Type Enhanced BERT for Correcting NER Errors

Kuai Li[*], **Chen Chen**[*], **Tao Yang, Tianming Du, Peijie Yu, Dong Du and Feng Zhang**
Machine Learning Platform Department, Tencent
{kuaili, chenzchen, rigorosyang}@tencent.com
{blackdu, peijieyu, dongdu, jayzhang}@tencent.com

Abstract

We introduce the task of correcting named entity recognition (NER) errors without retraining the model. After a NER model is trained and deployed in production, it makes prediction errors, which usually need to be fixed quickly. To address this problem, we firstly construct a gazetteer containing named entities and corresponding possible entity types. And then, we propose type-enhanced BERT (TyBERT), a method that integrates the named entity's type information into BERT by an adapter layer. When errors are identified, we can repair the model by updating the gazetteer. In other words, the gazetteer becomes a trigger to control the NER model's output. The experiment results in multiple corpus show the effectiveness of our method, which outperforms strong baselines.

1 Introduction

Named entity recognition (NER) is the task of identifying spans that belong to particular categories, such as *person*, *location*, *organization*, etc. The NER task is important in the information extraction area and NER models are widely deployed in real production systems (Yadav and Bethard, 2019). In recent years, many neural-based methods were proposed to push NER accuracy by designing novel network architectures (Lample et al., 2016; Devlin et al., 2018; Straková et al., 2019; Xue et al., 2022) or incorporating external knowledge (Liu et al., 2019; Wang et al., 2021). Unfortunately, all approaches are still far from perfect. When the model is served in production, we may still encounter recognition errors (e.g., bad cases).

Typically, to fix those bad cases, model developers need to (1) annotate the input sentences causing errors with correct labels, (2) combine newly annotated sentences with existing training data, (3) train and tune a new model with the new training data and held-out evaluation data, and finally (4) deploy the new model in production. As one can tell, the above process is time-consuming, and cannot meet the requirement of fixing urgent errors quickly in a real production environment.

[*]Equal contribution.

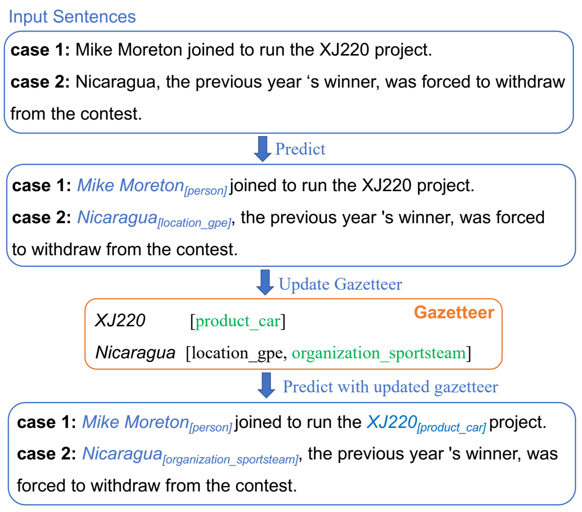
Input Sentences
case 1: Mike Moreton joined to run the XJ220 project.
case 2: Nicaragua, the previous year 's winner, was forced to withdraw from the contest.
Predict
case 1: Mike Moreton[person] joined to run the XJ220 project.
case 2: Nicaragua[location_gpe], the previous year 's winner, was forced to withdraw from the contest.
Update Gazetteer
Gazetteer
XJ220 [product_car]
Nicaragua [location_gpe, organization_sportsteam]
Predict with updated gazetteer
case 1: Mike Moreton[person] joined to run the XJ220[product_car] project.
case 2: Nicaragua[organization_sportsteam], the previous year 's winner, was forced to withdraw from the contest.

Figure 1: Two motivating examples and the overall process to fix errors by updating the gazetteer.

