A Simple but Effective Pluggable Entity Lookup Table for Pre-trained Language Models

Anonymous ACL submission

Abstract

Pre-trained language models (PLMs) cannot well recall rich factual knowledge of entities exhibited in large-scale corpora, especially those rare entities. In this paper, we propose to build a simple but effective **P**luggable **E**ntity **L**ookup **T**able (PELT) on demand by aggregating the entity's output representations of multiple occurrences in the corpora. PELT can be compatibly plugged as inputs to infuse supplemental entity knowledge into PLMs. Compared to previous knowledge-enhanced PLMs, PELT only requires 2‰∼5% pre-computation with capability of acquiring knowledge from out-of-domain corpora for domain adaptation scenario. The experiments on knowledge-related tasks demonstrate that our method, PELT, can flexibly and effectively transfer entity knowledge from related corpora into PLMs. We will make all the data and codes publicly available to facilitate future research.

1 Introduction

Recent advance in pre-trained language models (PLMs) has achieved promising improvements in various downstream tasks (Devlin et al., 2019; Liu et al., 2019). Some latest works reveal that PLMs can automatically acquire knowledge from large-scale corpora via self-supervised pre-training and then encode the learned knowledge into their model parameters (Tenney et al., 2019; Petroni et al., 2019; Roberts et al., 2020). However, due to the limited capacity of vocabulary, existing PLMs face the challenge of recalling the factual knowledge from their parameters, especially for those rare entities (Gao et al., 2019a; Wang et al., 2021a).

To improve PLMs' capability of entity understanding, a straightforward solution is to exploit an external entity embedding acquired from the knowledge graph (KG) (Zhang et al., 2019; Liu et al., 2020; Wang et al., 2020), the entity description (Peters et al., 2019), or the corpora (Pörner et al., 2020). In order to make use of the external knowledge, these models usually learn to align the external entity embedding (Bordes et al., 2013; Yamada et al., 2016) to the their original word embedding. However, previous works ignore to explore entity embedding from the PLM itself, which makes their learned embedding mapping is not available in the domain-adaptation. Other recent works attempt to infuse knowledge into PLMs' parameters by extra pre-training, such as learning to build an additional entity vocabulary from the corpora (Yamada et al., 2020; Févry et al., 2020), or adopting entity-related pre-training tasks to intensify the entity representation (Xiong et al., 2020; Sun et al., 2020; Wang et al., 2021b). However, their huge pre-computation increases the cost of extending or updating the customized vocabulary for various downstream tasks.

Model	#Ent	Pre-Comp.	D-Adapt
Zhang et al. (2019)	5.0M	∼160h	No
Wang et al. (2021b)	4.6M	∼3,400h	No
Yamada et al. (2020)	0.5M	∼3,800h	No
PELT (our model)	4.6M	7h	Yes

Table 1: Comparison of recent knowledge-enhanced PLMs. We report the pre-computation of `BASE` models on Wikipedia entities on a V100 GPU. Pre-Comp.: Pre-computation; D-Adapt: Domain Adaptation.

In this paper, we introduce a simple but effective **P**luggable **E**ntity **L**ookup **T**able (PELT) to infuse knowledge into PLMs. To be specific, we first revisit the connection between PLMs' input features and output representations for masked language modeling. Based on this, given a new corpus, we aggregate the output representations of masked tokens from the entity's occurrences, to recover an elaborate entity embedding from a well-trained PLM. Benefiting from the compatibility and flexibility of the constructed embedding, we can directly insert them into the corresponding positions of the input sequence to provide supplemental en-

tity knowledge. As shown in Table 1, our method merely consumes 2‰∼5% pre-computation compared with previous works, and it also supports the vocabulary from different domains simultaneously.

We conduct experiments on two knowledge-related tasks, including knowledge probe and relation classification, across two domains (Wikipedia and biomedical publication). Experimental results show that PLMs with PELT can consistently and significantly outperform the corresponding vanilla models. In addition, the entity embedding obtained from multiple domains are compatible with the original word embedding and can be applied and transferred swiftly.

