Decouple knowledge from paramters for plug-and-play language modeling

Xin Cheng [1], **Yankai Lin** [2,6], **Xiuying Chen** [3], **Dongyan Zhao** [1,4,5]*, **Rui Yan** [2,6]*

[1] Wangxuan Institute of Computer Technology, Peking University

[2] Gaoling School of Artificial Intelligence, Renmin University of China

[3] Computational Bioscience Reseach Center, KAUST [4] BIGAI, Beijing, China

[5] National Key Laboratory of General Artificial Intelligence

[6] Engineering Research Center of Next-Generation Intelligent Search and Recommendation, Ministry of Education

chengxin1998@stu.pku.edu.cn yankailin@ruc.edu.cn

xiuying.chen@kaust.edu.sa zhaody@pku.edu.cn

ruiyan@ruc.edu.cn

Abstract

Pre-trained language models (PLM) have made impressive results in various NLP tasks. It has been revealed that one of the key factors to their success is the parameters of these models implicitly learn all kinds of knowledge during pre-training. 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 humans from understanding which knowledge PLM requires for a certain problem. In this paper, we introduce PlugLM, a pre-training model with differentiable plug-in memory (DPM). The key intuition 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. To justify this design choice, we conduct evaluations in three settings including: (1) domain adaptation. PlugLM obtains 3.95 F1 improvements across four domains on average without any in-domain pre-training. (2) knowledge update. PlugLM could absorb new knowledge in a training-free way after pre-training is done. (3) in-task knowledge learning. PlugLM could be further improved by incorporating training samples into DPM with knowledge prompting[1].

1 Introduction

Large 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 in multiple downstream tasks (Rogers et al., 2020; Izacard et al., 2022). As Geva et al. (2021) pointed out, the feed-forward layers (FFN), constituting two-thirds of a transformer model's parameters, are essentially key-value memories and store all kinds of knowledge of PLM. The first linear layer of FFN acts like a set of sparsely activated keys detecting input patterns while the second is the corresponding value. 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 upper bound (Ouyang et al., 2022).

However, a fundamental question still remains: **For PLM, is it the optimal way to implicitly encode knowledge in its parameters?** 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 (e.g., BERT doesn't know what is a BERT). Nevertheless, 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 retrained for the newly coming one. Second, the current PLMs lack interpretability at the knowledge level. Implicit knowledge encoding fails to provide provenance for model's prediction and makes PLM a black box preventing humans from understand-

* Corresponding author.

[1] Code available at https://github.com/Hannibal046/PlugLM

Findings of the Association for Computational Linguistics: ACL 2023, pages 14288–14308

ing which knowledge PLM requires for a certain problem.

In this work, we propose a novel architecture of PLM, PlugLM, which decouples the knowledge storage from model parameters 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 as well as editable. Each slot of DPM encodes the knowledge to a pair of key and value, and thus we can explicitly retrieve the required knowledge in natural language from DPM rather than unnamed vectors in FFN.

To justify the design choice of decoupling the knowledge from parameters, we conduct extensive evaluations under different settings. In the domain adaptation setting, PlugLM could be easily adapted to different domains with pluggable in-domain memory—obtaining 3.95 F1 improvements across four domains on average and up to 11.55 F1 improvement on ACL-ARC citation intent classification dataset, without any in-domain pre-training. In the knowledge update setting, PlugLM could absorb new knowledge after pre-training is done in a training-free way by knowledge updating operation in the DPM, with an improvement up to 4 F1 scores in LINNAEUS NER dataset. PlugLM could further be improved by incorporating training samples into DPM with knowledge prompting as a kind of in-task knowledge.

2 Related Work

Investigating FFN Feed-forward layers constitute two-thirds of a transformer model's parameters and are essential 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., 2021). 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.

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., 2020; Agarwal et al., 2021; Févry et al., 2020) or texual (Guu et al., 2020; Lewis et al., 2020c; Borgeaud et al., 2022; Lewis et al., 2020a; Verga et al., 2020; de Jong et al., 2022). They formulate the task as retrieve then predict process by using extra neural dense retriever or sparse retriever to find most relevant supporting knowledge and combine it with input using either concatenation (Guu et al., 2020; Lewis et al., 2020c), attention methods (de Jong et al., 2022; Chen et al., 2022) or interpolation (Khandelwal et al., 2020; Zhong et al., 2022)

PlugLM differs from previous works in that we do not try to equip the model with additional knowledge to perform knowledge-intensive tasks. The key insight is to transform FFN architecture into deep retrieval in the interest of decoupling the knowledge which would otherwise be stored in the parameters and this is orthogonal to all retrieval-augmented PLMs.