Therefore, in this paper, we aim to tackle the problem of how to correct NER errors without retraining models.[1] Taking case 1 and 2 from Figure 1 as examples, there are two kinds of common NER errors when we train and evaluate a model in the English Few-NERD (Ding et al., 2021) corpus: (1) the model fails to recognize the span "XJ220" as a named entity; (2) the model correctly identifies the boundary of the named entity "Nicaragua", but assigns a wrong entity type to it.

For the first error, we find the span "XJ220" never appears in the training dataset. Therefore, it is difficult for the model to classify this span as a

[1]One may argue that this task is trivial if we simply construct a database containing sentences with recognition errors, and then always look up the database before requesting the NER model. But this naive approach is not sustainable as the number of bad cases grows, and the database cannot generalize to any unseen cases.

named entity with limited context. For the second error, the mention "Nicaragua" is found in the training dataset, but it is labeled with a different type *location*. Because of the incomplete type information, the model mistakenly classifies the mention as type *location*, though the correct label should be *organization_sportsteam*.

The above examples suggest that if we have proper type information about the span, the model may correct its mistakes, even without re-training. It motivates us to propose the Type Enhanced BERT (TyBERT) method that combines BERT with type information from a gazetteer.

As shown in Figure 1, the gazetteer is a list of pairs of spans and possible entity types. During training, we first look up spans from the gazetteer in training examples, and then integrate the matched span's type information into BERT layers by an adapter layer. In the inference stage, the test examples are processed in the same way. In such a manner, the model is tied to the gazetteer, which will play an important role when the model makes predictions. When encountering the aforementioned two kinds of errors, we can update the gazetteer: we insert a new named entity "XJ220" with the expected type *product_car*, and add a new type *organization_sportsteam* for the existing named entity "Nicaragua". Moreover, we introduce a noise rate parameter λ to randomly add some noise to the gazetteer. This parameter serves as an adjuster to balance the strength of the gazetteer and the generalization ability of the model.

To our knowledge, this is the first work to systematically study how to improve NER models without re-training models. When evaluated in four NER corpus in English and Chinese, the proposed method performs well in fixing errors and outperforms strong baselines. Our code and data will be released after publication.

2 Related Work

Our work is influenced by existing methods which combine both neural networks and lexicons or gazetteers for NER. For example, Zhang and Yang (2018) proposed a lattice-structured LSTM encoding both a sequence of input characters and potential words that match a pre-gathered lexicon. Sui et al. (2019) presented Collaborative Graph Network to solve the challenges of self-matched lexical words and the nearest contextual lexical words. Gui et al. (2019) aimed to alleviate the word ambiguity issue by a lexicon-based graph neural network with global semantics. Lin et al. (2019) designed an attentive neural network to explicitly model the mention-context association and gazetteer network to effectively encode name regularity of mentions only using gazetteers. Li et al. (2020) introduced a flat-lattice Transformer to incorporate lexicon information for Chinese NER. Meng et al. (2021) invented GEMNET to include a Contextual Gazetteer Representation encoder, combined with a novel Mixture-of-Expert gating network to conditionally utilize this information alongside any word-level model. Fetahu et al. (2022) invented an approach of using a token-level gating layer to augment pre-trained multilingual transformers with gazetteers from a target domain. Finally, Liu et al. (2021) proposed Lexicon Enhanced BERT (LEBERT) for Chinese sequence labeling, which integrates external lexicon knowledge into BERT layers directly by a Lexicon Adapter layer.

It is worth noting that none of the previous works can be directly applied for correcting NER models without re-training. For example, LEBERT requires learning lexicon embeddings in the adapter layer. If we want to add a new span in the lexicon to fix a bad case, the model has to be re-trained to learn the new span's embedding.

3 Method

3.1 Gazetteer Construction

As noted before, the gazetteer contains a list of named entities and their possible entity types. In this paper, we collect the gazetteer solely from NER annotations in the dataset. For instance, given the following two annotated sentences from the Few-NERD corpus:

London$_{[art-music]}$ *is the fifth album by the British*$_{[location-gpe]}$ *rock band.*

He is domiciled in London$_{[location-gpe]}$.

