LANGUAGE MODEL WITH PLUG-IN KNOWLEDGE MEMORY

Anonymous authors
Paper under double-blind review

ABSTRACT

Large-scale pre-trained language models (PLM) have made impressive results in a wide range of NLP tasks and it has been revealed that one of the key factors to their success is the parameters of these models implicitly learn various types of knowledge in the pre-training corpus. However, encoding knowledge implicitly in the model parameters has two fundamental drawbacks. First, the knowledge is neither editable nor scalable once the model is trained, which is especially problematic in that knowledge is consistently evolving. Second, it lacks interpretability and prevents us from understanding what kind of knowledge PLM needs to solve a certain task. In this paper, we introduce PlugLM, a pre-training model with differentiable plug-in memory (DPM). The key intuition behind is to decouple the knowledge storage from model parameters with an editable and scalable key-value memory and leverage knowledge in an explainable manner by knowledge retrieval in the DPM. We conduct extensive experiments under various settings to justify this design choice. In domain adaptation setting, PlugLM could be easily adapted to different domains with pluggable in-domain memory—obtaining 3.95 F1 improvements across four domains, without any in-domain training. PlugLM could also keep absorbing new knowledge after pre-training is done by knowledge updating operation in the DPM without re-training. Finally, we show that by incorporating training samples into DPM with knowledge prompting, PlugLM could further be improved by the instruction of in-task knowledge.

1 INTRODUCTION

Large-scale pre-trained language models (PLM) (Peters et al., 2018; Devlin et al., 2019; Radford et al., 2018) have become a revolutionary breakthrough in NLP area. Optimized by carefully designed self-supervised objectives on unlabeled corpus and fine-tuned on downstream tasks, PLMs perform remarkably well in a wide range of NLP benchmarks. Recent studies (Warstadt et al., 2019; Petroni et al., 2019) have revealed that one of the key factors to the success of PLMs is that the parameters of these models implicitly learn various types of knowledge in the pre-training corpus. Owing to these learned syntactic, semantic, factual and commonsense knowledge, PLMs show great understanding, generalization and reasoning abilities (Rogers et al., 2020; Izacard et al., 2022) in multiple downstream tasks.

Geva et al. (2021) pointed out that the knowledge of PLMs is implicitly encoded in the feed-forward layers (FFN) of Transformer architecture. FFN layers can be viewed as key-value memories (Weston et al., 2014; Sukhbaatar et al., 2015), where the first linear layer of FFN acts like a set of sparsely activated keys detecting input pattern and the second layer is the corresponding value where knowledge is stored. And to aggressively capture more knowledge, larger PLMs are continuously proposed, from 110M BERT (Devlin et al., 2019) to 530B MT-NLG (Smith et al., 2022), yet PLM has not reached its upper bound (Qiu et al., 2020).

However, we still have a question: **Is it the optimal way to encode knowledge implicitly for PLMs?** We argue that the implicit knowledge encoding approach has two fundamental drawbacks. First, the learned knowledge is neither editable nor scalable once the model is trained. Nevertheless, the world knowledge is actually infinite and evolving. We thus would never expect an ever-large model to capture all the knowledge in its parameters and to be continuously re-trained to encode the newly coming knowledge. Second, the current PLMs lack interpretability in the knowledge level.

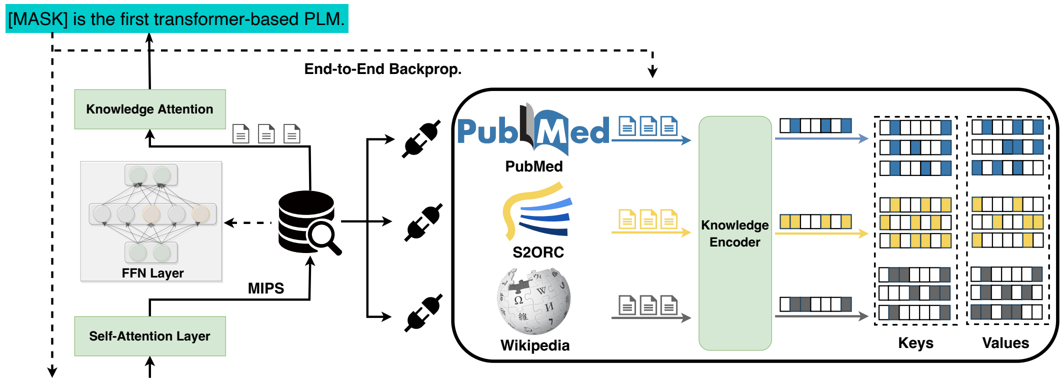


Figure 1: Overview of our approach. We equip PLM with a Differentiable Plug-in Memory (DPM) by which PLM could store and leverage knowledge in an explainable manner.

Implicit knowledge encoding fails to provide provenance for model's prediction and prevents us from understanding what kind of knowledge does PLMs require when performing reasoning for a certain task.

In this work, we propose a novel architecture of PLM, PlugLM, which decouples the knowledge storage from Transformer architecture and explicitly leverages the knowledge in an explainable manner. As shown in Figure 1, we balance the functionality of FFN layer with a differentiable plug-in key-value memory (DPM), which is highly scalable and editable. Each slot of DPM is to encode the knowledge of a specific document to a pair of key and value, and thus we can explicitly retrieve relevant knowledge from DPM for each sample.

To justify the design choice of decoupling the knowledge from PLM, we conduct extensive empirical evaluations under different settings. In domain adaptation setting, PlugLM could be easily adapted to different domains with pluggable in-domain memory—obtaining averaged 3.95 F1 improvements across four domains and up to 11.55 F1 improvement on CS-relevant ACL-ARC dataset, without any in-domain training. PlugLM could also keep absorbing new knowledge after pre-training is done by knowledge updating operation in the DPM, with an improvement of 4 F1 scores in LINNAEUS NER dataset. Finally, we show that by incorporating training samples into DPM with knowledge prompting, PlugLM could further be improved by the instruction of in-task knowledge.

2 Language Model with Differentiable Plug-in Memory

2.1 Preliminary

Feed-forward Layers Transformer (Vaswani et al., 2017), the backbone for all PLMs, is made of stacked self-attention (Self-Attn) and feed-forward layers (FFN). While Self-Attn captures the contextual interaction among inputs, the FFN process each input independently. Let $x \in \mathbb{R}^{d_1}$ be a vector as input to FFN layer, we could formulate the FFN as follows:

$$\mathrm{FFN}(x) = f(x \cdot K^\top) \cdot V \tag{1}$$

where $K, V \in \mathbb{R}^{d_2 \times d_1}$, f is an activation function such as RELU (Devlin et al., 2019).

Key-Value Memory Network The Key-Value Memory (KVMN) is based on the Memory Network (Weston et al., 2014; Sukhbaatar et al., 2015). It corresponds to d_2 key-value pairs ($K, V \in \mathbb{R}^{d_2 \times d_1}$) and they are the generalization of the way knowledge is stored (e.g., context in Dialogue (Eric et al., 2017), documents in QA (Miller et al., 2016)). For an input $x \in \mathbb{R}^{d_1}$, there are two stages for KVMN. First, the lookup (addressing) stage would compute the matching degree between x and each key of K. In the second stage, x would be transformed by the weighted sum of V according to the distribution of the matching degree in the first stage. We can formally define it as:

$$\text{MemoryNetwork}(x) = \text{Softmax}(x \cdot K^\top) \cdot V \quad (2)$$

Comparing equation (1) and (2), we could find that the FFN is an unnormalized version of MemoryNetwork. The keys in FFN are pattern detectors and would be activated only when certain patterns occur in the input. This explains how FFN stores knowledge in a key-value manner (Geva et al., 2021; Sukhbaatar et al., 2019).

