Revisiting the Knowledge Injection Frameworks

Anonymous ACL submission

Abstract

In recent years, pre-trained language models (PLMs) have achieved some eye-catching results on many natural language processing (NLP) tasks. Upon that, a plethora of knowledge-injected PLMs — assisted by external knowledge graphs — have been proposed to further enhance or adapt original PLMs on specific downstream tasks. Among these exciting results, we may identify some (potentially) strange odd phenomena such as the imbalance across downstream tasks, little correlation between the injected knowledge and the chosen tasks, the mismatch between knowledge and sentences, etc. These phenomena concern us about the effect of the specific injected knowledge on the model while doing the downstream task. In this work, we intend to comprehensively revisit a series of well-known knowledge-injected frameworks on most common benchmarks, by conducting extensive ablation and control experiments that were previously mostly omitted. Coupled with dense analysis by tracking down the transfer path of the knowledge vectors, we may draw a frustrating conclusion that the current knowledge-injected frameworks may have minimal effect in leveraging the injected knowledge. We further cast a hypothesis to interpret the performance enhancement of the knowledge-injected PLMs from a data augmentation perspective.

1 Introduction

Large-scale pre-trained language models — iconically BERT(Devlin et al., 2019), GPT-3(Brown et al., 2020), etc. — manage to encode distributed representation through training on massive unlabeled text corpora. These methods have continuously extended the frontier of numerous NLP tasks and established new architectural standards in the field. One notable incremental scheme upon the PLMs is to inject external knowledge, which leads to the name of "knowledge-injected PLMs". The knowledge injection is mostly made possible by vectorial representation from external knowledge graphs and conducted in either pre-training or fine-tuning stages. For instance, BioBERT(Lee et al., 2019) and SCIBERT(Beltagy et al., 2019), pre-trained on the domain-specific corpora, display some decent improvement over the vanilla BERT on the biomedical and scientific tasks. Similar examples also emerged like LUKE(Yamada et al., 2020), ERNIE(Zhang et al., 2019) and KnowBert(Peters et al., 2019) on tasks such as question answering, entity typing, entity disambiguation, etc. Further, for the low-resource domain such as medical NLP (Lee et al., 2019; Yuan et al., 2021), to be able to engage a deep neural network with knowledge is conceptually indispensable.

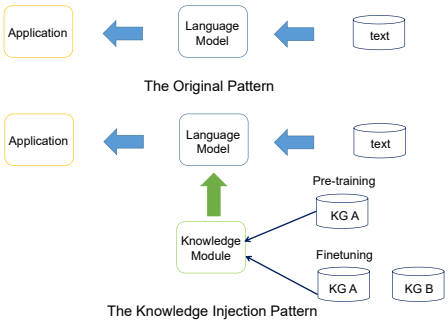

Figure 1: Knowledge Injection Diagram.

Despite the promise, we find some slight clues implying the odd pattern from the current knowledge-injected frameworks. For instance, the best performance of common entity linking tools displays a non-trivial variance on different benchmarking datasets — from 38.0 to 73.0 on the Macro F1 scores (Kolitsas et al., 2018). It may lead to a mismatch between the injecting knowledge and the text. In what follows, for tasks like domain-specific question answering that naturally suits the knowledge injection, the models do not necessarily render the expected performance gain (Zhang et al., 2019; Broscheit, 2019; Whitehouse et al., 2022). In the meantime, Zhang et al. (2021) points out that some-

times the redundant knowledge may unexpectedly lead to a negative infusion effect on the performance. To sum up, these results have concerned us with the actual effect of the injected knowledge towards feeding through the model pathway.

In this work, we may take a comprehensive revisiting of the grand spectrum of knowledge injection literature. To fully understand the dynamics, we design a series of quantitative and qualitative empirical protocols: (i)-we establish an ablation protocol which is often omitted by prior published papers, via injecting randomized, irrelevant, and/or (intentional) false knowledge entities; (ii)-we track down the effect caused by the injected knowledge alongside the feed-forward pathway of the model. Based upon these results on various benchmarks and methodologies, we (frustratingly) found that the injected knowledge has little or minimal effect on the actual dynamics. Besides, we further conduct (iii)-injecting random Gaussian noise into the model. Thereby we find: (i) the knowledge injection is not better than random injection; (ii) the word embeddings especially the `[cls]` embedding injected with different knowledge are highly similar; (iii) there is a possibility that the reason for the prior knowledge-injected frameworks yielding performance gain could be due to a data augmentation effect similar to noising-based data augmentation methods (Kos and Song, 2017).

Note, the goal of this paper is not to propose a novel framework, nor do we claim to have further pushed the boundary of knowledge injection. Rather, through extensive and rigorous empirical findings, we hope to alert the community to scrutinize the related methods. To sum up, we make the following contribution:

- We may conclude the current schemes fail to leverage the injected knowledge, nor to recognize the relevance of the knowledge and the text input;
- We cast a (wild) hypothesis that the injected knowledge may work as a data augmentation module, to explain the performance gain;
- We provide a new set of experimental proxies as ablations for the future work, with a full pack of code planned to be open-sourced upon publication.