We will construct the following gazetteer:

London [art-music, location-gpe]

British [location-gpe]

We employ this simple approach because it is applicable for NER tasks in any language or domain. One can also use external resources such as Wikipedia to construct a larger gazetteer (Fetahu et al., 2021). We will explore a larger gazetteer in future work because it is not the focus in this paper.

Furthermore, although the generated gazetteer is pretty accurate, a downside is that when we integrate such a high-quality gazetteer in the model, the model tends to put too much trust in the gazetteer. In the other way round, it hurts the model's generalization ability. Therefore, we intentionally add some noise to the gazetteer. Specifically, with probability λ, we choose one of the following three strategies to add noise: (1) randomly select a span that is not labeled as named entity, and then add it to the gazetteer with a random entity type; (2) for a labeled named entity span, add it to the gazetteer with a randomly assigned wrong entity type; (3) skip over adding a labeled named entity span to the gazetteer. In practice, we set λ to a small value, so that it gives the gazetteer strong control in making final predictions, while the model's generalization ability is still reserved to some degree.

Note that during training, the gazetteer is constructed using training and development data. When we want to fix errors in test data, the gazetteer is updated using test data.

3.2 Model Architecture

TyBERT is built on standard BERT with two modifications: (1) given a sentence, the input word sequence is converted to a word-type pair sequence that will be the input for TyBERT; (2) a type adapter for integrating type information in BERT is attached between Transformer layers.

Word-Type Pair Sequence. Given a gazetteer G and a sentence with a sequence of words $s_w = \{w_1, w_2, ..., w_n\}$, we match the word sequence with G to find out all potential named entities inside the sentence. So we have a word-type pair sequence $s_{wt} = \{wt_1, wt_2, ..., wt_n\}$. When the word w_i is not a part of any potential named entity, wt_i is w_i. Otherwise, wt_i is (w_i, t_i), where t_i is all matched entities' types with *B-* or *I-* as prefix to indicate whether it begins or inside a named entity.

Taking the sentence "London Bridge is famous" for example, the word "London" is a part of two potential named entities, i.e., (1) "London" with type *art-music* and *location-gpe*, and (2) "London Bridge" with type *building*. Therefore, t_i for the word "London" is $\{[B\text{-}art\text{-}music, B\text{-}location\text{-}gpe], [B-building]\}$.

Formally, we have t_i=$\{Type(x_{ij})\}$. x_{ij} is the j^{th} potential named entity that contains the word w_i. $Type(x)$=$[et_1, et_2, ..et_k]$ represents all possible entity types of named entity x based on G, and et_i is one of the possible labels, such as $B\text{-}art\text{-}music$, etc.

Type Adapter. Our Type Adapter (TA) is shown

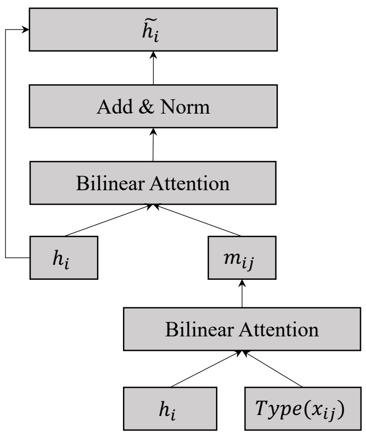

Figure 2: Structure of Type Adapter (TA).

Lang.	Dataset	Type	Train	Dev	Test
English	OntoNotes v5.0 (18 types)	Sent	75.8k	9.4k	9.6k
		Token	1299k	163k	169k
		Entity	81k	11k	11k
	Few-NERD (66 types)	Sent	131k	18.8k	37.6k
		Token	3227k	463k	921k
		Entity	340k	48.7k	96.9k
Chinese	OntoNotes v4.0 (8 types)	Sent	15.7k	4.3k	4.3k
		Token	491k	200k	208k
		Entity	13.3k	6.9k	7.6k
	Weibo (4 types)	Sent	13.5k	0.27k	0.27k
		Token	7.4k	14.6k	14.9k
		Entity	1.8k	0.38k	0.41k

Table 1: The Statistics of four corpus.

in Figure 2, which is inspired by Lexicon Adapter proposed in Liu et al. (2021). Specifically, as discussed above, t_i has a two-level structure, so we propose a two-level attention mechanism.