3 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 (FFN) layers. The former captures the contextual interaction among inputs and the latter process each input independently. Let $x \in \mathbb{R}^{d_1}$ be a vector as input, the FFN could be formulated as:

$$\text{FFN}(x) = \sigma(x \cdot \mathbf{W_1}^\top) \cdot \mathbf{W_2} \qquad (1)$$

where $\mathbf{W_1}, \mathbf{W_2} \in \mathbb{R}^{d_2 \times d_1}$ and σ is the activation function. The bias term is omitted for brevity.

Key-Value Memory Network The Key-Value Memory Network (Weston et al., 2014; Sukhbaatar et al., 2015) corresponds to d_2 key-value pairs and each key/value is a vector in $\mathbb{R}^{d_1}$. They are the generalization of the way knowledge is stored (Eric et al., 2017; Miller et al., 2016). For an input $x \in \mathbb{R}^{d_1}$, there are two stages for a key-value memory

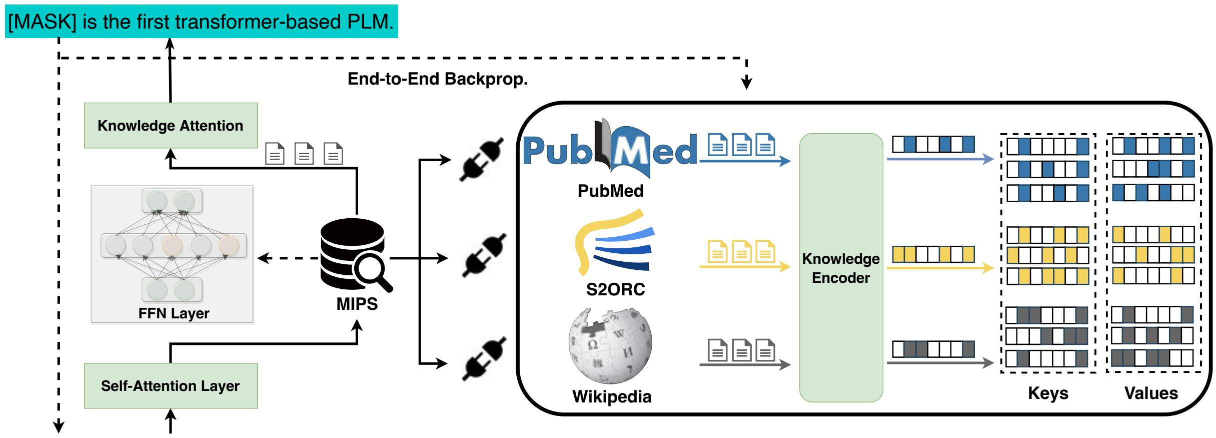


Figure 1: Overview of our PlugLM. We replace FFN in PLM with a Differentiable Plug-in key-value Memory (DPM) by which PLM could store and leverage knowledge in an explainable manner.

network. First, the lookup (addressing) stage would compute the matching degree between x and each key. In the second stage, x would be transformed by the weighted sum of values according to the distribution of the matching degree in the first stage. We can formally define it as:

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

where $\mathbf{K}, \mathbf{V} \in \mathbb{R}^{d_2 \times d_1}$. 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).

4 PlugLM

The overall architecture of PlugLM is illustrated in Figure 1. Because FFN is essentially a key-value memory network (Geva et al., 2021; Dai et al., 2022a; Meng et al., 2022), PlugLM creatively decouples the knowledge storage from model parameters by replacing[2] FFN with a Differential Plug-in key-value Memory, DPM (§4.1) and conducting knowledge retrieval in DPM with knowledge attention (§4.2) for explicit knowledge usage instead of storing all knowledge implicitly in the model parameters. In §4.3, we detailedly explain how PlugLM is trained in both pre-training and fine-tuning stages.

[2]Because different layers in transformer capture different knowledge, the lower layer for shallow patterns while the upper layers for more semantic ones (Geva et al., 2021; ?), we only consider replacing FFN in Top-L layers with DPM while keeping FFN in the lower layers untouched to encode the intrinsic language understanding knowledge as detailed in §5.4.