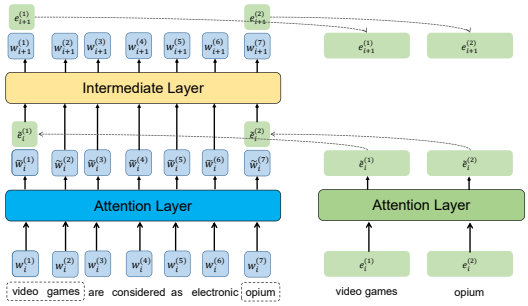

Figure 2: Word And Knowledge Embedding Fusion Process Diagram.

2 Preliminary

In this work, we prepare to conduct some additional experiments on mainstream knowledge-enhanced models, to explore the injecting knowledge's effect on the models. First, let us take a look at the mainstream knowledge injection methods.

Knowledge Graph (KG) is a directed graph G consisting of a large number of triples T and $T = \{e, r, t\}$, where e, t are the head entity and the tail entity respectively and r is the relation between e and t. KG embedding models can map the structural information of the KG into a vector space, such as TransE (Bordes et al., 2013).

Entity Linking is the entity alignment tool, like TagMe (Ferragina and Scaiella, 2010a). Specifically, given a KG and text, it detects KG's entities mentioned in the text and links them to the correct KG entry (Broscheit, 2019). After entity alignment, knowledge-injected models generally consider relation r and tail entity t as knowledge. Some knowledge-injected models use aligned KG entities as entity knowledge, such as LUKE (Yamada et al., 2020).

Knowledge Representation/Embedding refers to the embedding of the entity in the KG embedding model or embedding trained in the pre-training stage.

Knowledge Injection Methods aim to inject knowledge information from KGs into PLMs, to improve the performance of downstream tasks. According to the stage in which they inject knowledge, the methods can generally be divided into two categories, pre-training or finetuning.

Many works intend to integrate external knowledge base knowledge into PLMs, however, due

to the differences in downstream tasks and injection patterns, different knowledge-enhanced models may have differences in specific methods. Here, we take ERNIE (Zhang et al., 2019) as an example to introduce the knowledge injection methods, for ERNIE is relatively representative in these models and many subsequent works are adjusted on its basis.

To be specific, in finetuning, it firstly aligns mentions in text with the KG's entities via entity linking. Then, the sentence and related knowledge begin to be fused. As shown in Figure 2, the text and the KG entities enter the different encoders separately. Like BERT, word embedding and knowledge embedding enter the attention, intermediate, and output layers.

Here, we denote the i-th layer word embedding W_i as $\{w_i^{(1)}, \ldots, w_i^{(n)}\}$, and the i-th layer knowledge embedding E_i as $\{e_i^{(1)}, \ldots, e_i^{(m)}\}$, where n, m is the length of sentence and entities. For a word embedding $w_i^{(j)}$ and its aligned knowledge embedding $e_i^{(k)}$, they are usually fused in the intermediate layer and the process can be summarized as

$$\begin{aligned} \boldsymbol{h}_i &= \sigma(\boldsymbol{W_i^{(t)}}\tilde{\boldsymbol{w}}_i^{(j)} + \boldsymbol{W_i^{(e)}}\tilde{\boldsymbol{e}}_i^{(k)} + \boldsymbol{b_i}), \\ \boldsymbol{w_{i+1}^{(j)}} &= \sigma(\boldsymbol{W}_i^{(t)}\boldsymbol{h}_i + \boldsymbol{b}_i^{(j)}), \\ \boldsymbol{e_{i+1}^{(k)}} &= \sigma(\boldsymbol{W}_i^{(e)}\boldsymbol{h}_i + \boldsymbol{b}_i^{(e)}), \end{aligned} \tag{1}$$

where h_i is the i-th hidden state and $\sigma(\cdot)$ is the non-linear activation function. $\tilde{w}_i^{(j)}$, $\tilde{e}_i^{(k)}$ are the representation of $w_i^{(j)}$ and $e_i^{(k)}$ after entering the attention layer. We exclude a comprehensive description of the specific fusion method and refer readers Zhang et al. (2019).

For knowledge injecting in pre-training, the model usually adds a new pre-training task based on the Masked Language Model (MLM), which randomly masks off the KG's entities and lets the model predict it.

3 Models and Datasets

In this section, we introduce chosen knowledge-injection models, the downstream tasks, and datasets on which its effects are evaluated.

3.1 Models

Knowledge-injected PLMs have benefited a variety of NLP applications, especially those knowledge-intensive ones (Wei et al., 2021). However, different downstream tasks may require different types and quantities of knowledge, and knowledge injection methods for different NLP applications may vary widely. To take a comprehensive revisiting of the main knowledge-enhanced models, we choose the better performance knowledge-injected PLMs as baselines according to different downstream tasks and datasets. According to these principles, we choose LUKE (Yamada et al., 2020), ERNIE (Zhang et al., 2019), KnowBERT (Peters et al., 2019), ATOMIC-BERT (Hosseini et al., 2022), K-BERT (Liu et al., 2020) and KeBioLM (Yuan et al., 2021) as baselines, please refer to Appendix A.1 for more details.