2 Methodology

In this section, we first revisit the masked language modeling pre-training objective. After that, we introduce the pluggable entity lookup table and explain how to apply it to incorporate knowledge into PLMs.

2.1 Revisit Masked Language Modeling

PLMs conduct self-supervised pre-training tasks, such as masked language modeling (MLM) (Devlin et al., 2019), to learn the semantic and syntactic knowledge from the large-scale unlabeled corpora (Rogers et al., 2020). MLM can be regarded as a kind of cloze task, which requires the model to predict the missing tokens based on its contextual representation. Formally, given a sequence of tokens $X = (x_1, x_2, \ldots, x_n)$, with x_i substituted by `[MASK]`, PLMs, such as BERT, first take tokens' word embedding and position embedding as input and obtain the contextual representation:

$$\boldsymbol{H} = \text{Enc}(\text{LayerNorm}(\mathbf{E}(X) + \boldsymbol{P})), \quad (1)$$

where $\text{Enc}(\cdot)$ denotes a deep bidirectional Transformer encoder, $\text{LayerNorm}(\cdot)$ denotes layer normalization (Ba et al., 2016), $\mathbf{E} \in \mathbb{R}^{|V| \times D}$ is the word embedding matrix, V is the word vocabulary, P is the absolute position embedding and $\boldsymbol{H} = (\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_n)$ is the contextual representation. After that, BERT applies a feed-forward layer (FFN) and layer normalization on the contextual representation to compute the output representation of x_i:

$$\boldsymbol{r}_{x_i} = \text{LayerNorm}(\text{FFN}(\boldsymbol{h}_i)). \quad (2)$$

Since the weights in the softmax layer and word embeddings are tied in BERT, the model calculate the product of $\boldsymbol{r}_{x_i}$ and the input word embedding matrix to further compute x_i's cross-entropy loss among all the words:

$$\begin{aligned} \mathcal{L} &= -\sum \log \Pr(x_i | \boldsymbol{r}_{x_i}) \\ &= -\sum \log \frac{\exp(\mathbf{E}(x_i)^T \boldsymbol{r}_{x_i})}{\sum_{w_j \in V} \exp(\mathbf{E}(w_j)^T \boldsymbol{r}_{x_i})}. \end{aligned} \quad (3)$$

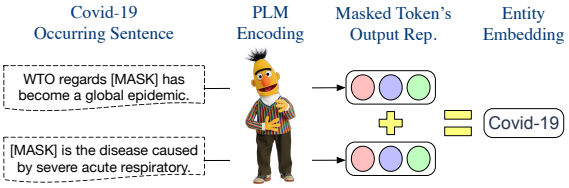

Figure 1: An illustration of the our PELT.

2.2 Construct Pluggable Entity Embedding

Due to the training efficiency, the vocabulary sizes in existing PLMs typically range from 30K to 60K subword units, and thus PLMs have to disperse the information of massive entities into their subword embeddings. Through revisiting the MLM loss in Eq. 3, we could intuitively observe that the word embedding and the output representation of BERT are located in the same vector space. Hence, we are able to recover the entity embedding from BERT's output representations to infuse their contextualized knowledge to the model.

To be specific, given a general or domain-specific corpus, we design to build the lookup table for entities that occurs in the downstream tasks on demand. For an entity e, such as a Wikidata entity or a proper noun entity, we construct its embedding $\mathbf{E}(e)$ as follows:

Direction A feasible method to add entity e to the vocabulary of PLM is to optimize its embedding $\mathbf{E}(e)$ for the MLM loss with other parameters frozen. We collect the sentences S_e that contain entity e and substitute it with `[MASK]`. The total influence of $\mathbf{E}(e)$ to the MLM loss in S_e can be formulated as:

$$\begin{aligned} \mathcal{L}(e) &= -\sum_{x_i \in S_e} \log \Pr(e | \boldsymbol{r}_{x_i}) \\ &= \sum_{x_i \in S_e} \log Z_{x_i} - \mathbf{E}(e)^T \sum_{x_i \in S_e} \boldsymbol{r}_{x_i}, \end{aligned} \quad (4)$$

where $Z_{x_i} = \sum_{w_j \in V \cup \{e\}} \exp(\mathbf{E}(w_j)^T \boldsymbol{r}_{x_i})$, x_i is the replaced masked token for entity e and $\boldsymbol{r}_{x_i}$ is the PLM's output representation of x_i.

