

Establishing Strong Baselines for the New Decade: Sequence Tagging, Syntactic and Semantic Parsing with BERT

Han He

Computer Science
Emory University
Atlanta GA 30322, USA
han.he@emory.edu

Jinho D. Choi

Computer Science
Emory University
Atlanta GA 30322, USA
jinho.choi@emory.edu

Abstract

This paper presents new state-of-the-art models for three tasks, part-of-speech tagging, syntactic parsing, and semantic parsing, using the cutting-edge contextualized embedding framework known as BERT. For each task, we first replicate and simplify the current state-of-the-art approach to enhance its model efficiency. We then evaluate our simplified approaches on those three tasks using token embeddings generated by BERT. 12 datasets in both English and Chinese are used for our experiments. The BERT models outperform the previously best-performing models by 2.5% on average (7.5% for the most significant case). All models and source codes are available in public so that researchers can improve upon and utilize them to establish strong baselines for the next decade. We also provide a dedicated error analysis and extensive dissections in <https://arxiv.org/abs/1908.04943>.

1 Introduction

It is no exaggeration to say that word embeddings trained by vector-based language models (Mikolov et al. 2013; Pennington, Socher, and Manning 2014; Bojanowski et al. 2017) have changed the game of NLP once and for all. These pre-trained word embeddings trained on large corpus improve downstream tasks by encoding rich word semantics into vector space. However, word senses are ignored in these earlier approaches such that a unique vector is assigned to each word, neglecting polysemy from the context.

Recently, contextualized embedding approaches emerge with advanced techniques to dynamically generate word embeddings from different contexts. To address polysemous words, Peters et al. (2018) introduce ELMo, which is a word-level Bi-LSTM language model. Akbik, Blythe, and Vollgraf (2018) apply a similar approach to the character-level, called Flair, while concatenating the hidden states corresponding to the first and the last characters of each word to build the embedding of that word. Apart from these unidirectional recurrent language models, Devlin et al. (2018) replace the transformer decoder from Radford et al. (2018) with a bidirectional transformer encoder, then train the BERT system

on 3.3B word corpus. After scaling the model size to hundreds of millions parameters, BERT brings markedly huge improvement to a wide range of tasks without substantial task-specific modifications.

In this paper, we verify the effectiveness and conciseness of BERT by first generating token-level embeddings from it, then integrating them to task-oriented yet efficient model structures (Section 3). With careful investigation and engineering, our simplified models significantly outperform many of the previous state-of-the-art models, achieving the highest scores for 11 out of 12 datasets (Section 4).

To the best of our knowledge, it is the first work that tightly integrates BERT embeddings to these three downstream tasks and present such high performance. All our resources including the models and the source codes are publicly available.¹

2 Related Work

Our work builds off recent work in representation learning, tagging and parsing. To learn contextualized representations, BERT (Devlin et al. 2018) employ masked LM to jointly condition on both left and right contexts, showing impressive improvement in various tasks. As a trend for tagging, fine grained features often result in better performance. These features include the morphological and contextual information from contextual string embeddings (Akbik, Blythe, and Vollgraf 2018), the representations from both string and token based character Bi-LSTM language models (Bohnet et al. 2018), and the ensemble of multilingual BERT and conventional representations (Heinzerling and Strube 2019). Among parsing community, graph-based parsers (Dozat and Manning 2017; Clark et al. 2018; Ma et al. 2018) resurge due to GPU parallelization. Recently, (Zhou and Zhao 2019) achieved impressive results by jointly learning constituency and dependency parsing with BERT.

3 Approach

3.1 Token-level Embeddings with BERT

BERT splits each token into subwords using WordPiece (Wu et al. 2016), which do not necessarily reflect any morphology in linguistics. For example, *Rainwater* gets split into

¹<https://github.com/emorynlp/bert-2019>

Rain+water, whereas *running* or *rapidly* remain unchanged although typical morphology would split them into *run+ing* and *rapid+ly*. To obtain token-level embeddings for tagging and parsing, the following two methods are experimented:

Last Embedding Since the subwords from each token are trained to predict one another during language modeling, their embeddings must be correlated. Thus, one way is to pick the embedding of the last subword as a representation of the token.

Average Embedding For a compound word like ‘dog-house’ that gets split into ‘dog’ and ‘##house’, the last subword does not necessarily convey the key meaning of the token. Hence, another way is to take the average embedding of the subwords.