3.2 Downstream tasks and Datasets

We choose the following tasks to demonstrate the actual performance of knowledge injection methods in the additional experiments.

Named Entity Recognition (NER) is the task of finding the corresponding span of the named entity in the given sentence. The introduction to datasets for NER tasks is in Appendix A.2.

Entity Typing is the task to find the correct type of the corresponding label entities in giving a sentence. Open Entity (Choi et al., 2018), commonly used in knowledge-enhanced PLMs, has about 6000 sentences with six entity types. Each sentence has five entity labels on average.

Relation Classification is the task of identifying the relation between label entities in a given sentence. TACRED (Zhang et al., 2017), is a relation extraction dataset with 106,264 examples. Examples in TACRED cover 42 relation types.

Question Answering is the task of answering questions such as reading comprehension questions. SQuAD1.1 (Rajpurkar et al., 2016), is a reading comprehension dataset, consisting of questions from Wikipedia articles. SQuAD 1.1 contains 107,785 question-answer pairs on 536 articles.

Word Sense Disambiguation is the task to let the model find label words' most suitable entry in the sense inventory. WiC (Pilehvar and Camacho-Collados, 2019), is a benchmark that is used for evaluating context-sensitive word embeddings. Each instance in WiC has a target word, and the task is to identify if the occurrences of the target word in the two contexts correspond to the same meaning or not.

Commonsense Causal Reasoning is the task of finding corresponding options through the causal

setup	MedicalNER + MedicalKG	MedicalNER + HowNet	MedicalNER + CnDbpedia	FinancialNER + HowNet	FinancialNER + CnDbpedia
BERT (Liu et al., 2020)	92.5	-	-	86.1	-
K-BERT (Liu et al., 2020)	94.2	93.3	93.8	87.3	87.4
original entities	94.00±0.18	93.51±0.15	93.48+0.26	**87.49**±0.08	87.37±0.19
random entities	93.89±0.33	93.52±0.22	93.55±0.25	87.35±0.19	87.46±0.11
constant entities	**94.24**±0.10	**94.24**±0.34	**94.06**±0.10	87.40±0.15	**87.80**±0.10

(a) Results of K-BERT. BERT and K-BERT come from Liu et al. (2020).

setup	BC5chem	BC5dis	NCBI	BC2GM	JNLPBA
BioBERT (Yuan et al., 2021)	92.9	84.7	89.1	83.8	79.4
KeBioLM (Yuan et al., 2021)	93.3	86.1	89.1	85.1	82.0
original entities	**93.24**±0.71	87.96±1.05	88.46±0.66	**83.99**±0.22	78.81±2.51
random entities	93.06±0.69	**88.57**±0.92	**88.91**±0.25	83.25±0.61	78.81±2.48
constant entities	92.93±0.70	87.55±0.45	88.72±0.73	83.22±0.27	**79.17**±2.24

(b) Results of KeBioLM. BioBERT and KeBioLM come from Yuan et al. (2021).

Table 1: Results of Named Entity Recognition Task. *original*, *random* and *constant entities* correspond setup 1, 2 and 3 respectively.

dependencies. COPA (Roemmele et al., 2011) is a benchmark for evaluating commonsense causal reasoning. It consists of 1000 questions, 500 each for the training and test sets. Each question has a premise and two alternatives.

4 Is the Knowledge Injection Better Than Random Injection?

In this section, we demonstrate that there is no significant difference between knowledge injection and random injection in different downstream tasks. Specifically, we design a set of ablation experiments with strictly controlled variables to explore the practical effect of knowledge information.

Some believe the knowledge-enhanced models benefit from the domain knowledge in KG or that injecting information from KGs helps the model make full of training data (Zhang et al., 2019; Liu et al., 2020). However, they omit to compare the random, irrelevant injection with normal injection. In that case, it can judge whether this injection action or the injected knowledge information (or a combination of the two) enhances the model. For this purpose, we design a new ablation experiment.

Ablation Experiment Setup. We regard linked knowledge as proper knowledge, random and constant knowledge as faulty knowledge and noise, and evaluate the effectiveness of knowledge injection methods by injecting different types of knowledge. We adopt the following settings:

1. *original entities* refers to the original knowledge injected model;
2. *random entities* refers to randomly selecting one entity from the entire entity set as a new entity input;
3. *constant entities* refers to choosing one entity node as all entity inputs regardless of the entities matching the text.

To ensure that the experimental results are not affected by other factors, we try to ensure the other experimental settings and hyperparameters are consistent with baselines. To reduce the effect of random seeds, all these experiments are run 5 times with varying random seeds.

Main Results. We conduct new ablation experiments on 5 downstream tasks. As shown in table 1 to 4, we can see that: (i) The knowledge injection is not superior to random or constant injection, and in some cases, the latter is even better than the former;(ii) The performance differences of different knowledge injections are very small, generally within 1.0, and some are even lower than 0.1.