2.2 Overall Architecture

The overall architecture of our PlugLM is illustrated by Figure 1. Similar to BERT (Devlin et al., 2019), the backbone of our model is a multi-layer bidirectional Transformer encoder (Vaswani et al., 2017). Following the line of works that take the view of FFN as KVMN, PlugLM involves balancing FFN with Plug-in Differential key-value Memory (DPM) and instead of storing all knowledge in the model parameters, PlugLM uses a *Knowledge Encoder* (KnowEnc$_\theta$, parameterize by θ) to transform a specific document in the knowledge base to a key-value pair. Therefore, in PlugLM, for the basic block of each layer, we can flexibly employ FFN to encode the intrinsic language understanding knowledge or DPM to encode the external knowledge from a textual corpus.

DPM Construction In this paper, we view each knowledge $d = \{T_1, T_2, ..., T_{|d|}\}$ as consecutive tokens from unlabeled corpora and the knowledge base is $\mathbb{D} = \{d_1, d_2, ..., d_{|\mathbb{D}|}\}$. In the pre-training phase, Wikipedia is chosen as the source of knowledge and in the domain adaptation setting, corpora from other domains are treated as knowledge sources detailed in §3.1. For knowledge base sized of $|\mathbb{D}|$, we get dense vector representation for each knowledge d_n from KnowEnc$_\theta$ and use separate mapping function to project it to the key spaces and value spaces:

$$K_n = W_k \cdot h_n \qquad V_n = W_v \cdot h_n \quad (3)$$

where h_n is from KnowEnc$_\theta(d_n)$. W_k and W_v are trainable parameters and the bias term is omitted for brevity. We get $\mathbb{K}$ and $\mathbb{V}$ for DPM $\langle \mathbb{D}, \mathbb{K}, \mathbb{V} \rangle$:

$$\mathbb{K} = \{K_1, K_2, ..., K_{|\mathbb{D}|}\} \qquad \mathbb{V} = \{V_1, V_2, ..., V_{|\mathbb{D}|}\} \quad (4)$$

Knowledge Encoder KnowEnc$_\theta$ converts a sequence of tokens into dense representation which is supposed to have the properties of alignment and uniformity (Wang & Isola, 2020). In this paper, we choose the following function as our KnowEnc$_\theta$. For a given knowledge d_n:

$$h_n = \text{AttentivePooling}(\text{TokenEmbedding}(d_n) + \text{PositionEmbedding}(d_n)) \quad (5)$$

where AttentivePooling function (Xu et al., 2021) corresponds to a trainable pattern detector aggregating information from a sequence of input. We give its pseudo-code in Appendix A.

Knowledge Retrieval For hidden states $h \in \mathbb{R}^{l \times d}$ from Self-Attn, FFN would transform h with unnormalized key-value memory as in Equation (1). Our key insight is that instead of retrieving implicit knowledge from FFN, we conduct Maximum Inner Product Search (MIPS) to retrieve named knowledge from $\langle \mathbb{D}, \mathbb{K}, \mathbb{V} \rangle$ where each triple corresponds to one knowledge along with its key and value representation. For h, we first get its sentence-level representation by an attentive pooling function $h' = \text{AttentivePooling}(h)$, then we use h' as the query vector for $\langle \mathbb{D}, \mathbb{K}, \mathbb{V} \rangle$ to get Top-N knowledge and corresponding values by MIPS:

$$K_{h'} = \text{Top-N}(\text{MIPS}(h', \mathbb{K})) \quad V_{h'} = \{V_i \text{ if } K_i \text{ in } K_{h'}\} \quad D_{h'} = \{D_i \text{ if } K_i \text{ in } K_{h'}\} \quad (6)$$

where Top-N also corresponds to the indexing operation. $D_{h'}$ is the explicit knowledge model used to get the current prediction. By knowledge retrieval, we explore an interpretable way to incorporate knowledge into the model and direct modification on $\mathbb{D}$ of DPM empowers the model with much flexibility and scalability in various settings as discussed in §3.1 and §3.2.

Knowledge Attention For Top-N retrieved knowledge $\langle D_{h'}, K_{h'}, V_{h'} \rangle$, we use knowledge attention to incorporate it:

$$\text{KnowledgeAttention}(h, K_{h'}, V_{h'}) = \text{Softmax}(\frac{hK_{h'}^\top}{\sqrt{d}})V_{h'} \quad (7)$$

$$O = \text{LayerNorm}(h + \text{KnowledgeAttention}(h, K_{h'}, V_{h'})) \quad (8)$$

For more fine-grained interaction, we also use a multi-head version as in Vaswani et al. (2017) and d is the head dimension.

2.3 TRAINING

There are two phases in our framework: pre-training and fine-tuning. In the pre-training phase, to make the whole training process end-to-end trainable, we use asynchronous index refreshing to optimize our model as done in Guu et al. (2020) and Cai et al. (2021). Concretely, we update the indices of DPM every T steps. The MIPS results are based on the stale index while the scores of selected Top-N results are recomputed using KnowEnc_θ which facilitates the gradient flow back to the knowledge retriever and knowledge encoder. The training objective is Masked Language Modeling (Devlin et al., 2019) where we randomly mask tokens in a sentence and ask our model to predict it. More details about model architecture and pre-training are shown in Appendix B. In the fine-tuning phase, the $\mathbb{K}$ and $\mathbb{V}$ of DPM are fixed, and we view it as an editable and scalable knowledge lookup table.

3 EXPERIMENTS

In this paper, we mainly try to decouple the knowledge storage from PLM and leverage knowledge in an explainable way. With decoupled knowledge $\langle\mathbb{D}, \mathbb{K}, \mathbb{V}\rangle$, we conduct comprehensive experiments with respect to the modification of $\mathbb{D}$ in different settings. First, in §3.1 we demonstrate the advantage of PlugLM in domain adaptation by flexibly switching domain-specific DPM without changing model parameters. Second, we show that PlugLM can adjust to evolving knowledge with knowledge updating operation on $\langle\mathbb{D}, \mathbb{K}, \mathbb{V}\rangle$ in §3.2. In §3.3, we show that with carefully designed knowledge prompting, PlugLM could further improve its performance by enlarging the scope of knowledge to include training samples as in-task knowledge.

3.1 PLUG-IN MEMORY FOR DOMAIN ADAPTATION

Learning robust and transferable representation has been the core of language model pre-training (Peters et al., 2019). For the general-purposed PLM to generalize well on domain-specific tasks, endowing the model with domain knowledge via in-domain training remains the go-to approach (Gururangan et al., 2020; Whang et al., 2020; Zhang et al., 2020). In this section, we measure the effectiveness of PlugLM for domain adaptation, in which we use a domain-specific DPM to adapt the model, without any in-domain training. This is a challenging task for the current PLM because sometimes it is computationally unaffordable to keep training the model (Smith et al., 2022) and it can not guarantee the generalization across multiple domains due to catastrophic forgetting problem (Kirkpatrick et al., 2017). We consider two adaptation settings: domain adaptive post-training and in-domain pre-training. The former is conducted after PLM was trained on the general domain and the latter trains a domain-specific PLM from scratch.

3.1.1 DOMAIN ADAPTIVE POST-TRAINING

Following Gururangan et al. (2020), we conduct experiments on four domains: BIOMED, CS, NEWS and REVIEWS across eight domain-specific downstream tasks, in both low and high resource settings. More details can be found in Appendix C. When fine-tuning on downstream classification tasks, we pass the final layer [CLS] token representation to a task-specific feed-forward layer for prediction following the standard practice in Devlin et al. (2019).

We have the following baselines: **WikiBERT** uses the architecture of BERT_{base} (Devlin et al., 2019) and is pre-trained on Wikipedia. To adapt WikiBERT to other domains, we use DAPT following the training setting in Gururangan et al. (2019). **REALM** (Guu et al., 2020) and **PlugLM** are models that have an dexternal knowledge base and can be simply adapted to other domains with a different base. We have two variants for adaptation: DAA, short for Domain Adaptive Addition, appends domain knowledge to the knowledge base and DAR, Domain Adaptive Replacement, replaces general knowledge with domain-specific knowledge.

