given by Graves [23].

978-1-5386-1091-9/17/$31.00 ©2017 IEEE

3) Task-Specific Encoder: This component is used to model task-specific information such as the given predicate for SRL.

- SRL-Specific Encoder: For SRL, two additional information resources are used for the encoder [10], [12]: a predicted lemma embedding le_t and a predicate indicator embedding ind_t. We obtain the hidden feature hid_t of each word with context information from the general sentence encoder, and concatenate it with two additional embeddings $r_t^{(en)} = hid_t \circ le_t \circ ind_t$. The lemma and indicator embeddings are active when the position is for predicate in the given sentence. Otherwise, a padding is used. The new feature representations $t_t^{(en)}$ are fed into one-layer bidirectional LSTM for encoding.
- Parser-Specific Encoder: For dependency parsing, we simply feed the general representation hid to a one-layer bidirectional LSTM in order to train parser-specific parameters.

B. Task-Specific Decoder

1) SRL Decoder: Given a sentence and a predicate p, the SRL decoder assigns a semantic role to each word with role label r. Because this prediction is predicate-related, we concatenate the hidden representation of each word $r_t^{(srl)}$ with the predicate hidden representation $r_p^{(srl)}$ and feed them into a classifier:

$$p(r|r_i^{srl}, r_p^{(srl)}, l) \propto exp(W_{l,r}(r_i^{(srl)} \circ r_p^{(srl)})),$$

where l is the predicted lemma of predicate p. We follow [7], [9] and [12] , using a predicate-role related transform parameter $W_{l,r}$ instead of a fixed parameter W_r. We concatenate the embeddings of predicted lemma and semantic role label, and then feed them into multilayer perceptron (MLP).

$$W_{l,r} = ReLU(U(le_l \circ label_r)),$$

where U is a parameter, le_l is fetched from the previous embedding introduced in Section II-A3, and $label_r$ is the embedding of the role labels, which is initialized randomly and fine-tuned during training. This setting makes the role prediction predicate-specific. The classifier computes the probability of each role for each word, including *NULL*, which is used when that word is not an argument of the specific predicate.

2) Parser Decoder: A deep bilinear attention mechanism [22] is used for the parser decoder. In particular, the recurrent states $r^{(parser)}$ of the parser encoder are fed into four MLPs for parsing and dependency relation classification.

$$\begin{aligned}
h_i^{(arc-dep)} &= MLP^{(arc-dep)}(r_i^{(parser)}) \\
h_j^{(arc-head)} &= MLP^{(arc-head)}(r_j^{(parser)}) \\
h_i^{(rel-dep)} &= MLP^{(rel-dep)}(r_i^{(parser)}) \\
h_{y_i^{(arc)}}^{(rel-head)} &= MLP^{(rel-head)}(r_{y_i^{(arc)}}^{(parser)})
\end{aligned}$$

MLPs are used to solve the overfitting problem by dimension reduction before modeling the relation between word i and word j. Four different representations can be derived here. $h_i^{(arc-dep)}$ is the representation when word i is the dependent node in the arc. Similarly, $h_j^{(arc-head)}$ is the feature when the word j is the head node in the arc. The other two representations have similar meanings but are used for label prediction. $y_i^{(arc)}$ denotes the head node of the word i (gold head word is used during training and predicted one is used during testing). $h_i^{(rel-dep)}$ and $h_{y_i^{(arc)}}^{(rel-head)}$ have similar meanings. Hence $y_i^{(arc)}$ is different from j, because $y_i^{(arc)}$ represents the head of i in the relationship but not an arbitrary node j.

Biaffine transformation is employed for calculating the probability of j being the head of i:

$$\begin{aligned}
s_{ij}^{(arc)} = {} & h_i^{T(arc-dep)} U^{(arc)} h_j^{(arc-head)} \\
& + W^{T(arc)} h_j^{(arc-head)},
\end{aligned}$$

where $U^{(arc)}$ and $w^{T(arc)}$ are parameters.