For the named entity recognition task, Table 1 shows the results of K-BERT and KeBioLM on their own NER datasets under the setup of the ablation experiment. In K-BERT, We conduct experiments on 2 NER datasets and inject knowledge from different KGs. And we find using constant or randomized entities does not make the testing performance worse. At the same time, replacing the injected knowledge base does not change this phenomenon, and its impact on downstream tasks is also small, within 0.5.

setup	Open Entity			TACRED		
	P	R	F1	P	R	F1
BERT (Zhang et al., 2019)	76.37	70.96	73.56	67.23	64.81	66.00
ERNIE (Zhang et al., 2019)	78.42	72.90	75.56	69.97	66.08	67.97
original entities	78.81±1.05	72.15±0.92	75.33±0.41	71.09±1.62	58.15±5.88	**63.79**±3.60
random entities	**77.85**±1.13	73.12±1.07	**75.37**±0.31	68.29±5.91	58.08±5.83	63.73±3.64
constant entities	77.48±0.49	**73.28**±1.11	75.31±0.39	**71.34**±1.32	**59.86**±5.16	63.17±3.68

Table 2: Results of ERNIR on Open Entity and TACRED. BERT and ERNIE come from Zhang et al. (2019). *original*, *random* and *constant entities* correspond setup 1, 2 and 3 respectively.

setup	SQuAD 1.1	
	Dev Acc	Dev F1
BERT-large (Lan et al., 2019)	84.1	90.9
LUKE (Yamada et al., 2020)	86.1	92.3
original entities	**86.22**±0.37	92.34±0.09
random entities	86.15±0.15	92.39±0.11
constant entities	86.18±0.07	**92.40**±0.06

Table 3: Results of LUKE. BERT-large and LUKE come from Lan et al. (2019) and Yamada et al. (2020). *random* and *constant entities* correspond setup 1, 2 and 3 respectively.

setup	WiC
	Dev Acc
original entities	**69.53**±1.24
random entities	69.25±1.09
constant entities	69.31±1.01

Table 4: Results of KnowBert, *random* and *constant entities* correspond setup 1, 2 and 3 respectively.

Table 2 shows the results of ERNIE on Open Entity and TACRED. We change the batch size in TACRED from 32 to 16. Table 3 and 4 show the results of LUKE on SQuAD 1.1 and KnowBert with KBs (Wiki + WordNet) on WiC. And we choose RoBERTa base as the baseline in LUKE. These results all verify our conjecture that knowledge injection is not better than random injection.

Discussion. These results do not match our general impression of knowledge injection methods, that the injected knowledge information enhances the textual representation of the model. However, It can partially explain some odd phenomena of those knowledge-enhanced PLMs, such as little correlation between the injected knowledge and the chosen tasks. These results seem to indicate that the injection model does not fully utilize the injected knowledge information, however, we do not know whether it is the reason for this knowledge injection failure. In the next section, we track down the effect caused by the injected knowledge alongside the feed-forward pathway of the model and try to verify our conjecture.

5 Why There Is Little Difference Between Knowledge Injection And Random Injection?

To find the reasons for the knowledge injection failure, we designed new analysis experiments to track the change of knowledge information in the knowledge injection path. As we introduced before 1, the knowledge representations usually are fused in the intermediate layer of the encoder. We only need to compare the embedding similarity with different knowledge before and after the fusion process. Our experimental results demonstrate that text embeddings with different knowledge are highly similar.

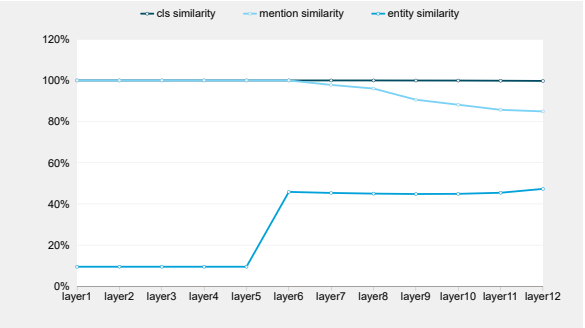

Figure 3: Schematic diagram of embedding similarity change. In this experiment, we inject *original* and *random entities* into ERNIE on the Open Entity test dataset. We count the cosine similarity of `[cls]`, mentions, and entities embedding in the hidden layers. The similarity in the figure is the absolute average of the 1000 data.