Model	In-domain	Out-of-domain
BERT _{BASE} : LAST	86.7	79.5
BERT _{BASE} : AVERAGE	86.7	79.8
BERT _{LARGE} : LAST	86.8	79.4
BERT _{LARGE} : AVERAGE	86.4	79.5

Table 1: Results from the PSD semantic parsing task (§4.3) using the last and average embedding methods.

Table 1 shows results from a semantic parsing task, PSD (Section 4.3), using the last and average embedding methods with BERT_{BASE} and BERT_{LARGE} models.² The average method is chosen for all our experiments since it gives a marginal advantage to the out-of-domain dataset.

3.2 Input Embeddings with BERT

While Devlin et al. (2018) report that adding just an additional output layer to the BERT encoder can build powerful models in a wide range of tasks, its computational cost is too high. Thus, we separate the BERT architecture from downstream models, and feed pre-generated BERT embeddings, \mathbf{e}^{BERT} , as input to task-specific encoders:

$$\mathcal{F}^i = \text{Encoder}(\mathbf{X} \oplus \mathbf{e}^{\text{BERT}})$$

Alternatively, BERT embeddings can be concatenated with the output of a certain hidden layer:

$$\mathcal{F}^h = \text{Encoder}_{[h]}(\text{Encoder}_{[1:h]}(\mathbf{X}) \oplus \mathbf{e}^{\text{BERT}})$$

where $\text{Encoder}_{[1:h]}$ denotes for encoder layers from 1 to h , and $\text{Encoder}_{[h]}$ denotes for layers from h to the last one. Table 2 shows results from the PSD semantic parsing task (Section 4.3) using the average method from Section 3.1. \mathcal{F}^i shows a slight advantage for both BERT_{BASE} and BERT_{LARGE} over \mathcal{F}^h ; thus, it is chosen for all our experiments.

3.3 Bi-LSTM-CRF for Tagging

For sequence tagging, the Bi-LSTM-CRF (Huang, Xu, and Yu 2015) with the Flair embeddings (Akbi, Blythe, and Vollgraf 2018), is used to establish a baseline for English.

²BERT_{BASE} uses 12 layers, 768 hidden cells, 12 attention heads, and 110M parameters, while BERT_{LARGE} uses 24 layers, 1024 hidden cells, 16 attention heads, and 340M parameters. Both models are uncased, since they are reported to achieve high scores for all tasks except for NER (Devlin et al. 2018).

Model	In-domain	Out-of-domain
BERT _{BASE} : \mathcal{F}^i	86.7	79.8
BERT _{BASE} : \mathcal{F}^h	86.5	79.5
BERT _{LARGE} : \mathcal{F}^i	86.4	79.5
BERT _{LARGE} : \mathcal{F}^h	85.9	79.1

Table 2: Results from the PSD semantic parsing task (Section 4.3) using \mathcal{F}^i and \mathcal{F}^h .

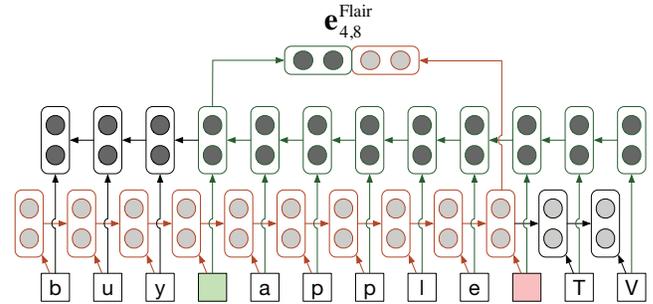


Figure 1: Generating the Flair embedding for ‘apple’.

Given a token w in a sequence where c_i and c_j are the starting and ending characters of w (i and j are the character offsets; $i \leq j$), the Flair embedding of w is generated by concatenating two hidden states of c_{j+1} from the forward LSTM and c_{i-1} from the backward LSTM (Figure 1):

$$\mathbf{e}_{i,j}^{\text{Flair}} = \mathbf{h}^f(c_{j+1}) \oplus \mathbf{h}^b(c_{i-1})$$

$\mathbf{e}_{i,j}^{\text{Flair}}$ is then concatenated with a pre-trained token embedding of w and fed into the Bi-LSTM-CRF. In our approach, we present two models, one substituting the Flair and pre-trained embeddings with BERT, and the other concatenating BERT to the other embeddings. Note that variational dropout is not used in our approach to reduce complexity.