4.1 Differential Plug-in Memory

In this paper, we view n-th knowledge $d_n = \{t_n^1, t_n^2, ..., t_n^{|d_n|}\}$ as consecutive tokens from unlabeled corpora as in Guu et al. (2020). For each d_n, we get its dense representation h_n from a knowledge encoder KnowEncoder$(\cdot)$:

$$h_n = \text{AttnPooling}(\text{E}_{\text{Token}}(d_n) + \text{E}_{\text{Pos}}(d_n)) \quad (3)$$

where AttentivePooling function (Xu et al., 2021; Cheng et al., 2023a) corresponds to a trainable pattern detector aggregating information from a sequence of input. And E_{Token} and E_{Pos} denote token embedding and positional embedding. Then we use two independent mapping functions to project h_n to the key space and value space:

$$k_n = \mathbf{W_k} \cdot h_n + \mathbf{b_k} \quad (4)$$

$$v_n = \mathbf{W_v} \cdot h_n + \mathbf{v_k} \quad (5)$$

where $\mathbf{W_k}$, $\mathbf{W_v}$, $\mathbf{b_k}$ and $\mathbf{v_k}$ are trainable parameters. And DPM is a triplet of $\langle \mathbb{D}, \mathbb{K}, \mathbb{V} \rangle$:

$$\mathbb{D} = \{d_1, d_2, ..., d_{|\mathbb{D}|}\} \quad (6)$$

$$\mathbb{K} = \{k_1, k_2, ..., k_{|\mathbb{D}|}\} \quad (7)$$

$$\mathbb{V} = \{v_1, v_2, ..., v_{|\mathbb{D}|}\} \quad (8)$$

4.2 Memory Fusion

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 interacting with unnamed vectors in FFN, we conduct Maximum Inner Product Search (MIPS) to retrieve knowledge in natural language from $\langle \mathbb{D}, \mathbb{K}, \mathbb{V} \rangle$ where each triplet corresponds to one knowledge along with its key and value representation. For h, we first get its sentence-level representation z by an attentive pooling function $z = \text{AttentivePooling}(h)$, then we use z as the query vector to $\langle \mathbb{D}, \mathbb{K}, \mathbb{V} \rangle$. Since PLM is internally sparse (Li et al., 2022), we only consider Top-N knowledge $\mathbb{D}_z$ with corresponding keys $\mathbb{K}_z$ and values $\mathbb{V}_z$:

$$\mathbb{K}_z = \text{Top-N}(\text{MIPS}(z, \mathbb{K})) \quad (9)$$

$$\mathbb{V}_z = \{v_i \text{ if } k_i \text{ in } \mathbb{K}_z\} \quad (10)$$

$$\mathbb{D}_z = \{d_i \text{ if } k_i \text{ in } \mathbb{K}_z\} \quad (11)$$

where Top-N also corresponds to the indexing operation. With $\mathbb{K}_z$ and $\mathbb{V}_z$, we use knowledge attention to fuse retrieved knowledge into our model:

$$\text{Attention}(h, \mathbb{K}_z, \mathbb{V}_z) = \text{softmax}(\frac{h\mathbb{K}_z^\top}{\sqrt{d}})\mathbb{V}_z \quad (12)$$

where d is the head dimension. By knowledge retrieval and fusion, we explore an interpretable way to incorporate knowledge into the model where $\mathbb{D}_z$ is the actual knowledge that PLM would leverage. And direct modification on $\mathbb{D}$ without changing model parameters empowers PlugLM with much flexibility and scalability in domain adaptation (§5.1) and knowledge update (§5.2) scenarios.

4.3 Training

The backbone of our model is a multi-layer bidirectional transformer encoder (Devlin et al., 2019). 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 KnowEncoder$(\cdot)$ which facilitates the gradient flow back to memory. The training objective is Masked Language Modeling (Devlin et al., 2019) where we randomly mask tokens in a sentence and ask PlugLM to predict it. In the pre-training phase, Wikipedia is chosen as the source of knowledge and in the domain adaptation fine-tuning stage, corpora from other domains are treated as knowledge sources detailed in §5.1. More details are shown in Appendix A. 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.