Experiment Setup. We first load the *original*, *random entities* and *constant entities* data on the trained knowledge enhanced models and print their output at each hidden layer of the encoder. Then, we compare the similarity among these outputs. We adopt cosine similarity as a measure of variation in word embeddings and entity embeddings, defined

setup	MedicalNER + MedicalKG	MedicalNER + HowNet	MedicalNER + CnDbpedia	FinancialNER + HowNet	FinancialNER + CnDbpedia
random entities	93.89±0.27	93.52±0.18	93.55±0.20	87.35±0.15	87.46±0.09
random noise	93.79±0.32	93.73±0.13	93.80±0.13	87.15±0.11	87.10±0.11

(a) Results of nosie injecting to K-BERT for NER datasets.

setup	SQuAD 1.1 Dev Acc	SQuAD 1.1 Dev F1
random entities	86.22±0.37	92.34±0.09
random noise	86.09±0.48	92.33±0.04

(b) Results of nosie injecting to LUKE

setup	Open Entity P	Open Entity R	Open Entity F1
random entities	78.81±1.05	72.15±0.92	75.33±0.41
random noise	77.28±0.54	72.98±0.42	75.07±0.06

(c) Results of nosie injecting to ERNIE.

Table 5: Results of Gaussian Noise.

as follows:

$$similarity = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|}. \quad (2)$$

Among these similarities, we only keep similarities of [cls], word embeddings related to knowledge, and knowledge embeddings. After this, we output the predictions of the same sentence with different knowledge injected. The result validates the previous inference in fine-grained dimensions.

Analysis. Our analytical experiments find that word embeddings injected with different knowledge are highly similar. Figure 3 shows the similarity change of word and entity embeddings from layer1 to layer12. From layer1 to layer5, there is no interaction between the entity and word embedding, so the similarities did not change. Starting from layer6, the entity and word embeddings begin to fuse, and the corresponding similarity begins to change, but the [cls] embedding changes are always small. After layer12, [cls] embedding inputs into the linear layer and outputs logits.

It can be found that the similarities of [cls] embeddings are very high in the whole process, generally above 99.5%. In this case, it is difficult for the model to find the difference between the three sets of inputs. This result shows that the model hardly obtains valuable information from the knowledge representation.

To further verify our point of view, we output the prediction results of the model on three sets of experimental data. Figure 4 shows the results of Open Entity, TACRED and FewRel (Han et al., 2018) in ERNIE, which load the test data by injecting *original*, *random* and *constant entities*. It is easy to find that the model has a high probability of outputting the same result for sentences

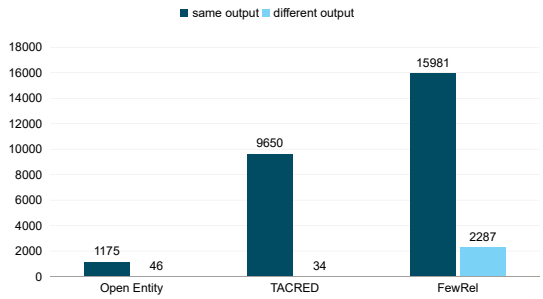

Figure 4: The output of injecting different knowledge. We only count the output of data with injected knowledge. The same output means the text injecting *original*, *random* and *constant entities* output the same predicted value. If there are differences between the three predicted values, it is attributed to the different outputs.

injected with different knowledge, and this data exceeds 99.6% in TACRED. It further illustrates that the model may not recognize the knowledge information integrated into sentences.

6 Why Does Random Injection Still Work?

Our previous experiments validated our inference that the injected knowledge information may not affect the actual performance of the model. However, in our experiments, random injection seems still performs better than the non-injected models. In this section, we further speculate that knowledge injection has a data augmentation effect similar to injecting noise.

Experiment Setup. To explore the actual effect of the random injection, we design two related experiments: a. Replace the injected knowledge with Gaussian white noise and compare it with the results of *random entities*; b. Compare the overfitting

Setup	Open Entity		
	P	R	F1
BERT	76.37	70.96	73.56
ERNIE*	77.24±0.71	69.54±0.63	73.18±0.32
random knowledge	77.02±0.36	70.78±0.54	73.76±0.18

Table 6: Results of ERNIE on Open Entity. BERT comes from Zhang et al. (2019). ERNIE* is the result of our reproduction Zhang et al. (2019) pre-training. *random knowledge* corresponds setup 2.

setup	COPA
	Acc
BERT large	78.60±1.11
ATOMIC-BERT*	78.84±1.62
original knowledge	75.60±2.81
random knowledge	68.40±7.69

Table 7: Results of ATOMIC-BERT on COPA. *original* and *random knowledge* correspond setup 1 and 2 respectively. ATOMIC-BERT* is the result of our reproduction Hosseini et al. (2022).

degree of the model with or without knowledge injection (including Gaussian white noise). We use the loss and accuracy gap between the training and validation phase as evaluation metrics for model overfitting degrees.

Discussion. Indeed, given all the aforementioned empirical results, it is still undeniable that knowledge-injected frameworks have a positive outcome from the perspective of eventual performance. To form a full-circle study, we hereby cast a hypothesis, perhaps wild, that the injected knowledge is picked by the model as a data augmentation module.

Table 5a to 5c shows the comparison of injecting *random entities* and Gaussian white noise in different downstream tasks. We can see the difference between the two is minimal, almost within 0.3 of the F1 score. It seems to indicate that knowledge-injected methods may work like noising-based data augmentation methods.