We also include the results of ¬DAPT, ¬DAA and DACT. The former two use irrelevant domain corpora for post-training and knowledge bank construction, which are used to test the robustness of the adaptation method and rule out the factor that improvements might be attributed simply to

Model	BIOMED		CS		NEWS		REVIEWS			
	CHEM.	RCT†	ACL.	SCI.	HYP.	AG.†	HP.†	IMDB†	Avg. Gain	Avg. Cost
WikiBERT	77.72	86.52	61.58	79.95	83.54	93.38	67.62	89.79	-	-
+ DAPT	78.24	86.71	67.56	80.82	86.22	93.49	68.11	90.12	+1.40	47.7 h
¬ DAPT	75.82	86.11	62.11	78.42	80.12	93.31	68.11	89.54	-0.82	-
+ DACT	76.34	86.11	61.19	78.56	80.52	93.29	68.08	89.88	-0.77	-
REALM	78.28	85.12	62.07	78.41	84.12	92.58	67.06	90.56	-	-
+ DAA	79.32	85.98	68.92	80.41	85.36	92.61	68.51	**93.01**	+1.98	6.3 h
¬ DAA	77.61	85.12	64.78	75.31	82.28	92.41	66.13	91.21	-0.41	-
+ DAR	80.56	85.32	70.12	81.16	86.58	93.01	67.42	92.16	+2.26	6.3 h
PlugLM	78.02	87.12	63.77	78.56	84.32	93.23	67.83	91.24	-	-
+ DAA	82.56	88.13	72.51	**83.00**	88.16	**94.11**	69.28	92.56	+3.28	**0.16 h**
¬ DAA	77.98	86.13	64.78	78.13	84.18	92.99	67.56	90.88	-0.18	-
+ DAR	**83.80**	**88.98**	**75.32**	82.56	**89.26**	93.55	**69.41**	92.78	**+3.95**	**0.16 h**

Table 1: Performance of domain adaptive post-training. Each result is averaged with five different random seeds. Reported results are test macro-F1, except for RCT and CHEMPROT, for which we report micro-F1, following Beltagy et al. (2019). † denotes high-resource setting. The DAA and DAR substantially outperforms existing DAPT and REALM-based methods with no additional in-domain training. The best scores are in bold, and the second best scores are underlined.

exposure to more data[1]. For DACT, Domain Adaptive Continual Training, we sequentially post-train WikiBERT in different domains in the hope that it can capture and store knowledge from various domains in a lifelong learning manner (Rostami, 2021).

The results are shown in Table 1. The Avg.Cost is the cost for adaptation measured by hour. For WikiBERT, it's the time to post-train model in domain-specific corpus. For REALM and PlugLM, it is the time to encode domain knowledge into the knowledge bank. We can observe: (1) In-domain training helps model better generalize to tasks requiring domain knowledge while irrelevant knowledge misleads the model and causes performance degradation. And by comparing ¬DAPT and ¬DAA, it shows that models with external knowledge base (PlugLM and REALM) are more robust when faced with noisy out-of-domain knowledge. (2) For the model that implicitly encodes knowledge in the parameters, it fails to generalize across domains as the result of DACT indicates. For example, in CS domain, we keep training model in NEWS domain after DAPT in CS domain and fine-tune it on the CS downstream tasks. It performs on par with model that is never exposed to CS domain (¬DAPT). While PlugLM could alleviate this catastrophic forgetting problem by implicitly storing all kinds of knowledge in DPM and using it for the specific domain. (3) Direct modification on external memory helps PlugLM efficiently and effectively adapt to different domains without in-domain training. In 254 times less time compared with DAPT and in 40 times less time compared with REALM, PlugLM significantly outperforms DAPT and REALM-based methods.

To give a more explainable illustration of how PlugLM works, in Figure 2, we present a visualization for the distribution of actual retrieved knowledge for DAA, DAR and original PlugLM. We randomly sample 50 samples from ACL-ARC test set and check what kind of knowledge does PlugLM use to solve CS-specific tasks. A clear pattern here is that with more domain knowledge involved, the model performs better (63.77, 72.51 and 75.32) and surprisingly, although pre-trained on the general domain, the PlugLM has managed to learn what to retrieve when there are both general knowledge and domain-specific knowledge in DPM shown in DAA visualization.

3.1.2 IN-DOMAIN PRE-TRAINING

In-domain pre-training is another line of works to train a domain-specific PLM from scratch like BioBERT (Lee et al., 2019) and SciBERT (Beltagy et al., 2019) in the biomedical and scientific

[1]Following Gururangan et al. (2020), we use the following irrelevant domain mapping: for NEWS, we use a CS LM; for REVIEWS, a BIOMED LM; for CS, a NEWS LM; for BIOMED, a REVIEWS LM.

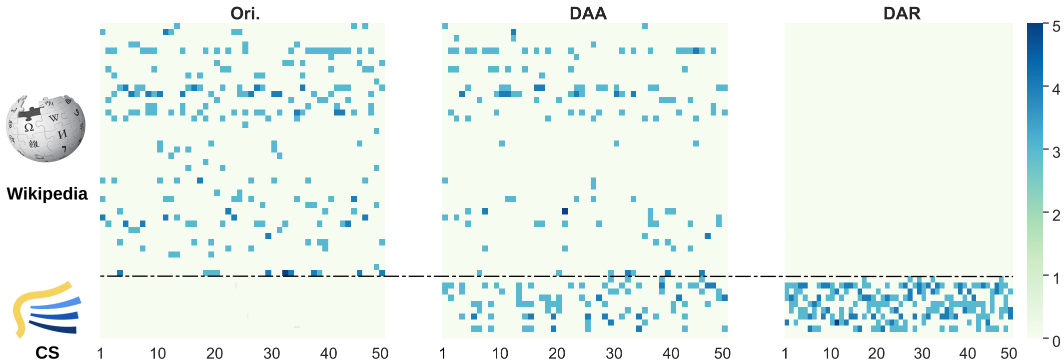


Figure 2: Knowledge retrieval visualization. Each column is one sample and the row is the index of retrieved knowledge in DPM. Their corresponding F1 scores are 63.77, 72.51 and 75.32.

Question	Answer	Prediction	Label
How much of Jacksonville is made up of water?	According to the United States Census Bureau, the city has a total area of 874.3 square miles (2,264 km^2), making Jacksonville the largest city in land area in the contiguous United States; of this, 86.66% (757.7 sq mi or 1,962 km^2) is land and ; 13.34% (116.7 sq mi or 302 km^2) is water.	Entailment	Entailment
Knowledge	this article lists the 3, 143 states of america. the 50 states of "... by the united states census bureau...the united states census bureau (usc ##b) , officially the bureau of the census...% white, 9. 3 % african american, 0.3 % native american, 2.9 % asian, 0.1 % pacific islander, 0.2 % from other races, and 3.9 % from two or more races. hispanic or latino of any race were 6.8 % of the population...		

Table 2: Example from QNLI dataset. The knowledge is filtered for brevity. For the full example and more other examples please refer to Appendix F.

domain; FinBERT (Araci, 2019) in Financial domain and PatentBERT (Lee & Hsiang, 2019) for patent classification tasks that require extensive domain knowledge.

We choose the biomedical domain and compare PlugLM with model in the architecture of BERT$_{base}$, pre-trained on the general domain (i.e. WikiBERT) and pre-trained on the biomedical domain (i.e. PubmedBERT). The statistics of datasets and pre-training details are listed in Appendix D. We test two fold ability of these PLMs. First, we test how they perform in biomed-relevant downstream tasks. Specifically, we conduct experiments on eight representative biomedical NER datasets which aim at recognizing domain-specific proper nouns in the biomedical corpus. Then we test their general language understanding ability in GLUE (Wang et al., 2018) and SQUAD (Rajpurkar et al., 2016; 2018). For SQUAD and GLUE, the DPM is from Wikipedia; while for biomedical NER, DPM is constructed from Pubmed (Canese & Weis, 2013).