A similar method is used to calculate the probability of i baring relation rel with $y_i^{(arc)}$.

$$\begin{aligned}
s_i^{(rel)} = {} & h_i^{(rel-dep)} U^{(rel)} h_{y_i^{(arc)}}^{(rel-head)} \\
& + W^{(rel)}(h_i^{(rel-dep)} \oplus h_{y_i^{(arc)}}^{(rel-head)}) \\
& + b^{(rel)},
\end{aligned}$$

where $U^{(rel)}$, $W^{(rel)}$ and $b^{(rel)}$ are parameters.

The Chu-Liu-Edmonds Algorithm is used for tree structure decoding.

C. Predicate Disambiguation

We follow the setting of the CoNLL 2009 task, in which the predicates are given for each sentence during both training and testing. For the sense disambiguation subtask, a simple Bi-LSTM model is used. A word is represented by the concatenation of its word embedding, predicted POS embedding, pretrained word embedding, predicate lemma embedding and predicate flag embeddings. This representation is fed into a single-layer Bi-LSTM, from which the concatenation of the hidden states of the predicate and predicate lemma embeddings is fed into a MLP and then a softmax classifier to obtain the predicate sense. At test time, if a predicate has never been seen during training, the first sense is used for this predicate.

III. Experiments

A. Experimental Settings

1) Dataset: We follow [17] to evaluate our model on the CoNLL 2009 data set including English, Chinese and German Dataset, with standard training, development and test splits. Predicted POS tags and lemmas are used for all experiments.

2) Model Details: GloVe embeddings [24] are used for predicate disambiguation. For semantic role labeling, we used external embeddings of [19] trained by the structured skip n-gram approach of [25]. Word2Vec [26] is used in Chinese Dataset and skip n-gram is used in German Dataset for both predicate disambiguation and joint task.

Table I
SRL LABELED F_1-SCORE OF OUR MODEL AND SEVERAL STATE-OF-THE-ART SYSTEMS.

	Excluding predicate senses			**Including predicate senses**	
Model	WSJ-dev	WSJ-test	Brown-test	WSJ-test	Brown-test
Lei et al. 2015	81.03	82.51	70.77	86.58	75.57
FitzGerald et al., 2015	82.3	83.6	71.9	87.3	75.2
Roth and Lapata, 2016	-	-	-	86.7	75.3
Swayamdipta et al., 2016	-	-	-	84.97	74.48
Guo et al., 2016	83.51	85.04	73.22	88.37	77.34
Marcheggiani et al. (2017a)	-	-	-	87.6	77.3
Marcheggiani et al. (2017b)	-	-	-	88.0	77.2
this work	**85.58**	**87.11**	**77.43**	**89.07**	**78.93**
Model + Reranker/Ensemble	WSJ-dev	WSJ-test	Brown-test	WSJ-test	Brown-test
Roth and Lapat, 2016 + R,E	-	-	-	87.9	76.5
Marcheggiani et al. 2017b + E	-	-	-	**89.1**	**78.9**

Table II
CONLL-2009 ENGLISH RESULTS. STATISTICAL SIGNIFICANCE (JOINT *vs*. SEMANTIC-ONLY) WITH $p < 0.05$ IS MARKED WITH $*$

Model	**UAS**	**LAS**	**Sem.** F_1
Syntax-only	94.35	89.72	-
Semantic-only	-	-	88.88
Joint	94.47	89.75	89.07*

Table III
$MacroF_1$ FOR JOINT MODELS ON CONLL 2009.

Language	*CoNLL'09 best*	*Swayam-dipta'16*	*this work*
English	87.69	87.45	**89.42**
German	82.44	81.05	**83.50**
Chinese	76.38	79.27	**82.67**

We set the words as *UNK* when frequency is 1. Word embedding and pretrained word embedding size is 100 for English and Chinese, 300 for German[1]. All other embeddings size is 100. All LSTM hidden states size are all set to 512 and all MLPs have one hidden layer of size 100.

Model parameters are optimized using Adam [27], with $\beta_1 = \beta_2 = 0.9$ and learning rate $5e^{-3}$. The best model parameters are selected according to a score metric on the development set. In particular, we use F_1 for SRL, LAS for parsing, and $MacroF_1$ [28] for the joint task. Other hyper-parameters follow the settings of [22].