3.4 Biaffine Attention for Syntactic Parsing

A simplified variant of the biaffine parser (Dozat and Manning 2017) is used for syntactic parsing (Figure 2). Compared to the original version, the trainable word embeddings are removed and lemmas are used instead of forms to retrieve pre-trained embeddings in our version, leading to less complexity yet better generalization. Given the i 'th token w_i , the feature vector is created by concatenating its pre-trained lemma embedding $\mathbf{e}_i^{\text{LEM}}$, POS embedding $\mathbf{e}_i^{\text{POS}}$ learned during training and the representation $\mathbf{e}_i^{\text{BERT}}$ from the last layer of BERT. This feature vector is fed into Bi-LSTM, generating two recurrent states \mathbf{r}_i^f and \mathbf{r}_i^b :

$$\begin{aligned} \mathbf{r}_i^f &= \text{LSTM}^{\text{forward}}(\mathbf{e}_i^{\text{LEM}} \oplus \mathbf{e}_i^{\text{POS}} \oplus \mathbf{e}_i^{\text{BERT}}) \\ \mathbf{r}_i^b &= \text{LSTM}^{\text{backward}}(\mathbf{e}_i^{\text{LEM}} \oplus \mathbf{e}_i^{\text{POS}} \oplus \mathbf{e}_i^{\text{BERT}}) \end{aligned}$$

Two multi-layer perceptrons (MLP) are then used to extract features for w_i being a head $\mathbf{h}_i^{\text{arc-h}}$ or a dependent $\mathbf{h}_i^{\text{arc-d}}$, and two additional MLP are used to extract $\mathbf{h}_i^{\text{rel-h}}$ and $\mathbf{h}_i^{\text{rel-d}}$ for labeled dependency parsing:

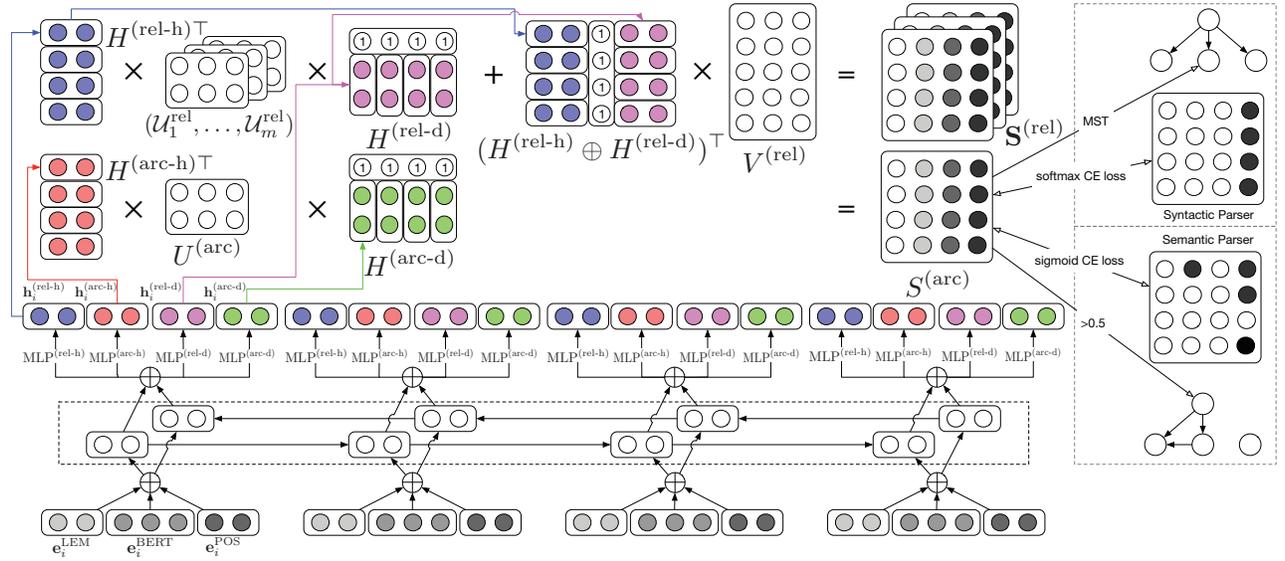


Figure 2: Biaffine attention parser

$$\begin{aligned} \mathbf{h}_i^{(\text{arc-h})} &= \text{MLP}^{(\text{arc-h})}(\mathbf{r}_i^f \oplus \mathbf{r}_i^b) \in \mathbb{R}^{k \times 1} \\ \mathbf{h}_i^{(\text{arc-d})} &= \text{MLP}^{(\text{arc-d})}(\mathbf{r}_i^f \oplus \mathbf{r}_i^b) \in \mathbb{R}^{k \times 1} \\ \mathbf{h}_i^{(\text{rel-h})} &= \text{MLP}^{(\text{rel-h})}(\mathbf{r}_i^f \oplus \mathbf{r}_i^b) \in \mathbb{R}^{l \times 1} \\ \mathbf{h}_i^{(\text{rel-d})} &= \text{MLP}^{(\text{rel-d})}(\mathbf{r}_i^f \oplus \mathbf{r}_i^b) \in \mathbb{R}^{l \times 1} \end{aligned}$$

$\mathbf{h}_{1..n}^{\text{arc-h}}$ are stacked into a matrix $H^{\text{arc-h}}$ with a bias for the prior probability of each token being a head, and $\mathbf{h}_{1..n}^{\text{arc-d}}$ are stacked into another matrix $H^{\text{arc-d}}$ as follows (n : # of tokens, $U^{(\text{arc})} \in \mathbb{R}^{k \times (k+1)}$):

$$\begin{aligned} H^{(\text{arc-h})} &= (\mathbf{h}_1^{(\text{arc-h})}, \dots, \mathbf{h}_n^{(\text{arc-h})}) \in \mathbb{R}^{k \times n} \\ H^{(\text{arc-d})} &= (\mathbf{h}_1^{(\text{arc-d})}, \dots, \mathbf{h}_n^{(\text{arc-d})}) \oplus \mathbf{1} \in \mathbb{R}^{(k+1) \times n} \\ S^{(\text{arc})} &= H^{(\text{arc-h})\top} \cdot U^{(\text{arc})} \cdot H^{(\text{arc-d})} \in \mathbb{R}^{n \times n} \end{aligned}$$

$S^{(\text{arc})}$ is called a bilinear classifier that predicts head words. Arc labels are predicted by another biaffine classifier $S^{(\text{rel})}$, which combines m bilinear classifiers for multi-classification (m : # of labels, $U^{(\text{rel})} \in \mathbb{R}^{l \times (l+1)}$, $V^{(\text{rel})} \in \mathbb{R}^{(2 \cdot l + 1) \times m}$):

$$\begin{aligned} H^{(\text{rel-h})} &= (\mathbf{h}_1^{(\text{rel-h})}, \dots, \mathbf{h}_n^{(\text{rel-h})}) \in \mathbb{R}^{l \times n} \\ H^{(\text{rel-d})} &= (\mathbf{h}_1^{(\text{rel-d})}, \dots, \mathbf{h}_n^{(\text{rel-d})}) \oplus \mathbf{1} \in \mathbb{R}^{(l+1) \times n} \\ \mathcal{U}_i^{\text{rel}} &= H^{(\text{rel-h})\top} \cdot U_i^{(\text{rel})} \cdot H^{(\text{rel-d})} \in \mathbb{R}^{n \times n} \\ \mathbf{S}^{(\text{rel})} &= (\mathcal{U}_1^{\text{rel}}, \dots, \mathcal{U}_m^{\text{rel}}) \\ &\quad + (H^{(\text{rel-h})} \oplus H^{(\text{rel-d})})\top \cdot V^{(\text{rel})} \in \mathbb{R}^{m \times n \times n} \end{aligned}$$

During training, softmax cross-entropy is used to optimize $S^{(\text{arc})}$ and $S^{(\text{rel})}$. Note that for the optimization of $S^{(\text{rel})}$, gold heads are used instead of predicted ones. During decoding, a maximum spanning tree algorithm is adopted for searching the optimal tree based on the scores in $S^{(\text{arc})}$.

3.5 Biaffine Attention for Semantic Parsing

Dozat and Manning (2018) adapted their original biaffine parser to generate dependency graphs for semantic parsing, where each token can have zero to many heads. Since the tree structure is no longer guaranteed, sigmoid cross-entropy is used instead so that independent binary predictions can be made for every token to be considered a head of any other token. The label predictions are made as outputting the labels with the highest scores in $S^{(\text{rel})}$ once arc predictions are made, as illustrated in Figure 2.