The results are shown in Table 3. Both pre-trained on the Wikipedia, PlugLM outperforms WikiBERT in 8/8 NER tasks with average 1.75 F1 scores by simply switching the knowledge domain of DPM. PlugLM also gives comparable results with PubmedBERT in BC4CHEMD, JNLPBA and LINNAEUS datasets. Although PubmedBERT works well for biomedical tasks, it shows less general language understanding ability and underperforms WikiBERT and PlugLM in GLUE benchmark (Table 4) and SQUAD (Table 5), especially in low resource scenario (e.g. RTE, COLA and MRPC datasets). With DPM, PlugLM shows great flexibility and performs well in both general domain and biomedical domain. We show one concrete example from QNLI dataset in Table 2.

3.2 Knowledge Update

Since the world is not fixed as a snapshot once the pre-training corpus is collected, the current PLM, no matter how large it is, fails to adapt to this changing world. For colossal PLMs like GPT-3 and MT-NLG, efficiently fine-tuning for downstream tasks remains an open challenge (Brown et al., 2020; Smith et al., 2022), let alone re-training it on the newly coming knowledge.

In this section, we show that PlugLM can efficiently absorb new knowledge by updating the $\langle \mathbb{D}, \mathbb{K}, \mathbb{V} \rangle$ without re-training. We consider the following two settings. (1) We only pre-train PlugLM

Type	Dataset	# Annotation	WikiBERT	PlugLM	PubmedBERT
DISEASE	NCBI-disease	6811	83.65	85.96	**88.39**
	BC5CDR	12694	80.37	82.10	**83.89**
DRUG/CHEM.	BC4CHEMD	79842	87.07	**89.93**	89.35
	BC5CDR	15411	88.79	90.56	**92.75**
GENE/PROTEIN.	B2CGM	20703	80.63	82.14	**83.16**
	JNLPBA	35460	75.49	**76.39**	76.25
SPECIES	LINNAEUS	4077	85.32	**87.01**	86.11
	SPECIES-800	3708	68.54	69.73	**71.32**

Table 3: Performance of biomedical NER measured by F1 score. The PlugLM here is pre-trained on the general domain while using PubMed as DPM when fine-tuning.

	#Paras	Avg. Latency	RTE	COLA	MRPC	STS-B	SST-2	QNLI	QQP	MNLI -(m/mm)
PubmedBERT	110M	$\times$1.00	61.17	50.06	84.56	85.73	88.64	90.11	88.78	82.14/82.56
WikiBERT	110M	$\times$1.00	65.70	**53.53**	88.85	88.64	**92.32**	90.66	89.71	83.91/84.10
PlugLM	109M	$\times$2.54	**70.40**	52.68	**91.54**	**89.20**	91.86	**91.28**	**90.56**	**84.56/85.35**

Table 4: GLUE results with PubmedBERT, WikiBERT and PlugInBERT. Matched/mistached accuracies are reported to MNLI; F1 score is reported for MRPC; Spearman correlation is reported for STS-B; Matthews correlation is reported for COLA; accuracy are reported for the other tasks. Detailed latency of each model is shown in Appendix E

with limited data and gradually enlarge the DPM with unseen knowledge when fine-tuning. (2) We pre-train PlugLM with full general-domain data and ask the model to perform domain adaptation in DAR manner by gradually increasing domain knowledge in $\mathbb{D}$.

The result is shown in Figure 3. For the first setting, we choose QA (SQUAD) and Sentiment Classification tasks (SST-2) for validation. Both WikiBERT and PlugLM are pre-trained with only 1/4 Wikipedia corpus. We have the following observations: (1) PlugLM trained with limited data already outperforms WikiBERT in both tasks (0.39 EM in QA and 0.59 Accuracy in classification) which verifies the effectiveness of PlugLM in low-resource setting; (2) A consistent pattern across two tasks is that PlugLM could absorb new knowledge simply by adding more slots in $\langle \mathbb{D}, \mathbb{K}, \mathbb{V} \rangle$ without re-training.

Task	**QNLI**	**QQP**
# Training	108K	363K
Ori.	91.28	90.56
Concate.	91.28	90.12
Tagged.	91.37	90.76
Prompting.	**91.58**	**91.47**

Table 6: Performance of in-task knowledge measured by Accuracy.

For the second setting, Figure 3c also shows our model can absorb new cross-domain knowledge under adaptation setting. It achieves a higher F1 score on the LINNAEUS NER dataset with increasingly more biomed-specific knowledge injected.

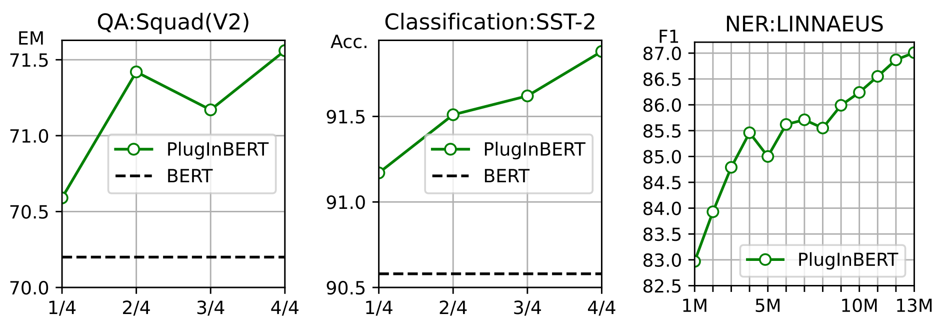
3.3 In-task knowledge

Inspired by Gururangan et al. (2020); Gu et al. (2018) and Wang et al. (2022), the training samples can also be viewed as a kind of in-task knowledge and explicit fusion of nearest training sample leads to significant gains on multiple NLG and NLU tasks.

In this section, we broaden the scope of knowledge by including the training samples in the DPM. The knowledge from Wikipedia is a textual description from domain experts (e.g., *"Machine learning (ML) is a field of inquiry devoted to understanding and building methods that 'learn', ..."*) while the training sample from a Question-answering NLI dataset is in the form of [*Question, Answer, Label*]. Considering this surface form distribution shift, we have the following injection methods.

	PubmedBERT		WikiBERT		PlugLM	
	EM	F1	EM	F1	EM	F1
SQUAD(v1)	76.68	84.56	<u>81.32</u>	<u>88.68</u>	**82.19**	**89.44**
SQUAD(v2)	68.44	71.12	<u>72.64</u>	<u>75.89</u>	**73.76**	**76.90**

Table 5: Squad results with PubmedBERT, WikiBERT and PlugLM and is measured by exact match (EM) and F1 score.



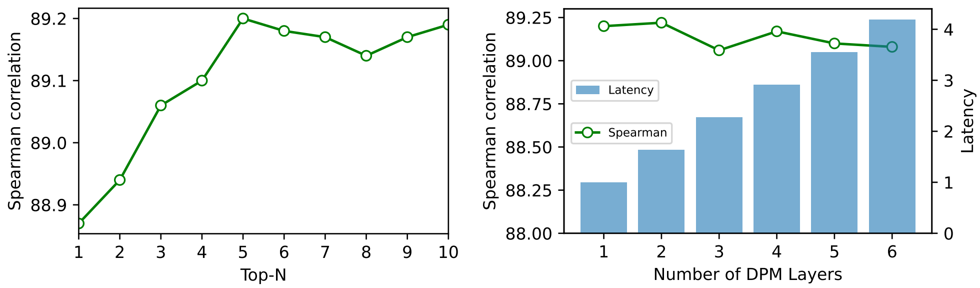
(a) QA results. (b) Classification results. (c) NER in adaptation setting.