5 Experiments

PlugLM mainly tries to decouple the knowledge storage from parameters and leverage knowledge in an explainable way. We conduct comprehensive experiments to show the superiority of this novel architecture: we could easily adapt the model to different domains without in-domain pre-training by switching DPM (§5.1.1 and §5.1.2), alleviate catastrophic forgetting by storing DPM (§5.1.1), inject new knowledge into the model by enlarging DPM (§5.2), further enhance the model by injecting in-task knowledge into DPM (§5.3) and unveil the black-box PLM with direct access to the knowledge retrieved from DPM (Appendix D). We also carefully examine each key design in PlugLM and point the direction for future work in §5.4.

5.1 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; Li et al., 2023). In this section, we show that without any in-domain pre-training, PlugLM could flexibly adapt to multiple domains with domain-specific DPM. For the existing PLM encoding knowledge in parameters, this is a challenging task in that it can not guarantee the generalization across multiple domains due to catastrophic forgetting (Kirkpatrick et al., 2016) and sometimes it is even computationally unaffordable to keep training the super large models (Smith et al., 2022; Brown et al., 2020).

We consider two adaptation scenarios: domain adaptive post-training (§5.1.1) and in-domain pre-training (§5.1.2). The former is conducted after PLM was trained on the general domain and the latter trains a domain-specific PLM from scratch.

5.1.1 Domain Adaptive Post-Training

Experimental Setting 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 B. When fine-tuning, we pass the final [CLS] representation to a task-specific head as in Devlin et al. (2019).

Model	BioMed		CS		News		Reviews		Avg. Gain	Avg. Cost
	CHEM.	RCT	ACL.	SCI.	HYP.	AG.	HP.	IMDB		
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	<u>6.3 h</u>
¬ 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	<u>6.3 h</u>
PlugLM	78.02	87.12	63.77	78.56	84.32	93.23	67.83	91.24	-	-
+ DAA	<u>82.56</u>	<u>88.13</u>	<u>72.51</u>	**83.00**	<u>88.16</u>	**94.11**	<u>69.28</u>	92.56	<u>+3.28</u>	**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**	<u>82.56</u>	**89.26**	<u>93.55</u>	**69.41**	<u>92.78</u>	**+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). The best scores are in bold, and the second best are underlined.

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 external knowledge base and can be simply adapted to other domains with a different base. We have two adaptation strategies: <u>DAA</u>, short for Domain Adaptive Addition, appends domain knowledge to the knowledge base, and <u>DAR</u>, Domain Adaptive Replacement, replaces general knowledge with domain-specific knowledge in the knowledge base.

We also include the results of <u>¬DAPT</u>, <u>¬DAA</u> and <u>DACT</u>. The former two use irrelevant domain corpora for post-training and knowledge base construction, which are used to test the robustness of the adaptation method and rule out the factor that improvements might be attributed simply to exposure to more data[3]. For DACT, Domain Adaptive Continual Training, we sequentially use DAPT for WikiBERT in multiple domains in the hope that it can capture and store knowledge from various domains in a lifelong learning way (Rostami, 2021).

[3]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.

Experimental Results 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 base. 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, we keep training WikiBERT 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). PlugLM could alleviate this catastrophic forgetting problem by storing all kinds of knowledge in DPM and using it in a plug-and-play manner. (3) Direct modification on external memory helps PlugLM efficiently and effectively adapt to different domains without in-domain training. In 254× less time compared with DAPT and in 40× less time compared with REALM, PlugLM significantly outperforms DAPT and REALM-based methods.

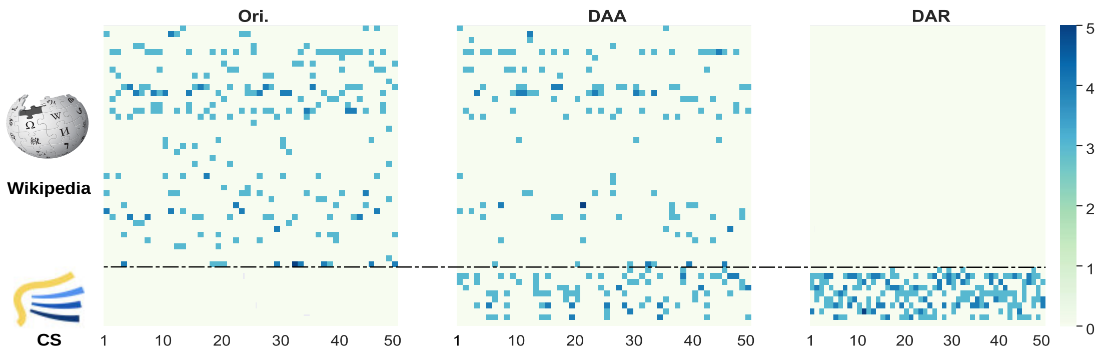


Figure 2: Knowledge retrieval visualization. We randomly sample 50 samples from ACL-ARC test set and check what kind of knowledge does PlugLM use to solve CS-specific tasks. Each column is one sample and the row is the index of retrieved knowledge in DPM. Their corresponding F1 scores are 63.77, 72.51 and 75.32.