This updated implementation is further simplified in our approach by removing the trainable word embeddings, the character-level feature detector, and their corresponding linear transformers. Moreover, instead of using the interpolation between the head and label losses, equal weights are applied to both losses, reducing hyperparameters to tune.

4 Experiments

Three sets of experiments are conducted to evaluate the impact of our approaches using BERT (Section 3). For sequence tagging (Section 4.1), part-of-speech tagging is chosen where each token gets assigned with a fine-grained POS tag. For syntactic parsing (Section 4.2), dependency parsing is chosen where each token finds exactly one head, generating a tree per sentence. For semantic parsing (Section 4.3), semantic dependency parsing is chosen where each token finds zero to many heads, generating a graph per sentence. Every task is tested on both English and Chinese to ensure robustness across languages. Standard datasets are adapted to all experiments for fair comparisons to many previous approaches. All our models are experimented three times and average scores with standard deviations are reported.

4.1 Sequence Tagging

For part-of-speech tagging, the Wall Street Journal corpus from the Penn Treebank 3 (Marcus, Marcinkiewicz, and San-

	ALL	OOV
Ma and Hovy (2016)	97.55	93.45
Ling et al. (2015)	97.78	n/a
Clark et al. (2018)	97.79	n/a
Akbik, Blythe, and Vollgraf (2018)	97.85 (± 0.01)	n/a
Bohnet et al. (2018)	97.96	n/a
Baseline	97.70 (± 0.05)	92.44 (± 0.03)
Baseline \ BERT _{BS}	96.96 (± 0.06)	91.23 (± 0.22)
Baseline \ BERT _{LG}	96.96 (± 0.05)	91.26 (± 0.25)
Baseline + BERT _{BS}	97.68 (± 0.06)	92.69 (± 0.32)
Baseline + BERT _{LG}	97.67 (± 0.02)	93.01 (± 0.27)

(a) Results from the English test set. BERT_{BS} and BERT_{LG} are BERT’s uncased base and cased large models, respectively.

	ALL	OOV
Zhang et al. (2015)	94.47*	n/a
Zhang et al. (2014)	94.62*	n/a
Kurita, Kawahara, and Kurohashi (2017)	94.84*	n/a
Hatori et al. (2011)	94.64	n/a
Wang and Xue (2014)	96.0	n/a
Baseline	95.65 (± 0.26)	83.57 (± 0.55)
Baseline \ BERT	96.38 (± 0.15)	88.13 (± 0.72)
Baseline + BERT	97.25 (± 0.18)	90.53 (± 0.91)

(b) Results from the Chinese test set. * are evaluated on the character-level due to automatic segmentation, so their results are not directly comparable to ours but reported for reference.

Table 3: Test results for part-of-speech tagging, where token-level accuracy is used as the evaluation metric. ALL: all tokens, OOV: out-of-vocabulary tokens.

torini 1993) is used for English, and the Penn Chinese Treebank 5.1 (Xue et al. 2005) is used for Chinese. Table 3 shows tagging results on the test sets.

For English, the baseline is our replication of the Flair model using both GloVe and Flair embeddings (Section 3.3). It shows a slightly lower accuracy, -0.15%, than the original model (Akbik, Blythe, and Vollgraf 2018) due to the lack of variational dropout. \BERT substitutes GloVe and Flair with BERT embeddings, and +BERT uses all three types of embeddings. The baseline outperforms all BERT models for the ALL test, implying that Flair’s Bi-LSTM character language model is more effective than BERT’s word-piece approach. No significant difference is found between BERT_{BS} and BERT_{LG}. However, an interesting trend is found in the OOV test, where the +BERT_{LG} model shows good improvement over the baseline. This implies that BERT embeddings can still contribute to the Flair model for OOV although the CNN character language model from Ma and Hovy (2016) is marginally more effective than +BERT for OOV tokens.

For Chinese, the Bi-LSTM-CRF model with FastText embeddings is used for baseline (Sec. 3.3). \BERT that substitutes FastText embeddings with BERT and +BERT that adds BERT embeddings to the baseline show progressive improvement over the prior model for both the ALL and OOV tests. +BERT gives an accuracy that is 1.25% higher than the previous state-of-the-art using joint-learning between tagging and parsing (Wang and Xue 2014).