Figure 3: Knowledge update results in different tasks and settings.

(1) Concate. We directly concatenate each training sample as a long string in the form of *"Q [SEP] A [SEP] Label"* and append this to DPM. (2) Tagged. To build the connection between model inputs and DPM, we tag each training sample by prepending a special token ([Tagged]). (3) Knowledge Prompting. Inspired by prompting method (Liu et al., 2021; Schick & Schütze, 2021), we transfer in-task knowledge to knowledge in the form of Wikipedia by a natural language prompting. For example, in QNLI dataset, we transform [*Question, Answer, Label*] with following prompting: "The first sentence (doesn't) entail(s) with the second. The first sentence is [Q] and the second is [A]". We choose moderate-sized QNLI and QQP tasks because in-task knowledge injection doesn't apply to low-resource setting in our preliminary experiments. The result is shown in Table 6. We can observe that PlugLM has managed to learn from in-task knowledge and the surface-form of knowledge actually impact the model performance. Concatenation of training sample fails to inform PlugLM the actual in-task knowledge (Zero retrieval in QNLI) and building connection between data and knowledge by a special tagged token only gives minor improvements. Instead, a well-designed knowledge prompting can actually help PlugLM learn task-specific knowledge.

3.4 TUNING PLUGLM

We investigate how key hyperparameters and architecture design affect the performance of PlugLM. (1) **Number of Retrieved Knowledge** For PlugLM, we only use the sparsely activated Top-N knowledge. Figure 4a shows the effects of different N in STS-B dataset and value of 5 proves to be optimal. (2) **Layers equipped with DPM** Considering that the upper layers in PLM capture more semantic information and are more sensitive to the input pattern (Geva et al., 2021), we equip the last encoder layer with DPM in PlugLM. Figure 4b shows that increasing DPM-enhanced encoder layer gives minor improvements but brings much latency because of extra MIPS search. (3) **FFN and DPM** To further explore the relation between FFN and DPM, we propose two model variants. First, we replace FFN in all encoder layers with a shared DPM denoted as ALL-PlugLM. Then we fuse FFN and DPM by modifying the model architecture from LayerNorm$(h +$ KnowledgeAttention$(h, K_{h'}, V_{h'}))$ to LayerNorm$(h +$ KnowledgeAttention$(h, K_{h'}, V_{h'}) + FFN(h))$ and we name it Fuse-PlugLM. Take STS-B dataset as an example (more results are shown in Appendix G), the Spearman correlation of WikiBERT, ALL-PlugLM, PlugLM and Fuse-PlugLM are 88.64, 86.82, 89.20 and 89.10. We could find that ALL-PlugLM, where there is no FFN, underperforms WikiBERT. And PlugLM performs comparably with Fuse-PlugLM. We conjecture that FFN in different layers may play different roles, which



(a) Top-N retrieved knowledge.

(b) Number of DPM-enhanced Layer.

Figure 4: Effect of number of retrieved knowledge and number of DPM-enhanced layers in STS-B.

is also reported in Geva et al. (2021). For the upper layer which captures more semantic knowledge (Geva et al., 2021; Jawahar et al., 2019), DPM is a flexible substitution of FFN, but for more shallow features, they are captured in the lower layer of Transformer. It inspires us as the future work to inject more types of knowledge, from syntactic to factual, into the DPM and design a hierarchical-structured DPM to adapt to the learning pattern of PLM.

4 Related Work

Investigating FFN Feed-forward layers constitute two-thirds of a transformer model's parameters and have been an essential component to unveil modern PLMs (Geva et al., 2021; 2022). A surge of works have investigated the knowledge captured by FFN (Dai et al., 2022a; Meng et al., 2022; Geva et al., 2021; 2022; Jiang et al., 2020; Yao et al., 2022; Wallat et al., 2020). Based on the view that FFN is essentially an unnormalized key-value memory network, Dai et al. (2022a) detects knowledge neurons in FFN and edit specific factual knowledge without fine-tuning. Meng et al. (2022) modifies FFN weights to update specific factual associations using Rank-One Model Editing. Yao et al. (2022) injects knowledge into the FFN via BM25. Dai et al. (2022b) and (Lample et al., 2019) enhance the model by expanding the size of FFN with extra trainable keys and values. One main difference of our model is that the DPM is grounded: each key-value pair is associated with one concrete knowledge rather than unnamed vectors.

Knowledge-Augmented Language Model There are two lines of works to equip PLM with knowledge. The first is introduce additional Knowledge Graph (KG) and knowledge-based training signal (e.g., entity linking) into the language model pre-training, like ERNIE (Zhang et al., 2019; Sun et al., 2019), KnowBERT (Peters et al., 2019) and KEPLER (Wang et al., 2021). Another line of works adopt retrieval mechanism to incorporate knowledge, either symbolic (Verga et al., 2021; Agarwal et al., 2021; Févry et al., 2020) or texual (Guu et al., 2020; Lewis et al., 2020b; Borgeaud et al., 2022; Lewis et al., 2020a; Verga et al., 2021; de Jong et al., 2021). They formulate the task as retrieve then predict process by using extra neural dense retriever (BERT) or sparse retriever (BM25) to find most relevant supporting knowledge and combine it with input using either concatenation (Guu et al., 2020) or attention methods (de Jong et al., 2021; Wu et al., 2021; Févry et al., 2020; Chen et al., 2022). One distinct difference of our work is that we do not try to equip the model with additional knowledge to perform knowledge-intensive tasks, but we managed to decouple the knowledge that would otherwise be stored in the parameters with an editable and scalable DPM and leverage knowledge in an explainable manner.

5 Conclusion

In this paper, we propose a novel knowledge encoding mechanism for PLMs, which decouples the learned knowledge during pre-training from the FFN parameters of the PLMs. This enables the knowledge encoding of PLM more flexible and interpretable. Extensive results verify the flexibility and scalability of our PlugLM in various settings including domain adaptation, knowledge updating and in-task knowledge learning. Future work would involve an efficient PlugLM for practical usage.

REFERENCES

Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3554–3565, 2021.

Dogu Araci. Finbert: Financial sentiment analysis with pre-trained language models. *arXiv preprint arXiv:1908.10063*, 2019.

Iz Beltagy, Kyle Lo, and Arman Cohan. Scibert: A pretrained language model for scientific text. 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)*, pp. 3615–3620, 2019.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. In *International Conference on Machine Learning*, pp. 2206–2240. PMLR, 2022.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Deng Cai, Yan Wang, Huayang Li, Wai Lam, and Lemao Liu. Neural machine translation with monolingual translation memory. 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)*, pp. 7307–7318, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.567. URL `https://aclanthology.org/2021.acl-long.567`.

Kathi Canese and Sarah Weis. Pubmed: the bibliographic database. *The NCBI handbook*, 2(1), 2013.

Wenhu Chen, Pat Verga, Michiel de Jong, John Wieting, and William Cohen. Augmenting pre-trained language models with qa-memory for open-domain question answering. *arXiv preprint arXiv:2204.04581*, 2022.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8493–8502, 2022a.

Damai Dai, Wenbin Jiang, Qingxiu Dong, Yajuan Lyu, Qiaoqiao She, and Zhifang Sui. Neural knowledge bank for pretrained transformers. *arXiv preprint arXiv:2208.00399*, 2022b.

Michiel de Jong, Yury Zemlyanskiy, Nicholas FitzGerald, Fei Sha, and William W Cohen. Mention memory: incorporating textual knowledge into transformers through entity mention attention. In *International Conference on Learning Representations*, 2021.

Franck Dernoncourt and Ji Young Lee. Pubmed 200k rct: a dataset for sequential sentence classification in medical abstracts. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pp. 308–313, 2017.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://aclanthology.org/N19-1423`.

Mihail Eric, Lakshmi Krishnan, Francois Charette, and Christopher D Manning. Key-value retrieval networks for task-oriented dialogue. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pp. 37–49, 2017.