To further understand PlugLM, in Figure 2, we present a visualization for the distribution of actual retrieved knowledge for DAA, DAR and original PlugLM. A clear pattern here is that with more domain knowledge involved, the model performs better (63.77, 72.51 and 75.32) and remarkably, 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.

5.1.2 In-domain Pre-Training

In-domain pre-training is another line of work for domain-specific PLM training from scratch like BioBERT (Lee et al., 2019), SciBERT (Beltagy et al., 2019) and FinBERT (Araci, 2019).

Experimental Setting In this section, we choose the biomedical domain and compare PlugLM with model in the architecture of BERT$_{base}$, pre-trained on the general domain, Wikipedia (i.e., WikiBERT) and pre-trained on the biomedical domain, Pubmed (i.e., PubmedBERT). The statistics of datasets and pre-training details are listed in Appendix F. We test two kinds of abilities 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., 2019) and SQUAD (Rajpurkar et al., 2016, 2018). For SQUAD and GLUE, the DPM is constructed from Wikipedia, and for biomedical NER, DPM is from PubMed (Canese and Weis, 2013).

Experimental Results 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 (Table 4) and SQUAD (Table 2), especially in low resource scenario (i.e., RTE, COLA and MRPC datasets). With DPM, PlugLM shows great flexibility and performs well in both general domain and biomedical domain. In Appendix D, we give concrete cases of PlugLM with respect to the retrieved knowledge.

	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 2: SQUAD results measured by EM and F1.

5.2 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 (Brown et al., 2020) and MT-NLG (Smith et al., 2022), efficiently fine-tuning for downstream tasks remains an open challenge, let alone re-training it on the newly coming knowledge.

Experimental Setting 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

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 across eight datasets.

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

Table 4: GLUE results. Detailed metrics and latency of each model is in Appendix C

consider the following two settings. (1) We only pre-train PlugLM 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 $\langle \mathbb{D}, \mathbb{K}, \mathbb{V} \rangle$.

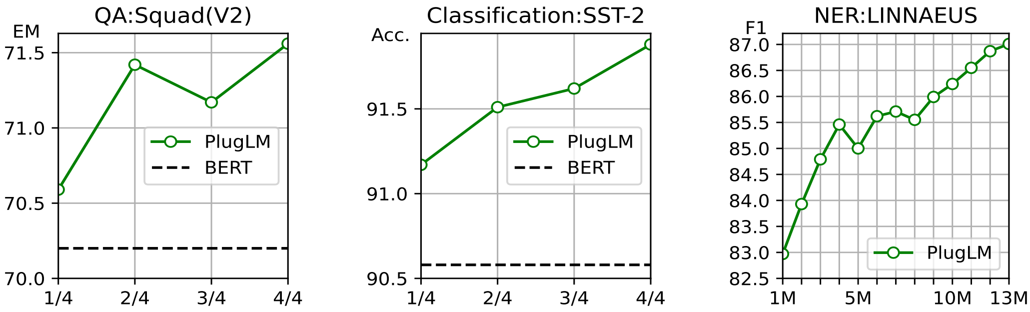
Experimental Results The results are shown in Figure 3a and 3b. For the first setting, we test on QA (SQUAD) and Sentiment Classification tasks (SST-2). Both WikiBERT and PlugLM are pre-trained with only 1/4 Wikipedia corpus. We have the following observations: (1) PlugLM trained with limited data already 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 verifies PlugLM could absorb new knowledge simply by adding more slots in $\langle \mathbb{D}, \mathbb{K}, \mathbb{V} \rangle$ without heavy re-training.

For the second setting, Figure 3c 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.

5.3 In-task Knowledge

Inspired by in-context learning (Brown et al., 2020) and example-augmented generation (Cheng et al., 2022, 2023b), the training samples can also be viewed as a kind of in-task knowledge. In this section, we broaden the scope of DPM knowledge by including the training samples.