Compared with the total impact of the entire vocabulary on Z_{x_i}, $\mathbf{E}(e)$ has a much smaller impact. If we ignore the minor effect of $\mathbf{E}(e)$ on Z_{x_i}, the optimal solution of $\mathbf{E}(e)$ for $\mathcal{L}(e)$ is proportional to $\sum_{x_i \in S_e} \boldsymbol{r}_{x_i}$. Hence, we set $\mathbf{E}(e)$ as:

$$\mathbf{E}(e) = C \cdot \sum_{x_i \in S_e} \boldsymbol{r}_{x_i}, \tag{5}$$

where C denotes the scaling factor.

Practically, $\mathbf{E}(e)$ also serves as the negative log-likelihood of other words' MLM loss (Kong et al., 2020). However, Gao et al. (2019a) indicates that the gradient from such negative log-likelihood will push all words to a uniformly negative direction, which weakens the quality of rare words' representation. Here, we ignore this negative term and obtain the informative entity embedding from Eq. 5.

Norm We define $\boldsymbol{p}(e)$ as the position embedding for entity e. Since the layer normalization in Eq. 1 makes the norm $|\mathbf{E}(e) + \boldsymbol{p}(e)|$ to $D^{\frac{1}{2}}$, we find that the norm $|\mathbf{E}(e)|$ has little effect on the input feature of the encoder in use. Therefore, we set the norm of all the entity embeddings as a constant L. Then, we evaluate the model with different L on the unsupervised knowledge probe task and choose the best L for those fine-tuning tasks.

2.3 Infuse Entity Knowledge into PLMs

Since the entity embedding we obtained and the original word embedding are both obtained from the masked language modeling objective, the entity can be regarded as a special input token. To infuse entity knowledge into PLMs, we apply a pair of bracket to enclose the constructed entity embedding and then insert it after the original entity's subwords. For example, the original input,

Steve Job works for `[MASK]`.

becomes

*Steve Job (**Steven_Job**) works for* `[MASK]`.

Here, the entity ***Steven_Job*** adopts our constructed entity embedding and other words use their original embedding. We simply convey the modified input to the PLM for encoding without any additional structures or parameters.

A note on entity links In previous section, we hypothesize that we know the entity linking annotations for the involved string name. In practice, we can obtain the gold entity links provided by some datasets like FewRel 1.0. For the datasets where the linking annotations are not available, we employ a heuristic string match for entity linking[1].

3 Experiment

3.1 Implementation Details

We choose RoBERTa$_{\texttt{Base}}$ (Liu et al., 2019), a well-optimized PLM, as our baseline model and we equip it with our constructed entity embedding to obtain the PELT model. We adopt Wikipedia and biomedical S2ORC (Lo et al., 2020) as the domain-specific corpora and split them into sentences with NLTK (Xue, 2011). For Wikipedia, we adopt a heuristic entity linking strategy with the help of hyperlink annotations. For the used FewRel 1.0 and Wiki80 datasets, we directly use the annotated linking information. For other datasets, we link the given entity name through a simple string match. For each necessary entity, we first extract up to 256 sentences containing the entity from the corpora. After that, we construct the entity embedding according to Section 2.2. In the fine-tuning process, we freeze the constructed embeddings as an lookup table. We run all the fine-tuning experiments with 5 different seeds and report the average score.