Firstly, at position i, we compute the cross attention between the hidden state h_i with the embeddings of possible entity types $Type(x_{ij})$ for a potential named entity x_{ij} to obtain m_{ij}. Then we compute another cross attention between the hidden state h_i and m_{ij}, and finally obtain the new hidden state $\widetilde{h_i}$.

Compared with BERT, the only extra parameters of TyBERT are the embeddings of entity type et_k and related weights in two cross attentions, which can be fully learned in training time. Thus, when updating the gazetteer in test time, we don't have to update any parameters in TyBERT. Following Liu et al. (2021), we only insert a TA after the first transformer layer.

Method	English						Chinese					
	OntoNotes V5.0			Few-NERD			OntoNotes V4.0			Weibo		
	P	R	F-1	P	R	F-1	P	R	F-1	P	R	F-1
BERT	89.32	86.94	88.11	69.65	67.19	68.4	83.45	81.39	82.41	72.16	70.09	71.11
BERT+Intersect	97.67	82.86	89.62	95.8	56.14	70.8	92.1	59.66	72.42	91.38	58.37	71.24
BERT+Union	78.12	89.87	87.28	54.69	94.91	69.39	34.99	94.06	51.07	51.01	90.19	65.16
TyBERT(λ=0.05)	**94.67**	**94.82**	**94.74**	**86.86**	**87.76**	**87.31**	**86.93**	**85.03**	**85.97**	**73.79**	**80.86**	**77.16**

Table 2: Experiment results in four corpus.

4 Evaluation

4.1 Experimental Setup

Datasets. For evaluation, we employ four datasets, two in English and two in Chinese. For English, we employ the commonly used OntoNotes 5.0 corpus (Pradhan et al., 2013) and also the challenging Few-NERD corpus (Ding et al., 2021) with 66 fine-grained types. For Chinese, we employ OntoNotes 4.0 corpus (Weischedel et al., 2011) and Weibo corpus (Peng and Dredze, 2015, 2016) from social media domain. The detailed statistics of four corpora are shown in Table 1.

Evaluation measures. Following previous NER works, Standard F1-score (F1), Precision (P) and Recall (R) are used as evaluation metrics.

Hyperparameter tuning. We tune training related hyper-parameters in the development set and reported results in the test set. The tuned hyperparameter values are shown in Appendix A.

Implementation details. The implementation details are explained in Appendix B.

4.2 Results

Baseline systems. To compare with our proposed method, we use BERT (Devlin et al., 2018) as a baseline. Because standard BERT cannot correct errors without model re-training, we further designed two additional baseline systems. These two baseline systems ensemble BERT and a rule-based method using a gazetteer as follows. We construct the gazetteer using all of training, development and test data. Then the gazetteer is used to match the sentences in test data to identify named entities. When a span has multiple entity types, we randomly assign a type. Depending on whether we intersect or union the output of BERT and the rule-based method, we name two baseline systems BERT+Intersect and BERT+Union respectively.

Discussions. Results of BERT, two extra baseline systems and our proposed TyBERT are shown in Table 2. As we can see, compared with BERT, BERT+Intersect improves BERT by a small margin in three corpora, and BERT+Union only improves BERT slightly in Few-NERD corpus. In contrast, with λ=0.05 (tuned on development set), our proposed method TyBERT improves BERT by a large margin, i.e., 6.63% and 18.91% in two English corpus, and 3.56% and 6.05% in two Chinese corpus. We notice that the improvement in Chinese corpus is smaller than in English corpus. The reason is that there are much more named entities with multiple types in Chinese corpus, e.g., the confusion of *location* and *gpe* have caused many errors. In future work, we plan to consider named entity's context to fix errors. We have separately analyzed the gains brought by our solution on the ontonotes v4.0 datasets are shown in Appendix D.