Experimental Setting Since the knowledge from Wikipedia is a textual description from domain experts while the training sample from a Question-answering NLI dataset is in the form of [Q, A, Label], this surface form distribution shift may affect the knowledge retrieval. We consider the following injection methods. (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]), and use these tagged samples in both DPM and as model input. (3) Knowledge Prompting. Inspired by prompting method (Liu et al., 2021; Schick and 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 [Q, A, Label] with the 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.



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

Figure 3: Knowledge update results in QA, Sentiment Classification and NER.

Experimental Results The result is shown in Table 5. We can observe that PlugLM has managed to learn from in-task knowledge and the surface-form of knowledge affect 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 help PlugLM learn task-specific knowledge.

Task	Ori.	Concate.	Tagged.	Prompting.
QNLI	91.28	91.28	91.37	**91.58**
QQP	90.56	90.12	90.76	**91.47**

Table 5: Performance of in-task knowledge on `QNLI` and `QQP` measured by accuracy.

5.4 Tuning PlugLM

We investigate how each key design affects the performance of PlugLM. (1) **Number of Retrieved Knowledge.** Figure 4 shows the effects of different N in STS-B dataset and the sparsely activated Top-5 knowledge proves to be optimal. (2) **Layers equipped with DPM.** Considering that the upper layers in PLM capture more semantic information (Geva et al., 2021), we equip the last encoder layer with DPM in PlugLM. Figure 4 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 PlugLM $_{\text{All}}$. Then we fuse FFN and DPM by modifying the model architecture from $\text{LayerNorm}(h + \text{KnowAttn}(h, \mathbb{K}_{h'}, \mathbb{V}_{h'}))$ to $\text{LayerNorm}(h + \text{KnowAttn}(h, \mathbb{K}_{h'}, \mathbb{V}_{h'}) + \text{FFN}(h))$ and we name it PlugLM $_{\text{Fuse}}$. The Spearman correlation (more results are shown in Appendix E) in STS-B dataset for WikiBERT, PlugLM $_{\text{All}}$, PlugLM and PlugLM $_{\text{Fuse}}$ is 88.64, 86.82, 89.20 and 89.10. We could find that PlugLM $_{\text{All}}$, where there is no FFN, underperforms WikiBERT. And PlugLM performs comparably with PlugLM $_{\text{Fuse}}$. We conjecture that FFN in different layers may play different roles, which is also reported in Geva et al. (2021). For the upper layer which captures more semantic knowledge (Jawahar et al., 2019), DPM is a flexible and extensible substitution of FFN, but for lower layers, shallow features should be captured in the model parameters.

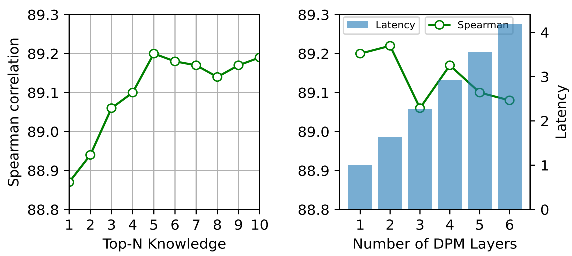


Figure 4: Effect of the number of retrieved knowledge and the number of DPM-enhanced layers in `STS-B` measured by spearman correlation.

6 Conclusion

For the first time, we challenge the current implicit knowledge encoding mechanism for PLMs with two fundamental drawbacks and insightfully propose to decouple knowledge storage from model parameters with an editable and scalable key-value memory. Inspired by the findings that FFN stores all kinds of knowledge and is essentially a key-value memory network, we transform FFN architecture into deep retrieval with a differentiable plug-in memory (DPM), which makes the knowledge encoding of PLMs more flexible and interpretable. Extensive experimental results in different scenarios including domain adaptation, knowledge update and in-task knowledge learning verify the design choice of PlugLM. We believe this architectural design would pave a new direction for future research on PLM, especially for super-large PLM.

Limitations

We discuss the limitations of PlugLM as follows:

(1) Despite the strong performance achieved by our approach with DPM, it results in a reduced inference efficiency at the same time due to the MIPS search. For example, PlugLM is about two times slower than pure transformer-based models in GLUE. This would be more crucial when the external memory is much larger. Potential solutions to this issue include (1) constructing the memory using a coarser granularity (Borgeaud et al., 2022); (2) compressing DPM by semantic clustering as in Tay et al. (2022) or knowledge summarization as in Xu et al. (2022).