4.2 Syntactic Parsing

Table 4 shows parsing results on the test sets. The same datasets for POS tagging are also used for syntactic parsing.

	UAS	LAS
Dozat and Manning (2017)	95.74	94.08
Kuncoro et al. (2017)	95.8	94.6
Ma et al. (2018)	95.87	94.19
Choe and Charniak (2016)	95.9	94.1
Clark et al. (2018)	96.6	95.0
Zhou and Zhao (2019)	97.20	95.72
Baseline	95.78 (± 0.04)	94.04 (± 0.04)
Baseline \ BERT	96.76 (± 0.09)	95.27 (± 0.13)
Baseline + BERT	96.79 (± 0.08)	95.29 (± 0.12)

(a) Results from the English test set.

	UAS	LAS
Dozat and Manning (2017)	89.30	88.23
Ma et al. (2018)	90.59	89.29
Baseline	91.02 (± 0.10)	89.89 (± 0.09)
Baseline \ BERT	93.21 (± 0.06)	92.21 (± 0.05)
Baseline + BERT	93.34 (± 0.21)	92.29 (± 0.22)

(b) Results from the Chinese test set.

Table 4: Test results for dependency parsing, where unlabeled and labeled attachment scores (UAS and LAS) are used as the evaluation metrics. Scores in *italic* are joint learning results.

Our simplified version of the biaffine parser (Section 3.4) is used for baseline, where GloVe and FastText embeddings are used for English and Chinese, respectively. The baseline model gives a comparable result to the original model (Dozat and Manning 2017) for English, yet shows a notably better result for Chinese, which can be due to higher quality embeddings from FastText. \BERT substitutes the pre-trained embeddings with BERT and +BERT adds BERT embeddings to the baseline. Moreover, BERT’s uncased base model is used for English.

Between \BERT and +BERT, no significant difference is found, implying that those pre-trained embeddings are not so useful when coupled with BERT. All BERT models show significant improvement over the baselines for both languages, and outperform the previous state-of-the-art approaches using cross-view training (Clark et al. 2018) and stack-pointer networks (Ma et al. 2018) by 0.29% and 3% in LAS for English and Chinese, respectively. Considering the simplicity of our +BERT models, these results are remarkable.

4.3 Semantic Parsing

The English dataset from the SemEval 2015 Task 18: Broad-Coverage Semantic Dependency Parsing (Oepen et al. 2015) and the Chinese dataset from the SemEval 2016 Task 9: Chinese Semantic Dependency Parsing (Che et al. 2016) are used for semantic dependency parsing.

Table 5 shows the English results on the test sets. The baseline, \BERT, and +BERT models are similar to the ones in Section 4.2, except they use the sigmoid instead of the softmax function in the output layer to accept multiple heads (Section 3.5). Our baseline is a simplified version of Dozat and Manning (2018); its average scores are 1.2% higher and 1.0% lower than the original model for ID and OOD, due to different hyperparameter settings. +BERT shows good improvement over \BERT for both test sets, implying that BERT

embeddings are complementary to those pre-trained embeddings, and surpasses the previous state-of-the-art scores by 3% and 2% for ID and OOD, respectively.

	DM	PAS	PSD	AVG
Du et al. (2015)	89.1	91.3	75.7	85.3
Almeida and Martins (2015)	89.4	91.7	77.6	86.2
Wang et al. (2018)	90.3	91.7	78.6	86.9
Peng, Thomson, and Smith (2017)	90.4	92.7	78.5	87.2
Dozat and Manning (2018)	93.7	93.9	81.0	89.5
Che et al. (2019)	92.9	94.4	81.6	89.6
Baseline	92.48	94.56	85.00	90.68
Baseline \ BERT	94.36	96.03	86.59	92.33
Baseline + BERT	94.57	96.13	86.80	92.50

(a) Results from the in-domain (ID) test sets.

	DM	PAS	PSD	AVG
Du et al. (2015)	81.8	87.2	73.3	80.8
Almeida and Martins (2015)	83.8	87.6	76.2	82.5
Wang et al. (2018)	84.9	87.6	75.9	82.8
Peng, Thomson, and Smith (2017)	85.3	89.0	76.4	83.6
Dozat and Manning (2018)	88.9	90.6	79.4	86.3
Che et al. (2019)	89.2	92.4	81.0	87.5
Baseline	86.98	91.35	77.28	85.34
Baseline \ BERT	90.49	94.31	79.31	88.07
Baseline + BERT	90.86	94.38	79.48	88.21

(b) Results from the out-of-domain (OOD) test sets.