λ	before updating gazetteer using test data			after updating gazetteer using test data		
	P	R	F1	P	R	F1
0	62.65	59.21	60.88	79.62	84.14	81.82
0.05	85.09	72.73	78.43	86.93	85.03	85.97
0.1	82.01	78.9	80.4	83.4	86.27	84.81
0.2	83.41	76.79	79.97	85.73	82.99	84.34

Table 3: Results of TyBERT with different λ.

4.3 Impact of gazetteer noise

We further conduct experiments to study the impact of gazetteer noise in Chinese OntoNotes corpus. Results are shown in Table 3. For each λ, we show the results of TyBERT before and after updating the gazetteer using test data. A few observations are obtained. When λ is set to 0, the model before updating gazetteer loses generalization ability, and hence performs poorly. After λ is set to a non-zero value, the model before updating gazetteer improves a lot, and many errors are fixed after updating the gazetteer using test data.

5 Conclusions

We introduced a new task of correcting NER errors without re-training models. We propose TyBERT which extended standard BERT model with an adapter layer to incorporate span's type information stored in a gazetteer. We further introduce a noise rate parameter to balance the strength of the gazetteer and model's generalization ability. Extensive results justified the effectiveness of the proposed method. We hope our work will inspire future studies towards NER error correction without model re-training.

Limitations

A limitation of the proposed method is that our gazetteer is constructed only by dataset annotations. And it affects the gazetteer coverage in unseen cases. Following previous work, such as Lin et al. (2019) and Fetahu et al. (2022), we will construct a larger gazetteer using external resources such as Wikipedia or knowledge bases. As mentioned in Section 3, we will leave this for future work.

Another limitation is that the gazetteer contains many spans that are associated with multiple entity types. Taking the running examples in Section 3.1 for example, the span "London" has type *location-gpe* in most cases, while it is sometimes labeled as type *art-music*. However, in our current design, given a named entity, there is no way to explicitly distinguish between different types. In future work, we will consider the context of named entity when fixing errors.

Ethics Statement

We declare that all authors of this work comply with the ACL Ethics Policy as published in `https://www.aclweb.org/portal/content/acl-code-ethics`.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Hai-Tao Zheng, and Zhiyuan Liu. 2021. Few-nerd: A few-shot named entity recognition dataset. *arXiv preprint arXiv:2105.07464*.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer enhanced named entity recognition for code-mixed web queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2022. Dynamic gazetteer integration in multilingual models for cross-lingual and cross-domain named entity recognition. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2777–2790.

Tao Gui, Yicheng Zou, Qi Zhang, Minlong Peng, Jinlan Fu, Zhongyu Wei, and Xuan-Jing Huang. 2019. A lexicon-based graph neural network for chinese ner. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1040–1050.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

Xiaonan Li, Hang Yan, Xipeng Qiu, and Xuanjing Huang. 2020. Flat: Chinese ner using flat-lattice transformer. *arXiv preprint arXiv:2004.11795*.

Hongyu Lin, Yaojie Lu, Xianpei Han, Le Sun, Bin Dong, and Shanshan Jiang. 2019. Gazetteer-enhanced attentive neural networks for named entity recognition. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6232–6237.

Tianyu Liu, Jin-Ge Yao, and Chin-Yew Lin. 2019. Towards improving neural named entity recognition with gazetteers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5301–5307.

Wei Liu, Xiyan Fu, Yue Zhang, and Wenming Xiao. 2021. Lexicon enhanced chinese sequence labeling using bert adapter. *arXiv preprint arXiv:2105.07148*.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gemnet: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Nanyun Peng and Mark Dredze. 2015. Named entity recognition for chinese social media with jointly trained embeddings. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 548–554.