We further conduct an extra set of experiments on Open Entity, by simply injecting gaussian noises. At the convergence point, we curate and report both the loss gap between the training and dev set. Ordinarily, the larger the gap, the more overfitted the trained model is. The result is summarized in the text as follows: (i)-injecting original knowledge entities, random entities, and random noises all manage to reduce the loss gap, i.e. help alleviate the overfitting; (ii)-injecting random noise has the most notable effect from this metric, that it reduces the gap by over **0.01** (e.g. from **0.176** to **0.163**); (iii)-through manipulating the magnitude of the randomized knowledge vector, we see the gap becomes smaller (but perhaps hurt the overall performance). In that regard, this pattern points out a resemblance of injecting knowledge with an incorporated data augmentation module.

At last, as an empirical study, we do not intend to make a deterministic conclusion. The hypothesis we case — that the knowledge injection acts as a data augmentation module — is based on their similar performance pattern and perhaps is only one among many other possibilities. We hope to use this finding to motivate the community to provide more theoretical and comprehensive evidence.

7 Knowledge Integration in Pre-training Stage

Many knowledge-enhanced PLMs design a new task in the pre-training phase and inject knowledge through the task. To explore these impacts on knowledge injection, we also designed a series of new experiments. Through experiments, we found that short-term knowledge injection (1-3 epochs) into pre-training tasks has little effect on downstream tasks.

Experiment Details. Since these new tasks are mostly based on MLM, we only keep the original and random knowledge as an experimental setup. The experimental setup is as follows:

1. *original knowledge* refers to using the original experimental setup;

2. *random knowledge* refers to using random knowledge to pre-train the model.

To investigate the impact of different new pre-training tasks, we choose ERNIE and ATOMIC-BERT as the research baselines. In ERNIE, we pre-train the model under *random knowledge*'s and *original knowledge*'s set up for one epoch each. And the *random knowledge* only changes the aligned entities of the mentions in the text. In ATOMIC-BERT, we pre-train the model for three epochs. Different from ERNIE, *random knowledge* in ATOMIC-BERT replaces the current triple's tail with the other triple's tail and converts the new triple to text, which turns "A is B" into "A is C".

Discussion. Table 6 and 7 shows the results of the two experiments. We can see that: (i) Short-term new pre-training tasks (1-3 epochs) has little effect on the performance of the knowledge-enhanced models. Since the difference between ERNIE* and BERT, ATOMIC-BERT* and BERT large are very small. (ii) Breaking the pre-trained corpus's text structure may drastically reduce model performance, however replacing the aligned entities may not. It seems to indicate that this kind of pre-training task does not enable the model to gain the corresponding knowledge.

8 Related Works

8.1 Knowledge-Enhanced PLMs

Since the large-scale application of pre-trained models in the NLP field, many works expect to improve the downstream tasks' performance by integrating external knowledge. ERNIE (Zhang et al., 2019), CokeBERT (Su et al., 2021), PELT (Ye et al., 2022), KnowBert(Peters et al., 2019) , KEPLER (Wang et al., 2021b), CoLAKE (Sun et al., 2020), LUKE (Yamada et al., 2020), K-BERT (Liu et al., 2020), K-Adapter (Wang et al., 2021a), MoP (Meng et al., 2021), KELM (Lu et al., 2022), ATOMIC-BERT (Hosseini et al., 2022), SentiLARE (Ke et al., 2020), BERT-MK (He et al., 2020), KeBioLM (Yuan et al., 2021) all propose a new method to enhance the model output through the external knowledge graphs.

Some works expect to introduce entity-level information to improve the performance of entity-related tasks such as NER, entity typing, relation classification, and machine reading comprehension while pre-training and finetuning (Wei et al., 2021). **ERNIE** (Zhang et al., 2019) interprets and implements this idea, by introducing the entities' knowledge representation via entity linking. **LUKE** (Yamada et al., 2020) further proposes an entity-aware self-attention mechanism and computes different attention scores regarding words or entities. **KnowBert**(Peters et al., 2019) improves the entity linker for entity disambiguation and recombines knowledge and word representations to inject knowledge. However, they do not fine-grained verify whether entity knowledge enhances the model.

There are also some works integrating the relation triples or subgraphs in KGs to make the model get more information. **K-BERT** (Liu et al., 2020) converts the relation triples and context into the sentence tree and then uses soft position and visible matrix to limit the impact of knowledge noise. CoLAKE (Sun et al., 2020) and KELM (Lu et al., 2022) integrate the subgraphs' information to enhance the PLMs. They also omit the question of whether injecting knowledge is helpful.

8.2 Interpretable Analysis In PLMs

Peters et al. (2019); Jiang et al. (2020); Cao et al. (2021) have proved that pre-trained language models can acquire substantial factual knowledge via pre-training on large-scale unlabeled data. Li et al. (2022) further analyzes that PLMs capture factual knowledge more by the close position and high co-occurrence. Zhang et al. (2021) points out redundant and irrelevant knowledge injections in knowledge-enhanced models, which lead to ineffective knowledge injection. However, it only analyzes the phenomenon of negative knowledge infusion and omits other reasons that may lead to the failure of knowledge injection.