(2) In this paper, we choose Wikipedia for DPM construction and PlugLM pre-training. While Wikipedia is the most commonly used data source for language model pre-training (Devlin et al., 2019; Liu et al., 2019), there are also many other types of knowledge not covered in Wikipedia, and how to integrate different types of knowledge (e.g., factual, commonsense, syntactic and semantic knowledge) into our framework remains under-explored.

(3) Although this paper proposes a general architecture that is applicable to PLMs of all kinds and sizes including bidirectional (Devlin et al., 2019; Liu et al., 2019; Yang et al., 2019), unidirectional (Radford et al., 2018, 2019; Brown et al., 2020) and encoder-decoder-based PLM (Lewis et al., 2020b; Raffel et al., 2020; Song et al., 2019), we only experiment with bidirectional models in moderate size. In particular, we believe this architectural design would be greatly beneficial for LLM (Smith et al., 2022; Chowdhery et al., 2022; Ouyang et al., 2022) for the following reasons: (1) the parameters of LLM could not be easily updated once the pre-training is done due to the unaffordable training cost. (2) the additional latency cost by MIPS retrieval is negligible compared with that of the whole LLM.

Acknowledgement

This work was supported by National Natural Science Foundation of China (NSFC Grant No. 62122089), the National Key Research and Development Program of China (No. 2021YFC3340304).

References

Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. 2021. 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, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 3554–3565. Association for Computational Linguistics.

Dogu Araci. 2019. Finbert: Financial sentiment analysis with pre-trained language models. *CoRR*, abs/1908.10063.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. 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 2019, Hong Kong, China, November 3-7, 2019*, pages 3613–3618. Association for Computational Linguistics.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. 2022. Improving language models by retrieving from trillions of tokens. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 2206–2240. PMLR.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared 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 M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher 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 33:*

Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual.

Deng Cai, Yan Wang, Huayang Li, Wai Lam, and Lemao Liu. 2021. 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)*, pages 7307–7318, Online. Association for Computational Linguistics.

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

Wenhu Chen, Pat Verga, Michiel de Jong, John Wieting, and William W. Cohen. 2022. Augmenting pretrained language models with qa-memory for open-domain question answering. *CoRR*, abs/2204.04581.

Xin Cheng, Shen Gao, Lemao Liu, Dongyan Zhao, and Rui Yan. 2022. Neural machine translation with contrastive translation memories. *CoRR*, abs/2212.03140.

Xin Cheng, Shen Gao, Yuchi Zhang, Yongliang Wang, Xiuying Chen, Mingzhe Li, Dongyan Zhao, and Rui Yan. 2023a. Towards personalized review summarization by modeling historical reviews from customer and product separately.

Xin Cheng, Di Luo, Xiuying Chen, Lemao Liu, Dongyan Zhao, and Rui Yan. 2023b. Lift yourself up: Retrieval-augmented text generation with self memory.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways. *CoRR*, abs/2204.02311.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022a. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 8493–8502. Association for Computational Linguistics.

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

Michiel de Jong, Yury Zemlyanskiy, Nicholas FitzGerald, Fei Sha, and William W. Cohen. 2022. Mention memory: incorporating textual knowledge into transformers through entity mention attention. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Franck Dernoncourt and Ji Young Lee. 2017. Pubmed 200k RCT: a dataset for sequential sentence classification in medical abstracts. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing, IJCNLP 2017, Taipei, Taiwan, November 27 - December 1, 2017, Volume 2: Short Papers*, pages 308–313. Asian Federation of Natural Language Processing.

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.

Mihail Eric, Lakshmi Krishnan, François Charette, and Christopher D. Manning. 2017. Key-value retrieval networks for task-oriented dialogue. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue, Saarbrücken, Germany, August 15-17, 2017*, pages 37–49. Association for Computational Linguistics.

Thibault Févry, Livio Baldini Soares, Nicholas FitzGerald, Eunsol Choi, and Tom Kwiatkowski. 2020. Entities as experts: Sparse memory access with entity supervision. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 4937–4951. Association for Computational Linguistics.

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

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 5484–5495. Association for Computational Linguistics.

Suchin Gururangan, Tam Dang, Dallas Card, and Noah A. Smith. 2019. Variational pretraining for semi-supervised text classification. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5880–5894. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasovic, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8342–8360. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 3929–3938. PMLR.