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

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5484–5495, 2021.

Mor Geva, Avi Caciularu, Kevin Ro Wang, and Yoav Goldberg. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. *arXiv preprint arXiv:2203.14680*, 2022.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor OK Li. Search engine guided neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Suchin Gururangan, Tam Dang, Dallas Card, and Noah A Smith. Variational pretraining for semi-supervised text classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5880–5894, 2019.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8342–8360, 2020.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In *International Conference on Machine Learning*, pp. 3929–3938. PMLR, 2020.

Ruining He and Julian McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*, pp. 507–517, 2016.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Few-shot learning with retrieval augmented language models. *arXiv preprint arXiv:2208.03299*, 2022.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*, 2019.

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

David Jurgens, Srijan Kumar, Raine Hoover, Daniel A. McFarland, and Dan Jurafsky. Measuring the evolution of a scientific field through citation frames. *TACL*, 2018. URL `https://www.aclweb.org/anthology/Q18-1028/`.

Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. SemEval-2019 Task 4: Hyperpartisan news detection. In *SemEval*, 2019. URL `https://www.aclweb.org/anthology/S19-2145/`.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Jens Kringelum, Sonny Kim Kjærulff, Søren Brunak, Ole Lund, Tudor I. Oprea, and Olivier Taboureau. ChemProt-3.0: a global chemical biology diseases mapping. In *Database*, 2016. URL `https://www.ncbi.nlm.nih.gov/pubmed/26876982`.

Guillaume Lample, Alexandre Sablayrolles, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Large memory layers with product keys. *Advances in Neural Information Processing Systems*, 32, 2019.

Jieh-Sheng Lee and Jieh Hsiang. Patentbert: Patent classification with fine-tuning a pre-trained bert model. *arXiv preprint arXiv:1906.02124*, 2019.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 2019. URL https://arxiv.org/abs/1901.08746.

Mike Lewis, Marjan Ghazvininejad, Gargi Ghosh, Armen Aghajanyan, Sida Wang, and Luke Zettlemoyer. Pre-training via paraphrasing. *Advances in Neural Information Processing Systems*, 33: 18470–18481, 2020a.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33: 9459–9474, 2020b.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*, 2021.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel S Weld. S2orc: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4969–4983, 2020.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3219–3232, 2018.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pp. 142–150, 2011.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pp. 43–52, 2015.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *arXiv preprint arXiv:2202.05262*, 2022.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1400–1409, 2016.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL https://aclanthology.org/N18-1202.

Matthew E Peters, Mark Neumann, Robert L Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. Knowledge enhanced contextual word representations. *arXiv preprint arXiv:1909.04164*, 2019.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 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)*, pp. 2463–2473, 2019.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10): 1872–1897, 2020.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018. URL https://www.semanticscholar.org/paper/Improving-Language-Understanding-by-Generative-Radford/cd18800a0fe0b668a1cc19f2ec95b5003d0a5035.

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

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 784–789, 2018.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2020.

Mohammad Rostami. Lifelong domain adaptation via consolidated internal distribution. *Advances in Neural Information Processing Systems*, 34:11172–11183, 2021.

Timo Schick and Hinrich Schütze. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pp. 255–269, 2021.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*, 2022.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. *Advances in neural information processing systems*, 28, 2015.

Sainbayar Sukhbaatar, Edouard Grave, Guillaume Lample, Herve Jegou, and Armand Joulin. Augmenting self-attention with persistent memory. *arXiv preprint arXiv:1907.01470*, 2019.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Pat Verga, Haitian Sun, Livio Baldini Soares, and William Cohen. Adaptable and interpretable neural memory over symbolic knowledge. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3678–3691, 2021.

Jonas Wallat, Jaspreet Singh, and Avishek Anand. Bertnesia: Investigating the capture and forgetting of knowledge in bert. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pp. 174–183, 2020.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353–355, 2018.

Shuohang Wang, Yichong Xu, Yuwei Fang, Yang Liu, Siqi Sun, Ruochen Xu, Chenguang Zhu, and Michael Zeng. Training data is more valuable than you think: A simple and effective method by retrieving from training data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3170–3179, 2022.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pp. 9929–9939. PMLR, 2020.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194, 2021.

Alex Warstadt, Yu Cao, Ioana Grosu, Wei Peng, Hagen Blix, Yining Nie, Anna Alsop, Shikha Bordia, Haokun Liu, Alicia Parrish, et al. Investigating bert's knowledge of language: Five analysis methods with npis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 2877–2887, 2019.

Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

Taesun Whang, Dongyub Lee, Chanhee Lee, Kisu Yang, Dongsuk Oh, and Heuiseok Lim. An effective domain adaptive post-training method for bert in response selection. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, volume 2020, pp. 1585–1589, 2020.

Yuhuai Wu, Markus Norman Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers. In *International Conference on Learning Representations*, 2021.

Yichong Xu, Chenguang Zhu, Ruochen Xu, Yang Liu, Michael Zeng, and Xuedong Huang. Fusing context into knowledge graph for commonsense question answering. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 1201–1207, 2021.

Yunzhi Yao, Shaohan Huang, Li Dong, Furu Wei, Huajun Chen, and Ningyu Zhang. Kformer: Knowledge injection in transformer feed-forward layers. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pp. 131–143. Springer, 2022.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. Defending against neural fake news. *Advances in neural information processing systems*, 32, 2019.

Rong Zhang, Revanth Gangi Reddy, Md Arafat Sultan, Vittorio Castelli, Anthony Ferritto, Radu Florian, Efsun Sarioglu Kayi, Salim Roukos, Avi Sil, and Todd Ward. Multi-stage pre-training for low-resource domain adaptation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 5461–5468, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.440. URL `https://aclanthology.org/2020.emnlp-main.440`.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

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

A ATTENTIVE POOLING

Listing 1: Attentive Pooling

```
import torch
import torch.nn as nn

class AttentivePooler(nn.Module):
    def __init__(self,d_model):
        super().__init__()
        self.att_fc1 = nn.Linear(d_model,d_model)
        self.att_fc2 = nn.Linear(d_model,1)
    def forward(self,x,attn_mask = None):
        bz = x.shape[0]
        e = self.att_fc1(x)
        e = nn.Tanh()(e)
        alpha = self.att_fc2(e)
        alpha = torch.exp(alpha)
        if attn_mask is not None:
            alpha = alpha * attn_mask.unsqueeze(2)
        alpha = alpha / (torch.sum(alpha, dim=1, keepdim=True)
                        + 1e-8)
        x = torch.bmm(x.permute(0, 2, 1), alpha)
        x = torch.reshape(x, (bz, -1))
        return x
```

B PLUGLM PRETRAINING DETAILS

Hyperparameter	**Assignment**
vocab size	30522
num layers with DPM	top-1
top-N	5
number of layers	12
attention head	12
mlm masking	static
mlm masking rate	0.15
ffn size	3072
max knowledge length	288
Uncased	True
memory size	14802866
batch size	64
gradient accumulation steps	128
max train steps	8000
optimizer	FusedLAMBAMP
learning rate	1e-4
index refreshing step	200
learning rate scheduler	PolyWarmUpScheduler
Warmup proportion	0.2843
weight decay	0.01

Table 7: Hyperparameters for PlugLM pretraining.

C Data for Domain Adaptive Post-Training

The detailed statistics of domain corpora for post-training is listed in the Table 8 and downstream tasks in Table 9.

Domain	Pretraining Corpus	# Tokens	Size
BioMed	1.24M papers from S2ORC (Lo et al., 2020)	2.67B	12GB
CS	5.07M papers from S2ORC (Lo et al., 2020)	4.3B	18GB
News	11.90M articles from RealNews Zellers et al. (2019)	6.66B	39GB
Reviews	24.75M Amazon reviews (He & McAuley, 2016)	2.11B	11GB

Table 8: List of the domain-specific unlabeled datasets.