3) Objective Function: Cross-entropy loss is used for both tasks. For the joint system, the two tasks share one set of general representation. A simple objective function can be

$$loss(system) = loss(SRL) + loss(Parser)$$

We add another two hyper-parameters to emphasize SRL results and adjust objective function to

$$loss(system) = weight_{(srl)} loss(SRL) + weight_{(parser)} loss(Parser)$$

This strategy works well in practice. For English and Chinese Dataset, $w_{srl} : w_{parser} = 1.8 : 1.0$ is used. For German Dataset, $w_{srl} : w_{parser} = 1.9 : 1.0$ is used. These parameters are tuned on development set.

[1]These sizes are restricted by the pretrained word embeding size.

Table IV
AVERAGED F_1 ON DIFFERENT LENGTHS OF SENTENCES (TEST-DATASET)

Sent Length	<20	21-30	31-40	41-50	>50
Single Model	78.44	88.40	88.43	87.46	83.74
Joint Model	78.62	88.59	88.76	87.94	84.45

B. Results and Discussion

Table II shows the results on English. Syntax-only is the single parisng model and semantic-only is the single SRL model. The results show that joint training enhances the performance of SRL significantly (p-value$<$0.05) and parsing result is comparable with the syntax-only model. For Chinese and German dataset, we have similar findings(Chinese/LAS: 81.70→81.68, Chinese/Sem. F_1:83.47→84.12, German/LAS: 86.39→86.32, German/Sem. F_1:78.28→80.84). These results are in line with the finding of [17].

Results of state-of-the-art joint parsing and SRL models are shown in Table III. With the given the dataset, the overall performance of our joint system outperforms all the CoNLL 2009 systems together with [17]. This performance benefits from the power of the deep encoder and decoder framework and the incorporation heterogeneous attributes from other state-of-the-art systems. The encoder can induce lexical and contextual feature automatically from word and POS tags embeddings.

Table I[2] shows the results of SRL performance. We achieve competitive results with the state-of-the-art ensemble system of Marcheggiani and Titov [9] (89.07% against 89.1%, 78.93% against 78.9%). In order to alleviate of influence of the predicate sense (the accuracy of predicate sense disambiguation of our model is 93.43% on WSJ test data and 82.36% on Brown test data), the results of excluding predicate senses are also showed. Our system shows great generalization ability as the system has improvement by 4.21%(73.22%→77.43%) in comparison with the system of [21] on out-of-domain Brown evaluation.

[2]The system of Roth and Lapata uses automatic predicate for the pipeline system, which is not comparable with other systems which use the gold predicate.

Table IV shows that for different length of sentences. The SRL performance is improved in joint model benefit from deep parameters sharing. Especially for those sentences whose lengths are 40+, the improvement is more significant which benefits from the syntactic information in our joint model.

IV. Conclusion

We investigated deep parameter sharing for joint dependency parsing and semantic role labeling, taking two state-of-the-art models that use the encoder-decoder structure as our baselines. Results demonstrate that syntax information can improve a state-of-the-art semantic role labeler without explicit syntax input significantly. To our knowledge, our model gives the best results on CoNLL09 dataset. The code will be available on author's website.

References

[1] D. Gildea and D. Jurafsky, "Automatic labeling of semantic roles," *Computational linguistics*, vol. 28, no. 3, pp. 245–288, 2002.

[2] C. A. Thompson, R. Levy, and C. D. Manning, "A generative model for semantic role labeling," in *European Conference on Machine Learning*. Springer, 2003, pp. 397–408.

[3] S. Pradhan, K. Hacioglu, W. Ward, J. H. Martin, and D. Jurafsky, "Semantic role chunking combining complementary syntactic views," in *CoNLL*, Ann Arbor, Michigan, June 2005, pp. 217–220.

[4] R. Johansson and P. Nugues, "The effect of syntactic representation on semantic role labeling," in *Coling*, Manchester, UK, August 2008, pp. 393–400.