9 Conclusion

We aim to find out if the current knowledge injection framework works and explore its actual profound mechanism. Our comprehensive experiments demonstrate a frustrating conclusion that the injected knowledge is not picked up by the model in our expected manner. Furthermore, our analytical experiments prove that the model facilitates little effect from the injected knowledge. Among many possibilities, we cast a hypothesis that the injected knowledge may act like noising-based data augmentation methods. We need to rethink how the knowledge-injected models work and find a proper way to make them full of the injecting knowledge information.

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: Pretrained language model for scientific text. In *EMNLP*.

Olivier Bodenreider. 2004. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270.

A. Bordes, N. Usunier, Alberto Garcia-Duran, J. Weston, and O. Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Neural Information Processing Systems*.

Samuel Broscheit. 2019. Investigating entity knowledge in bert with simple neural end-to-end entity linking.

In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 677–685.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Boxi Cao, Hongyu Lin, Xianpei Han, Le Sun, Lingyong Yan, Meng Liao, Tong Xue, and Jin Xu. 2021. Knowledgeable or educated guess? revisiting language models as knowledge bases. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1860–1874.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 87–96.

Nigel Collier and Jin-Dong Kim. 2004. Introduction to the bio-entity recognition task at jnlpba. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (NLPBA/BioNLP)*, pages 73–78.

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.

Rezarta Islamaj Doğan, Robert Leaman, and Zhiyong Lu. 2014. Ncbi disease corpus: a resource for disease name recognition and concept normalization. *Journal of biomedical informatics*, 47:1–10.

P. Ferragina and U. Scaiella. 2010a. Tagme: on-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada, October 26-30, 2010*.

Paolo Ferragina and U Scaiella. 2010b. Tagme: on-the-fly annotation of short text fragments (by wikipedia entities). In *ACM Conference on Information and Knowledge Management (CIKM)*, pages 1625–1628. ACM Press.

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.

Bin He, Di Zhou, Jinghui Xiao, Xin Jiang, Qun Liu, Nicholas Jing Yuan, and Tong Xu. 2020. Bert-mk: Integrating graph contextualized knowledge into pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2281–2290.

Pedram Hosseini, David A Broniatowski, and Mona T Diab. 2022. Knowledge-augmented language models for cause-effect relation classification. In *ACL 2022 Workshop on Commonsense Representation and Reasoning*.

Jena D Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. (comet-) atomic 2020: On symbolic and neural commonsense knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6384–6392.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Pei Ke, Haozhe Ji, Siyang Liu, Xiaoyan Zhu, and Minlie Huang. 2020. Sentilare: Sentiment-aware language representation learning with linguistic knowledge. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6975–6988.

Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. 2018. End-to-end neural entity linking. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 519–529.

Jernej Kos and Dawn Song. 2017. Delving into adversarial attacks on deep policies. *arXiv preprint arXiv:1705.06452*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*.

Jiao Li, Yueping Sun, Robin J Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wiegers, and

Zhiyong Lu. 2016. Biocreative v cdr task corpus: a resource for chemical disease relation extraction. *Database*, 2016.

Shaobo Li, Xiaoguang Li, Lifeng Shang, Zhenhua Dong, Cheng-Jie Sun, Bingquan Liu, Zhenzhou Ji, Xin Jiang, and Qun Liu. 2022. How pre-trained language models capture factual knowledge? a causal-inspired analysis. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1720–1732.

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 *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2901–2908.

Yinquan Lu, Haonan Lu, Guirong Fu, and Qun Liu. 2022. Kelm: Knowledge enhanced pre-trained language representations with message passing on hierarchical relational graphs. In *ICLR 2022 Workshop on Deep Learning on Graphs for Natural Language Processing*.

Zaiqiao Meng, Fangyu Liu, Thomas Clark, Ehsan Shareghi, and Nigel Collier. 2021. Mixture-of-partitions: Infusing large biomedical knowledge graphs into bert. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4672–4681.

Matthew E. Peters, Mark Neumann, Robert L Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. In *EMNLP*.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. 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 1267–1273.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*.

Larry Smith, Lorraine K Tanabe, Cheng-Ju Kuo, I Chung, Chun-Nan Hsu, Yu-Shi Lin, Roman Klinger, Christoph M Friedrich, Kuzman Ganchev, Manabu Torii, et al. 2008. Overview of biocreative ii gene mention recognition. *Genome biology*, 9(2):1–19.

Yusheng Su, Xu Han, Zhengyan Zhang, Yankai Lin, Peng Li, Zhiyuan Liu, Jie Zhou, and Maosong Sun. 2021. Cokebert: Contextual knowledge selection and embedding towards enhanced pre-trained language models. *AI Open*, 2:127–134.

Tianxiang Sun, Yunfan Shao, Xipeng Qiu, Qipeng Guo, Yaru Hu, Xuan-Jing Huang, and Zheng Zhang. 2020. Colake: Contextualized language and knowledge embedding. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3660–3670.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuan-Jing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2021a. K-adapter: Infusing knowledge into pre-trained models with adapters. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1405–1418.