3.2 Baselines

We select two of the most representative entity-aware baselines, which adopt an external entity embedding or an entity-related pre-training task: (1) **ERNIE** (Zhang et al., 2019) involves the entity embedding learned from Wikidata relation (Bordes et al., 2013). We adopt the RoBERTa version of ERNIE provided by Wang et al. (2021b); (2) **KEPLER** (Wang et al., 2021b) encodes textual entity description into entity embedding and learns fact triples and language modeling simultaneously.

3.3 Relation Classification

Relation Classification (RC) aims to predict the relationship between two entities in a given text. We evaluate the models on two scenarios, the few-shot setting and the full-data setting.

The few-shot setting focuses on long-tail relations without sufficient training instances. We evaluate models on FewRel 1.0 (Han et al., 2018) and FewRel 2.0 (Gao et al., 2019b). FewRel 1.0 contains instances with Wikidata facts and FewRel 2.0 involves a biomedical-domain test set to examine the ability of domain adaptation. In the

[1]Details are shown in the Appendix.

Model	Extern KG	FewRel 1.0 5-1	FewRel 1.0 5-5	FewRel 1.0 10-1	FewRel 1.0 10-5	FewRel 2.0 5-1	FewRel 2.0 5-5	FewRel 2.0 10-1	FewRel 2.0 10-5
ERNIE†	✓	$92.7_{\pm0.2}$	$97.9_{\pm0.0}$	$87.7_{\pm0.4}$	$96.1_{\pm0.1}$	$66.4_{\pm1.6}$	$88.2_{\pm0.5}$	$51.2_{\pm0.7}$	$80.1_{\pm1.0}$
KEPLER	✓	$90.8_{\pm0.1}$	$96.9_{\pm0.1}$	$85.1_{\pm0.1}$	$94.2_{\pm0.1}$	$74.0_{\pm1.0}$	$89.2_{\pm0.2}$	$61.7_{\pm0.1}$	$82.1_{\pm0.1}$
RoBERTa	-	$90.4_{\pm0.3}$	$96.2_{\pm0.0}$	$84.2_{\pm0.5}$	$93.9_{\pm0.1}$	$71.2_{\pm2.1}$	$89.4_{\pm0.2}$	$53.3_{\pm0.8}$	$83.1_{\pm0.4}$
PELT	-	**92.7**$_{\pm0.3}$	**97.5**$_{\pm0.0}$	**87.5**$_{\pm0.3}$	**95.4**$_{\pm0.1}$	**75.0**$_{\pm1.3}$	**92.1**$_{\pm0.2}$	**60.4**$_{\pm1.1}$	**85.6**$_{\pm0.2}$

Table 2: The accuracy on the FewRel dataset. N-K indicates the N-way K-shot configuration. Both of FewRel 1.0 and FewRel 2.0 are trained on the Wikipedia domain, and FewRel 2.0 is tested on the biomedical domain. ERNIE† has seen facts in the FewRel 1.0 test set during pre-training. We report standard deviations as subscripts.

Model	1%	10%	100%
ERNIE	$66.4_{\pm0.4}$	$87.7_{\pm0.2}$	$93.4_{\pm0.1}$
KEPLER	$62.3_{\pm1.0}$	$85.4_{\pm0.2}$	$91.7_{\pm0.1}$
RoBERTa	$59.8_{\pm1.7}$	$85.7_{\pm0.2}$	$91.7_{\pm0.1}$
PELT	**65.6**$_{\pm1.0}$	**88.3**$_{\pm0.3}$	**93.4**$_{\pm0.1}$

Table 3: The accuracy on the test set of Wiki80. 1%/ 10% indicate using 1%/ 10% supervised training data respectively.

Model	LAMA G-RE	LAMA T-REx	LAMA-UHN G-RE	LAMA-UHN T-REx
ERNIE	10.0	24.9	5.9	19.4
KEPLER	5.5	23.4	2.5	15.4
RoBERTa	5.4	24.7	2.2	17.0
PELT	**6.4**	**27.5**	**2.8**	**19.3**

Table 4: Mean P@1 on the knowledge probe benchmark. G-RE: Google-RE.