Domain	Task	Label Type	Train (Lab.)	Dev.	Test	Classes
BioMed	ChemProt	relation classification	4169	2427	3469	13
	†RCT	abstract sent. roles	18040	30212	30135	5
CS	ACL-ARC	citation intent	1688	114	139	6
	SciERC	relation classification	3219	455	974	7
News	HyperPartisan	partisanship	515	65	65	2
	†AGNews	topic	115000	5000	7600	4
Reviews	†Helpfulness	review helpfulness	115251	5000	25000	2
	†IMDB	review sentiment	20000	5000	25000	2

Table 9: Specifications of the various target task datasets. † indicates high-resource settings. Sources: ChemProt Kringelum et al. (2016), RCT Dernoncourt & Lee (2017), ACL-ARC Jurgens et al. (2018), SciERC Luan et al. (2018), HyperPartisan Kiesel et al. (2019), AGNews Zhang et al. (2015), Helpfulness McAuley et al. (2015), IMDB Maas et al. (2011).

D Details for Wikipedia and Pubmed

Dataset	Domain	Source	Size
Wikipedia	General	`https://dumps.wikimedia.org`	14.35GB
PubMed	Biomedical	`https://github.com/naver/biobert-pretrained`	28.12GB

Table 10: List of the PubMed and Wikipedia.

Hyperparameter	Assignment
vocab size	30522
Uncased	True
number of Layers	12
attention Head	12
ffn Size	3072
mlm masking	static
batch size	64
gradient accumulation steps	128
max train steps	8000
optimizer	FusedLAMBAMP
learning rate	6e-3
index refreshing step	200
learning rate scheduler	PolyWarmUpScheduler
Warmup proportion	0.2843
weight decay	0.01

Table 11: Hyperparameters for WikiBERT and PubmedBERT pretraining.

E LATENCY

	RTE	COLA	MRPC	STS-B	SST-2	QNLI	QQP	MNLI-(m/mm)
Size	0.27K	1.04K	0.41K	1.5K	0.87K	5.47K	40.43K	9.82K/9.83K
WikiBERT	1.01	1.98	1.33	2.43	1.75	7.01	52.32	15.03/15.02
PlugLM	1.73	4.41	2.22	5.94	3.86	20.01	141.15	34.60/34.58

Table 12: Testing Latency of WikiBERT and PlugLM measured by seconds. All experiments are computed in the same computational device with same batch size. The CPU is AMD EPYC 7K62 48-Core Processor. GPU is A100-SXM4. Driver Version is 450.156.00. CUDA Version is 11.1.

F Case Study

Question	Answer	Prediction	Label
How much of Jacksonville is made up of water?	According to the United States Census Bureau, the city has a total area of 874.3 square miles (2,264 km^2), making Jacksonville the largest city in land area in the contiguous United States; of this, 86.66% (757.7 sq mi or 1,962 km^2) is land and ; 13.34% (116.7 sq mi or 302 km^2) is water.	Entailment	Entailment
Knowledge	(1) this article lists the 3, 143 states of america. the 50 states of the united states are divided into 3, 007 " counties ", political and geographic subdivisions of a state ; 236 other local governments and geographic places are also first - order administrative divisions of their respective state / district / territory, but are called by different names. the latter are referred to collectively as " county equivalents " by the united states census bureau. the 236 county equivalents include 100 equivalents in the territories (such as those in puerto rico) outside the 50 states and the district of columbia. the large majority of counties and equivalents were organized by 1970. since that time, most creations, boundary changes and dissolutions have occurred in alaska and virginia. among the 50 states, 44 are partitioned entirely into counties, with no county equivalents. louisiana is instead divided into 64 equivalent parishes. (2) the united states census bureau (usc ##b) , officially the bureau of the census , is a principal agency of the u . s . federal statistical system , responsible for producing data about the american people and economy . the census bureau is part of the u . s . department of commerce and its director is appointed by the president of the united states . the census bureau ' s primary mission is conducting the u . s . census every ten years , which all ##oca ##tes the seats of the u . s . house of representatives to the states based on their population . [1] the bureau ' s various census ##es and surveys help all ##oca ##te over $ 67 ##5 billion in federal funds every year and it assists states , local communities , and businesses make informed decisions . [2] [3] [4] the information provided by the census informs decisions on where to build and maintain schools , hospitals , transportation infrastructure , and police and fire departments (3) the crestview – fort walton beach – destin, florida, metropolitan statistical area, as defined by the united states census bureau, is a metropolitan area consisting of two counties in northwest florida, anchored by the cities of crestview, florida, and fort walton beach, florida. as of the 2010 census, the msa had a population of 235, 865, and a 2012 population estimate of 247, 665. the metropolitan area is a part of the " northwest corridor " which includes the pensacola metropolitan area and the panama city metropolitan area. demographics. as of the census of 2010, there were 235, 865 people, 95, 892 households, and 63, 964 families residing within the msa. the racial makeup of the msa was 81. 1 % white, 9. 3 % african american, 0. 3 % native american, 2. 9 % asian, 0. 1 % pacific islander, 0. 2 % from other races, and 3. 9 % from two or more races. hispanic or latino of any race were 6. 8 % of the population. according to the 2010 american community survey 1 - year (4) analog to digital conversions were achieved through steinberg, and in some cases mytek, converters. the album was recorded and mixed exclusively with steinberg cubase digital audio workstations on microsoft windows operating systems with waves ssl and abbey road tg12413 plugins. it was revealed that neither brahm nor marc know how to operate autotune, so it was not used. the songs were often performed to a click track, but there was no " snapping the drums to a grid ", which is a popular computerized technique to ensure that drums are in perfect time while simultaneously sucking the life out of an otherwise real performance. production. " tears of the enchanted mainframe " was produced and engineered by taylor and kaducak. backmasking is used on the track " superusurper " during an interlude that features a reversed reading of a passage from the george orwell novel nineteen eighty four. the album was mastered by geoff pesche and alex wharton at abbey road studios in london. title and artwork. " tears of the enchanted mainframe " (5) the zafarnama (, lit. " book of victory ") is a biography of timur written by the historian nizam ad - din shami. it served as the basis for a later and better - known " zafarnama " by sharaf ad - din ali yazdi. one translation by felix tauer was published in prague in 1937.		

Table 13: Example from QNLI dataset.