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. *Transactions of the Association for Computational Linguistics*, 9:176–194.

Xiaokai Wei, Shen Wang, Dejiao Zhang, Parminder Bhatia, and Andrew Arnold. 2021. Knowledge enhanced pretrained language models: A compreshensive survey. *arXiv e-prints*, pages arXiv–2110.

Chenxi Whitehouse, Tillman Weyde, Pranava Madhyastha, and Nikos Komninos. 2022. Evaluation of fake news detection with knowledge-enhanced language models. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 16, pages 1425–1429.

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)*. Association for Computational Linguistics.

Deming Ye, Yankai Lin, Peng Li, Maosong Sun, and Zhiyuan Liu. 2022. A simple but effective pluggable entity lookup table for pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 523–529.

Zheng Yuan, Yijia Liu, Chuanqi Tan, Songfang Huang, and Fei Huang. 2021. Improving biomedical pretrained language models with knowledge. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 180–190, Online. Association for Computational Linguistics.

Ningyu Zhang, Shumin Deng, Xu Cheng, Xi Chen, Yichi Zhang, Wei Zhang, Huajun Chen, and Hangzhou Innovation Center. 2021. Drop redundant, shrink irrelevant: Selective knowledge injection for language pretraining. In *IJCAI*, pages 4007–4014.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical*

Methods in Natural Language Processing (EMNLP 2017), pages 35–45.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *Proceedings of ACL 2019*.

A Appendix

A.1 Baseline Details

LUKE (Yamada et al., 2020) chooses to enhance RoBERTa with the knowledge from Wikipedia. It use a new pre-training task which involves predicting randomly masked words and entities in a large entity-annotated corpus retrieved from Wikipedia. At the same time, LUKE also input wikipedia entities into the model which are based on the sentences in finetuning for the question answering dataset SQuAD1.1. In addition to injecting knowledge, LUKE propose an entity-aware self-attention mechanism and considers the types of tokens (words or entities) when computing attention scores (Yamada et al., 2020).

ERNIE (Zhang et al., 2019) injects entities knowledge from Wikipedia into BERT in pretraining and finetuning. ERNIE first uses TAGME (Ferragina and Scaiella, 2010b) to link entities mentioned in context to their corresponding entities in KG, then injects the corresponding entities embdedding into language models. Embeddings of the corresponding entities are trained on triples from WikiData via TransE (Zhang et al., 2019).

KnowBERT (Peters et al., 2019) integrates knowledge from WordNet and Wikipedia into BERT, and demonstrates improved perplexity and ability to recall facts. KnowBERT first trains an integrated entity linker to retrieve relevant entity embeddings, which is used to entity disambiguation. Then, the model use a Knowledge Attention and Recontextualization (KAR) mechanism to combine the knowledge representation and contextual word representations.

ATOMIC-BERT (Hosseini et al., 2022) adds a new pre-training corpus to integrate causal knowledge of ATOMIC (Hwang et al., 2021) on the basis of the original BERT. It first converts triples in ATOMIC knowledge graph to natural language texts, and then pretrains model on the generated text via MLM.

K-BERT (Liu et al., 2020) choose CN-DBpedia, HowNet and MedicalKG as external knowledge base. K-BERT is devised to feed a structural tree that is decoded from the sentence into a pretrained language model. The construction of the structural tree is driven by both the sentence itself together with an external knowledge graph. However, it inevitably brings the problem of knowledge noise. To solve this problem, K-BERT proposed to special a seeing layer,which make the injected triples can only affect its corresponding subject.

KeBioLM (Yuan et al., 2021) injects entity knowledge from UMLS (Bodenreider, 2004) by fusing the entities in the knowledge base and mentions in the text in the middle layer. Firstly, it uses a function to recognize if a span is an entity mention. then, it links to a set of the mention's k-nearest entities and integrate the entity embedding and the word embedding in the hidden layer, as the input of the model.

A.2 Dataset Details

Finance NER [1] includes 3000 financial news articles manually labeled, which contain over 65,000 name entities.

Medicine NER [2] is the Clinical Named Entity Recognition(CNER) task that was released in CCKS 2017. The dataset mainly extracts medical-related entity names from electronic medical records.

BC5-chem & BC5-disease (Li et al., 2016) contain 1500 PubMed abstracts that extract chemical and disease entities respectively.

NCBI-disease (Doğan et al., 2014) includes 793 PubMed abstracts that had been detected disease entities.

BC2GM (Smith et al., 2008) is a dataset including 20K PubMed sentences extracting gene entities.

JNLPBA (Collier and Kim, 2004) is a dataset includeing 2,000 PubMed abstracts that has been identified molecular biology-related entities.

A.3 Dataset License

We only find three dataset licenses,which is as following.

SQuAD: CC-BY-SA 4.0

WiC: CC BY-NC 4.0

COPA: BSD 2-Clause

[1] https://embedding.github.io/evaluation/#extrinsic

[2] https://biendata.net/competition/CCKS2017_2/