[5] V. Punyakanok, D. Roth, and W. tau Yih, "The importance of syntactic parsing and inference in semantic role labeling," 2008.

[6] T. Lei, Y. Zhang, L. Màrquez, A. Moschitti, and R. Barzilay, "High-order low-rank tensors for semantic role labeling," in *NAACL-HLT*, Denver, Colorado, May–June 2015, pp. 1150–1160.

[7] N. FitzGerald, O. Täckström, K. Ganchev, and D. Das, "Semantic role labeling with neural network factors," in *EMNLP*, Lisbon, Portugal, September 2015, pp. 960–970.

[8] M. Roth and M. Lapata, "Neural semantic role labeling with dependency path embeddings," in *ACL*, Berlin, Germany, August 2016, pp. 1192–1202.

[9] D. Marcheggiani and I. Titov, "Encoding sentences with graph convolutional networks for semantic role labeling," *arXiv preprint arXiv:1703.04826*, 2017.

[10] J. Zhou and W. Xu, "End-to-end learning of semantic role labeling using recurrent neural networks," in *ACL*, 2015.

[11] Z. Wang, T. Jiang, B. Chang, and Z. Sui, "Chinese semantic role labeling with bidirectional recurrent neural networks," in *EMNLP*, 2015, pp. 1626–1631.

[12] D. Marcheggiani, A. Frolov, and I. Titov, "A simple and accurate syntax-agnostic neural model for dependency-based semantic role labeling," *arXiv preprint arXiv:1701.02593*, 2017.

[13] P. Shi, Z. Teng, and Y. Zhang, "Exploiting mutual benefits between syntax and semantic roles using neural network," in *EMNLP*, Austin, Texas, November 2016, pp. 968–974.

[14] C. Sutton and A. McCallum, "Joint parsing and semantic role labeling," in *CoNLL*, 2005, pp. 225–228.

[15] S. A. Boxwell, D. N. Mehay, and C. Brew, "What a parser can learn from a semantic role labeler and vice versa," in *EMNLP*, 2010, pp. 736–744.

[16] A. Van Den Bosch, R. Morante, and S. Canisius, "Joint learning of dependency parsing and semantic role labeling," *Computational Linguistics in the Netherlands Journal*, vol. 2, pp. 97–117, 2012.

[17] S. Swayamdipta, M. Ballesteros, C. Dyer, and N. A. Smith, "Greedy, joint syntactic-semantic parsing with stack lstms," in *CoNLL*, 2016.

[18] J. Henderson, P. Merlo, G. Musillo, and I. Titov, "A latent variable model of synchronous parsing for syntactic and semantic dependencies," in *CoNLL*, Manchester, England, August 2008, pp. 178–182.

[19] C. Dyer, M. Ballesteros, W. Ling, A. Matthews, and N. A. Smith., "Transition-based dependency parsing with stack long short-term memory," in *Proc. ACL*, 2015.

[20] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *The Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011.

[21] J. Guo, W. Che, H. Wang, T. Liu, and J. Xu, "A unified architecture for semantic role labeling and relation classification," in *COLING*, Osaka, Japan, December 2016, pp. 1264–1274.

[22] T. Dozat and C. D. Manning, "Deep biaffine attention for neural dependency parsing," in *International Conference on Learning Representations*, 2017.

[23] A. Graves, "Supervised sequence labelling with recurrent neural networks," 2008.

[24] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *EMNLP*, Doha, Qatar, October 2014, pp. 1532–1543.

[25] W. Ling, C. Dyer, A. W. Black, and I. Trancoso, "Two/too simple adaptations of word2vec for syntax problems," in *NAACL-HLT*, Denver, Colorado, 2015, pp. 1299–1304.

[26] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *International Conference on Learning Representations*, 2013.

[27] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations*, 2014.

[28] J. Hajič, M. Ciaramita, R. Johansson, D. Kawahara, M. A. Martí, L. Màrquez, A. Meyers, J. Nivre, S. Padó, J. Štěpánek *et al.*, "The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages," in *CoNLL*, 2009, pp. 1–18.