Nanyun Peng and Mark Dredze. 2016. Improving named entity recognition for chinese social media with word segmentation representation learning. *arXiv preprint arXiv:1603.00786*.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using ontonotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152.

Jana Straková, Milan Straka, and Jan Hajič. 2019. Neural architectures for nested ner through linearization. *arXiv preprint arXiv:1908.06926*.

Dianbo Sui, Yubo Chen, Kang Liu, Jun Zhao, and Shengping Liu. 2019. Leverage lexical knowledge for chinese named entity recognition via collaborative graph network. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3830–3840.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021. Improving named entity recognition by external context retrieving and cooperative learning. *arXiv preprint arXiv:2105.03654*.

Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, Martha Palmer, Nianwen Xue, Mitchell Marcus, Ann Taylor, Craig Greenberg, Eduard Hovy, Robert Belvin, et al. 2011. Ontonotes release 4.0. *LDC2011T03, Philadelphia, Penn.: Linguistic Data Consortium*.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. Byt5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

Vikas Yadav and Steven Bethard. 2019. A survey on recent advances in named entity recognition from deep learning models. *arXiv preprint arXiv:1910.11470*.

Yue Zhang and Jie Yang. 2018. Chinese ner using lattice lstm. *arXiv preprint arXiv:1805.02023*.

A Hyperparameter

The tuned hyperparemeters are shown in Table 4.

B Implementation Details

We implemented the models using PyTorch. All models are initialized from BERT-base English or Chinese checkpoints(Devlin et al., 2018) which have about 110M parameters. Each experiment is trained on a single V100 GPU for about 1 to 4 hours depending on the corpus size.

	English		Chinese	
	OntoNotes	FewNERD	OntoNotes	Weibo
LR	1e-4	2e-5	5e-5	3e-4
Weight Decay	0.01	0.01	0.01	0.01
#Epoch	10	5	20	8
Batch Size	16	32	64	32

Table 4: The hyperparameters used in four corpus.

Dataset types	GPE	ORG	PER	LOC
Prediction	3501	1747	1933	313
Incorrect entities	573	417	124	126
In training set	397	181	52	70
Not in training set	176	236	72	56
Corrected by TyBert	35	4	10	9
In training set	8	-	-	1
Not in training set	27	4	10	8
New incorrect entities	15	1	2	7

Table 5: The distribution of error labels corrected by the model

Dataset types	GPE	ORG	PER	LOC
Golden label	3452	1877	1864	491
No recall	485	521	39	276
In training set	344	148	-	70
Not in training set	141	373	39	206
New recalled by Tybert	197	305	22	71
In training set	121	39	-	20
Not in training set	76	266	22	51
New incorrect entities	16	4	-	1

Table 6: The distribution of newly recalled labels by the model

C Corpus License

Few-NERD corpus is under the CC 821 BY-SA 4.0 license, Weibo corpus is under CC BY-SA 3.0 license and OntoNotes corpus are used under LDC license. These corpus does not contain any personally identifiable information or offensive content.

D Correct and Recall details in ontonotes datasets

Comparing BERT and TyBERT, mainly includes the following aspects: 1. number of errors for each type of entity 2. type of errors for each type of entity (substitution or deletion) 3. number of corrections for unseen data in the training 4. number of corrections for seen data in the training More details can be found in Table 5 and 6.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Limitations

☑ A2. Did you discuss any potential risks of your work?
Limitations

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract and Section 1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Section 4. We use the pre-trained language models including BERT-English and BERT-Chinese. In addition, we used four corpus datasets in our experiments.

☑ B1. Did you cite the creators of artifacts you used?
Section 2 and 4.

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Appendix C.

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Section 4.

☑ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Section 4.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Section 4.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Section 4.

C ☑ Did you run computational experiments?

Section 4.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 4.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 4.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 4.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Appendix B.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?
Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.