Table 5: Test results for semantic dependency parsing in English; labeled dependency F1 scores are used as the evaluation metrics. DM: DELPH-IN dependencies, PAS: Enju dependencies, PSD: Prague dependencies, AVG: macro-average of (DM, PAS, PSD).

Table 6 shows the Chinese results on the test sets. No significant difference is found between \BERT and +BERT. +BERT significantly outperforms the previous state-of-the-art by 4% and 7.5% in LF for NEWS and TEXT, which confirms that BERT embeddings are very effective for semantic dependency parsing in both English and Chinese.

	NEWS		TEXT	
	UF	LF	UF	LF
Artsymenia, Dounar, and Yermakovich (2016)	77.64	59.06	82.41	68.59
Wang et al. (2018)	81.14	63.30	85.71	72.92
Baseline	80.51	64.90	88.06	77.28
Baseline \ BERT	82.91	67.17	90.83	80.46
Baseline + BERT	82.92	67.27	91.10	80.41

Table 6: Test results for semantic dependency parsing in Chinese, where unlabeled and labeled dependency F1 scores (UF and LF) are used as the evaluation metrics. NEWS: newswire, TEXT: textbook.

5 Conclusion

In this paper, we describe our methods of exploiting BERT as token-level embeddings for tagging and parsing tasks. Our experiments empirically show that tagging and parsing can be tackled using much simpler models without losing accuracy. Out of 12 datasets, our approaches with BERT have established new state-of-the-art for 11 of them. As the first work of employing BERT with syntactic and semantic

parsing, our approach is much simpler yet more accurate than the previous state-of-the-art.

Acknowledgments

We gratefully acknowledge the support of the AWS Machine Learning Research Awards (MLRA) and the GluonNLP team (Guo et al. 2019).

References

- Akbik, A.; Blythe, D.; and Vollgraf, R. 2018. Contextual String Embeddings for Sequence Labeling. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING’18*, 1638–1649.
- Almeida, M. S. C., and Martins, A. F. T. 2015. Lisbon: Evaluating TurboSemanticParser on Multiple Languages and Out-of-Domain Data. In *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval’15*, 970–973.
- Artsymenia, A.; Dounar, P.; and Yermakovich, M. 2016. IHS-RD-Belarus at SemEval-2016 Task 9: Transition-based Chinese Semantic Dependency Parsing with Online Reordering and Bootstrapping. In *Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval’16*, 1207–1211.
- Bohnet, B.; McDonald, R.; Simões, G.; Andor, D.; Pitler, E.; and Maynez, J. 2018. Morphosyntactic Tagging with a Meta-BiLSTM Model over Context Sensitive Token Encodings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL’18*, 2642–2652.
- Bojanowski, P.; Grave, E.; Joulin, A.; and Mikolov, T. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics* 5:135–146.
- Che, W.; Shao, Y.; Liu, T.; and Ding, Y. 2016. SemEval-2016 Task 9: Chinese Semantic Dependency Parsing. In *Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval’16*, 1074–1080.
- Che, W.; Dou, L.; Xu, Y.; Wang, Y.; Liu, Y.; and Liu, T. 2019. HIT-SCIR at MRP 2019: A unified pipeline for meaning representation parsing via efficient training and effective encoding. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, 76–85. Hong Kong: Association for Computational Linguistics.
- Choe, D. K., and Charniak, E. 2016. Parsing as Language Modeling. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2331–2336.
- Clark, K.; Luong, M.-T.; Manning, C. D.; and Le, Q. 2018. Semi-Supervised Sequence Modeling with Cross-View Training. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP’18*, 1914–1925.
- Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv* 1810.04805.
- Dozat, T., and Manning, C. D. 2017. Deep Biaffine Attention for Neural Dependency Parsing. In *Proceedings of the 5th International Conference on Learning Representations*.