Input		Prediction	Label
Various approaches for computing semantic relatedness of words or concepts have been proposed , e.g. dictionary-based (Lesk , 1986) , ontology-based (Wu and Palmer , 1994 ; Leacock and Chodorow , 1998) , information-based (Resnik , 1995 ; Jiang and Conrath , 1997) or distributional (Weeds and Weir , 2005).		Background	Background
Knowledge	(1) instrumentation and control engineering (ice) is a branch of engineering that studies the measurement and control of process variables, and the design and implementation of systems that incorporate them. process variables include pressure, temperature, humidity, flow, ph, force and speed. ice combines two branches of engineering. instrumentation engineering is the science of the measurement and control of process variables within a production or manufacturing area. meanwhile, control engineering, also called control systems engineering, is the engineering discipline that applies control theory to design systems with desired behaviors. control engineers are responsible for the research, design, and development of control devices and systems, typically in manufacturing facilities and process plants. control methods employ sensors to measure the output variable of the device and provide feedback to the controller so that it can make corrections toward desired performance. automatic control manages a device without the need of human inputs for correction, such as cruise control for regulating a car's speed. control systems engineering activities are multi - disciplinary in nature. they focus on the implementation of control systems, mainly derived by mathematical modeling. because instrumentation and control play a significant role in gathering information from a system and changing its parameters, they are a key part of control loops. as profession. high demand for engineering professionals is found in fields associated with process automation. specializations include industrial instrumentation, system dynamics, process control, and control systems. additionally, technological knowledge, particularly in computer systems, is essential to the job of (2) instrumentation is the art and science of measurement and control. instrumentation may also refer to: (3) the scientific and technological innovation ability of colleges and universities, and strengthening the evaluation research of the scientific and technological innovation ability and efficiency of colleges and universities, can we better promote the scientific and technological innovation ability of colleges and universities. universities the evaluation of scientific and technological innovation ability in colleges and universities is a complex system engineering, and the understanding of its connotation is the most important problem to be considered in the comprehensive evaluation. by consulting the data, it is found that the previous researches are mainly focused on the following three aspects : 1. from the perspective of innovative resource demand and innovative achievements, the scientific and technological innovation in colleges and universities is regarded as an organic whole composed of various elements. in the whole innovation system, colleges and universities undertake the functions and tasks of knowledge production and dissemination, technological innovation and transformation as well as personnel training. according to the relationship between innovation elements, the scientific and technological innovation ability of colleges and universities is divided into basic strength of scientific and technological innovation, scientific and technological innovation input ability, knowledge innovation ability, technological innovation ability, scientific and technological innovation output ability. science and technology innovation achievement transformation ability, talent innovation ability. 2. from the perspective of innovation process, the ability of scientific and technological innovation in colleges and universities is embodied in the process of knowledge creation, knowledge dissemination, transformation and diffusion of technological inventions. it also includes the technological, economic and managerial abilities that the university relies on (4) automation engineering has two different meanings : automation engineer. automation engineers are experts who have the knowledge and ability to design, create, develop and manage machines and systems, for example, factory automation, process automation and (5) this learning methodology is called blended learning. blended learning can also incorporate machine learning and other such technologies to implement adaptive learning.		

Table 14: Example from ACL-ARC dataset.

Input		Prediction	Label
Although there are other discussions of the paragraph as a central element of discourse (e.g. Chafe 1979 , Halliday and Hasan 1976 , Longacre 1979 , Haberlandt et al. 1980) , all of them share a certain limitation in their formal techniques for analyzing paragraph structure .		CompareOrContrast	CompareOrContrast
Knowledge	(1) automation engineering has two different meanings : automation engineer. automation engineers are experts who have the knowledge and ability to design, create, develop and manage machines and systems, for example, factory automation, process automation and warehouse automation. scope. automation engineering is the integration of standard engineering fields. automatic control of various control system for operating various systems or machines to reduce human efforts & amp ; time to increase accuracy. automation engineers design and service electromechanical devices and systems to high - speed robotics and programmable logic controllers (plcs). work and career after graduation. graduates can work for both government and private sector entities such as industrial production, companies that create and use automation systems, for example paper industry, automotive industry, food and agricultural industry, water treatment, and oil & amp ; gas sector such as refineries, power plants. job description. automation engineers can design, program, simulate and test automated machinery and processes, and usually are employed in industries such as the energy sector in plants, car manufacturing facilities or food processing plants and robots. automation engineers are responsible for creating detailed design specifications and other documents, developing automation based on specific requirements for the process involved, and conforming to international standards like iec - 61508, local standards, and other process specific guidelines and specifications, simulate, test and commission electronic equipment for automation. (2) abstract. manipulator is a powerful tool which can help people to carry out the safe operation, production automation and improve the productivity of labor. based on the summary of the situation of research and development of manipulator, this article analyzes the functions of parts moving manipulator and carries out mechatronic design of parts moving manipulator according to the practical project items of parts moving manipulator of enterprises. on the basis of the analysis of the performance requirement and the operating characteristics of parts moving manipulator, this article analyses and designs the whole schemes for the mechanical structure, driving system, driving mode and the software and hardware control system of manipulator, and in which, the form of mechanical structure of cylindrical coordinate system is determined to be adopted in the design of manipulator, the driving scheme of pneumatic transmission is adopted, and the system control is carried out by plc. on this basis, this article analyses the kinematics and dynamics of parts moving manipulator and summarizes the relationship between displacement, speed, acceleration and joint angle. with the progress of science and technology and the development of social economy, the application area of manipulator has been becoming wider and wide. the manipulator can be found everywhere in human society. the application of manipulator has been extended to the civilian application fields such (3) in working environments with large manipulators, accidental collisions can cause severe personal injuries and can seriously damage manipulators, necessitating the development of an emergency stop algorithm to prevent such occurrences. in this paper, we propose an emergency stop system for the efficient and safe operation of a manipulator by applying an intelligent emergency stop algorithm. our proposed intelligent algorithm considers the direction of motion of the manipulator. in addition, using a new regression method, the algorithm includes a decision step that determines whether a detected object is a collision - causing obstacle or a part of the manipulator. we apply our emergency stop system to a two - link manipulator and assess the performance of our intelligent emergency stop algorithm as compared with other models. increasing the safety of robots, especially industrial manipulators, is just as important as improving their performance. a collision between a manipulator and a person, for example, may cause severe personal injury as well as damage to the machinery. thus, it is necessary to develop an algorithm that can detect collisions before they occur and make the manipulator stop before damage is done. various emergency stop or obstacle avoidance algorithms for robots, particularly those utilizing distance - measuring sensors [1] [2] [3] [4] or vision sensors have been reported [5] [6] [7] [8] and those algorithms using each (4) the reliability of kinematic trajectory of manipulators describes the ability that manipulators keep kinematic accurate. it is an important parameter to evaluate the performance of manipulators. the kinematic accuracy of manipulators can be improved when piezoelectricity material are used as a transducer to suppress the vibration of flexible manipulators. first, a 3 degree - of - freedom parallel manipulator system and its dynamic equations are introduced. the theory and experiment of a vibration suppression system are then presented. the calculation method of both error and reliability of kinematic trajectory of manipulator is further implemented. finally, the reliability of kinematic accuracy are calculated and analyzed for the 3 degree - of - freedom parallel manipulator with or without vibration suppressing control. the results show that the reliability of kinematic accuracy is improved using vibration suppressing control. the reliability of kinematic accuracy of manipulators is an important indicator to evaluate the accuracy of manipulator motion [1]. in manipulators, light weight linkages are employed to achieve high speed and acceleration motions for better performance. however, the light weight linkage will result in inherent structural vibration, and the structural vibration leads to inaccurate kinematic trajectory of manipulators. different methods have been proposed to reduce the vibration of the flexible link (5) abstract - economic dispatch and frequency regulation are typically viewed as fundamentally different problems in power systems and, hence, are typically studied separately. in this paper, we frame and study a joint problem that co - optimizes both slow timescale economic dispatch resources and fast timescale frequency regulation resources. we show how the joint problem can be decomposed without loss of optimality into slow and fast timescale subproblems that have appealing interpretations as the economic dispatch and frequency regulation problems, respectively. we solve the fast timescale subproblem using a distributed frequency control algorithm that preserves network stability during transients. we solve the slow timescale subproblem using an efficient market mechanism that coordinates with the fast timescale subproblem. we investigate the performance of our approach on the ieee 24 - bus reliability test system. abstract - economic dispatch and frequency regulation are typically viewed as fundamentally different problems in power systems and, hence, are typically studied separately. in this paper, we frame and study a joint problem that co - optimizes both slow timescale economic dispatch resources and fast timescale frequency regulation resources. we show how the joint problem can be decomposed without loss of optimality into slow and fast timescale subproblems that have appealing interpretations as the economic dispatch and frequency regulation problems, respectively. we solve the fast timescale subproblem		

Table 15: Example from ACL-ARC dataset.

G More Experiments for Tuning PlugLM

	WikiBERT	**ALL-PlugLM**	**Fuse-PlugLM**	**PlugLM**
STS-B	88.64	86.82	89.20	89.10
MRPC	88.85	87.42	91.27	91.54
QNLI	90.66	88.19	91.36	91.28

Table 16: Experimental Results as in Section 3.4 on `STS-b`, `MRPC` and `QNLI`. The evaluation metrics are Spearman correlation, F1 score and Accuracy respectively.