Model	[0,10)	[10,50)	[50,100)	[100,+)
RoBERTa	18.1	21.1	25.8	26.1
PELT	**21.9**	**24.8**	**29.0**	**28.7**

Table 5: Mean P@1 on T-Rex with respect to the subject entity's frequency in Wikipedia.

N-way K-shot setting, models are required to categorize the query as one of the existing N relations, each of which contains K supporting samples. We choose the state-of-the-art few-shot framework Proto (Snell et al., 2017) with different PLM encoders for evaluation. For the full-data setting, we evaluate models on the Wiki80, which contains 80 relation types from Wikidata. We also add 1% and 10% settings, meaning using only 1% / 10% data of the training sets.

As shown in Table 2 and Table 3, on FewRel 1.0 and Wiki80 in Wikipedia domain, RoBERTa with PELT beats the RoBERTa model by a large margin (e.g. +3.3% on 10way-1shot), and it even achieves comparable performance with ERNIE, which has access to the knowledge graph. Our model also gains huge improvements on FewRel 2.0 in the biomedical domain (e.g. +7.1% on 10way-1shot), while the entity-aware baselines have little advance in most settings. Compared with most existing entity-aware PLMs which merely obtain domain-specific knowledge in the pre-training phase, our proposed pluggable entity lookup table can dynamically update the models' knowledge from the out-of-domain corpus on demand.

3.4 Knowledge Probe

We conduct experiments on a widely-used knowledge probe dataset, LAMA (Petroni et al., 2019). It applies cloze-style questions to examine PLMs' ability on recalling facts from their parameters. For example, given a question template *Paris is the capital of* `[MASK]`, PLMs are required to predict the masked token properly. In this paper, we not only use Gooogle-RE and T-REx (ElSahar et al., 2018) which focus on factual knowledge, but also evaluate models on LAMA-UHN (Pörner et al., 2020) which filters out the easy questionable templates.

As shown in Table 4, without any pre-training, the PELT model can directly absorb the entity knowledge from the extended input sequence to recall more factual knowledge, which demonstrates that the entity embeddings we constructed are compatible with original word embeddings.

Effect of Entity Frequency Table 5 shows the P@1 results with respect to the entity frequency. While RoBERTa performs worse on rare entities than frequent entities, PELT brings a substantial improvement on rare entities, i.e., near 3.8 mean P@1 gains on entities that occur less than 50 times.

4 Conclusion and Future work

In this paper, we propose PELT, a flexible entity lookup table, to incorporate up-to-date knowledge into PLMs. By constructing entity embeddings on demand, PLMs with PELT can recall rich factual knowledge to help downstream tasks.

References

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *CoRR*, abs/1607.06450.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Hady ElSahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon S. Hare, Frédérique Laforest, and Elena Simperl. 2018. T-rex: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*. European Language Resources Association (ELRA).

Thibault Févry, Livio Baldini Soares, Nicholas FitzGerald, Eunsol Choi, and Tom Kwiatkowski. 2020. Entities as experts: Sparse memory access with entity supervision. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4937–4951, Online. Association for Computational Linguistics.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2019a. Representation degeneration problem in training natural language generation models. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Tianyu Gao, Xu Han, Hao Zhu, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2019b. FewRel 2.0: Towards more challenging few-shot relation classification. 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 6250–6255, Hong Kong, China. Association for Computational Linguistics.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. FewRel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4803–4809, Brussels, Belgium. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Lingpeng Kong, Cyprien de Masson d'Autume, Lei Yu, Wang Ling, Zihang Dai, and Dani Yogatama. 2020. A mutual information maximization perspective of language representation learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-BERT: enabling language representation with knowledge graph. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 2901–2908. AAAI Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. 2020. S2ORC: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983, Online. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. 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 43–54, Hong Kong, China. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? 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 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Nina Pörner, Ulli Waltinger, and Hinrich Schütze. 2020. E-BERT: efficient-yet-effective entity embeddings for BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 803–818. Association for Computational Linguistics.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5418–5426. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how BERT works. *Trans. Assoc. Comput. Linguistics*, 8:842–866.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4077–4087.