- Dozat, T., and Manning, C. D. 2018. Simpler but More Accurate Semantic Dependency Parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, ACL'18, 484–490.
- Du, Y.; Zhang, F.; Zhang, X.; Sun, W.; and Wan, X. 2015. Peking: Building Semantic Dependency Graphs with a Hybrid Parser. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, SemEval'15, 927–931.
- Guo, J.; He, H.; He, T.; Lausen, L.; Li, M.; Lin, H.; Shi, X.; Wang, C.; Xie, J.; Zha, S.; Zhang, A.; Zhang, H.; Zhang, Z.; Zhang, Z.; and Zheng, S. 2019. GluonCV and GluonNLP: Deep Learning in Computer Vision and Natural Language Processing. *arXiv preprint arXiv:1907.04433*.
- Hatori, J.; Matsuzaki, T.; Miyao, Y.; and Tsujii, J. 2011. Incremental Joint POS Tagging and Dependency Parsing in Chinese. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, IJCNLP'11, 1216–1224.
- Heinzerling, B., and Strube, M. 2019. Sequence tagging with contextual and non-contextual subword representations: A multilingual evaluation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 273–291. Florence, Italy: Association for Computational Linguistics.
- Huang, Z.; Xu, W.; and Yu, K. 2015. Bidirectional LSTM-CRF Models for Sequence Tagging. *arXiv* 1508.01991.
- Kuncoro, A.; Ballesteros, M.; Kong, L.; Dyer, C.; Neubig, G.; and Smith, N. A. 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*, EACL'17, 1249–1258.
- Kurita, S.; Kawahara, D.; and Kurohashi, S. 2017. Neural Joint Model for Transition-based Chinese Syntactic Analysis. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, ACL'17, 1204–1214.
- Ling, W.; Dyer, C.; Black, A. W.; Trancoso, I.; Fernandez, R.; Amir, S.; Marujo, L.; and Luis, T. 2015. Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP'15, 1520–1530.
- Ma, X., and Hovy, E. 2016. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, ACL'16, 1064–1074.
- Ma, X.; Hu, Z.; Liu, J.; Peng, N.; Neubig, G.; and Hovy, E. 2018. Stack-Pointer Networks for Dependency Parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, ACL'18, 1403–1414.
- Marcus, M. P.; Marcinkiewicz, M. A.; and Santorini, B. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics* 19(2):313–330.
- Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems*, NIPS'13, 3111–3119.
- Oepen, S.; Kuhlmann, M.; Miyao, Y.; Zeman, D.; Cinkova, S.; Flickinger, D.; Hajic, J.; and Uresova, Z. 2015. SemEval 2015 Task 18: Broad-Coverage Semantic Dependency Parsing. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, SemEval'15, 915–926.
- Peng, H.; Thomson, S.; and Smith, N. A. 2017. Deep Multitask Learning for Semantic Dependency Parsing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, ACL'17, 2037–2048.
- Pennington, J.; Socher, R.; and Manning, C. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the conference on empirical methods in natural language processing*, EMNLP'14, 1532–1543.
- Peters, M.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep Contextualized Word Representations. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL'18, 2227–2237.
- Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving Language Understanding by Generative Pre-training. Technical report, OpenAI.
- Wang, Z., and Xue, N. 2014. Joint POS Tagging and Transition-based Constituent Parsing in Chinese with Non-local Features. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, ACL'14, 733–742.
- Wang, Y.; Che, W.; Guo, J.; and Liu, T. 2018. A Neural Transition-Based Approach for Semantic Dependency Graph Parsing. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, AAAI'18, 5561–5568.
- Wu, Y.; Schuster, M.; Chen, Z.; Le, Q. V.; Norouzi, M.; Macherey, W.; Krikun, M.; Cao, Y.; Gao, Q.; Macherey, K.; et al. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *arXiv* 1609.08144.
- Xue, N.; Xia, F.; Chiou, F.-d.; and Palmer, M. 2005. The Penn Chinese TreeBank: Phrase Structure Annotation of a Large Corpus. *Natural Language Engineering* 11(2):207–238.
- Zhang, M.; Zhang, Y.; Che, W.; and Liu, T. 2014. Character-Level Chinese Dependency Parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, ACL'14, 1326–1336.
- Zhang, Y.; Li, C.; Barzilay, R.; and Darwish, K. 2015. Randomized Greedy Inference for Joint Segmentation, POS Tagging and Dependency Parsing. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL'15, 42–52.
- Zhou, J., and Zhao, H. 2019. Head-driven phrase structure grammar parsing on Penn treebank. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2396–2408. Florence, Italy: Association for Computational Linguistics.