Tianxiang Sun, Yunfan Shao, Xipeng Qiu, Qipeng Guo, Yaru Hu, Xuanjing Huang, and Zheng Zhang. 2020. CoLAKE: Contextualized language and knowledge embedding. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3660–3670, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Cunxiang Wang, Pai Liu, and Yue Zhang. 2021a. Can generative pre-trained language models serve as knowledge bases for closed-book qa? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3241–3251. Association for Computational Linguistics.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2020. K-adapter: Infusing knowledge into pre-trained models with adapters. *CoRR*, abs/2002.01808.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021b. KEPLER: A unified model for knowledge embedding and pre-trained language representation. *Trans. Assoc. Comput. Linguistics*, 9:176–194.

Wenhan Xiong, Jingfei Du, William Yang Wang, and Veselin Stoyanov. 2020. Pretrained encyclopedia: Weakly supervised knowledge-pretrained language model. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Nianwen Xue. 2011. Steven bird, evan klein and edward loper. *Natural Language Processing with Python*. o'reilly media, inc 2009. ISBN: 978-0-596-51649-9. *Nat. Lang. Eng.*, 17(3):419–424.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6442–6454, Online. Association for Computational Linguistics.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint learning of the embedding of words and entities for named entity disambiguation. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pages 250–259. ACL.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy. Association for Computational Linguistics.

A Heuristic String Match for Entity Linking

For the Wikipedia, we first create a mapping from the anchor texts with hyperlinks to their referent Wikipedia pages. After that, We employ a heuristic string matching to link other potential entities to their pages.

For preparation, we collect the aliases of the entity from the redirect page of Wikipedia and the relation between entities from the hyperlink. Then, we apply spaCy [2] to recognize the entity name in the text. An entity name in the text may refer to multiple entities of the same alias. We utilize the relation of the linked entity page to maintain an available entity page set for entity disambiguation .

Details of the heuristic string matching are shown in Algorithm 1, we match the entity name to surrounding entity page of the current page as close

[2] https://spacy.io/

Algorithm 1 Heuristic string match for entity disambiguation

$S \Leftarrow \{$ the linked entity page in anchor text$\}$
$E \Leftarrow \{$ potential entity name in text$\}$
repeat
  $S' \Leftarrow \{$ the neighbor entity pages that have hyperlink or Wikidata relation with pages in $S\}$
  $E' \Leftarrow \{e | e \in E$ and e can be uniquely linked to entity page in S' by string matching $\}$
  $E \Leftarrow E - E'$
  $S \Leftarrow E'$
until $S = \phi$

as possible. e will release all the source code and models with the pre-processed Wikipedia dataset.

For other datases, we adopt a simple string match for entity linking.

B Training Configuration

We train all the models with Adam optimizer (Kingma and Ba, 2015), 10% warming up steps and maximum 128 input tokens. Detailed training hyper-parameters are shown in Table 6.

For Wiki80, KBP37 and ChemProt, we run experiments with 5 different seeds $(42, 43, 44, 45, 46)$ and report the average scores and the standard deviations. And we run the 1% and 10% experiments with 5-25 times epochs as that of the 100% experiment.

For FewRel, we search the batch size among $[4, 8, 32]$ and search the training step in $[1500, 2000, 2500]$. We evaluate models every 250 on validation and save the model with best performance for testing. With our hyper-parameter tuning, the results of baselines in FewRel significantly outperforms that reported by KEPLER (Wang et al., 2021b).

Dataset	Epoch	Train Step	BSZ	LR
MLM	-	2,000	8192	1e-4
Wiki80	5	-	32	3e-5
KBP37	5	-	32	3e-5
ChemProt	5	-	32	3e-5
FewRel 1.0	-	2500	32	2e-5
FewRel 2.0	-	1500	32	2e-5

Table 6: Training Hyper-parameters. BSZ: Batch size; LR: Learning rate.