Ruining He and Julian J. McAuley. 2016. 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, WWW 2016, Montreal, Canada, April 11 - 15, 2016*, pages 507–517. ACM.

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

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3651–3657. Association for Computational Linguistics.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know. *Trans. Assoc. Comput. Linguistics*, 8:423–438.

David Jurgens, Srijan Kumar, Raine Hoover, Daniel A. McFarland, and Dan Jurafsky. 2018. Measuring the evolution of a scientific field through citation frames. *Trans. Assoc. Comput. Linguistics*, 6:391–406.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. 2019. SemEval-2019 Task 4: Hyperpartisan news detection. In *SemEval*.

James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2016. Overcoming catastrophic forgetting in neural networks. *CoRR*, abs/1612.00796.

Jens Kringelum, Sonny Kim Kjærulff, Søren Brunak, Ole Lund, Tudor I. Oprea, and Olivier Taboureau. 2016. ChemProt-3.0: a global chemical biology diseases mapping. In *Database*.

Guillaume Lample, Alexandre Sablayrolles, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2019. Large memory layers with product keys. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8546–8557.

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. *CoRR*, abs/1901.08746.

Mike Lewis, Marjan Ghazvininejad, Gargi Ghosh, Armen Aghajanyan, Sida Wang, and Luke Zettlemoyer. 2020a. Pre-training via paraphrasing. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020b. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020c. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Jinpeng Li, Yingce Xia, Xin Cheng, Dongyan Zhao, and Rui Yan. 2023. Learning disentangled representation via domain adaptation for dialogue summarization.

In *Proceedings of the ACM Web Conference 2023*, WWW '23, page 1693–1702, New York, NY, USA. Association for Computing Machinery.

Zonglin Li, Chong You, Srinadh Bhojanapalli, Daliang Li, Ankit Singh Rawat, Sashank J. Reddi, Ke Ye, Felix X. Chern, Felix X. Yu, Ruiqi Guo, and Sanjiv Kumar. 2022. Large models are parsimonious learners: Activation sparsity in trained transformers. *CoRR*, abs/2210.06313.

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

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

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel S. Weld. 2020. S2ORC: the semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 4969–4983. Association for Computational Linguistics.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. 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, Brussels, Belgium, October 31 - November 4, 2018*, pages 3219–3232. Association for Computational Linguistics.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pages 142–150. The Association for Computer Linguistics.

Julian J. McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, Santiago, Chile, August 9-13, 2015*, pages 43–52. ACM.

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

Alexander H. Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1400–1409. The Association for Computational Linguistics.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *CoRR*, abs/2203.02155.

Matthew E. Peters, Mark Neumann, Robert L. Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 43–54. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. 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)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2463–2473. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 784–789. Association for Computational Linguistics.

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, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

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

Mohammad Rostami. 2021. Lifelong domain adaptation via consolidated internal distribution. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 11172–11183.

Timo Schick and Hinrich Schütze. 2021. 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, EACL 2021, Online, April 19 - 23, 2021*, pages 255–269. Association for Computational Linguistics.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, Elton Zheng, Rewon Child, Reza Yazdani Aminabadi, Julie Bernauer, Xia Song, Mohammad Shoeybi, Yuxiong He, Michael Houston, Saurabh Tiwary, and Bryan Catanzaro. 2022. Using deepspeed and megatron to train megatron-turing NLG 530b, A large-scale generative language model. *CoRR*, abs/2201.11990.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: masked sequence to sequence pre-training for language generation. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 5926–5936. PMLR.

Sainbayar Sukhbaatar, Edouard Grave, Guillaume Lample, Hervé Jégou, and Armand Joulin. 2019. Augmenting self-attention with persistent memory. *CoRR*, abs/1907.01470.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2440–2448.

Yu Sun, Shuohuan Wang, Yu-Kun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. ERNIE: enhanced representation through knowledge integration. *CoRR*, abs/1904.09223.

Yi Tay, Vinh Q. Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Prakash Gupta, Tal Schuster, William W. Cohen, and Donald Metzler. 2022. Transformer memory as a differentiable search index. *CoRR*, abs/2202.06991.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Pat Verga, Haitian Sun, Livio Baldini Soares, and William W. Cohen. 2020. Facts as experts: Adaptable and interpretable neural memory over symbolic knowledge. *CoRR*, abs/2007.00849.

Jonas Wallat, Jaspreet Singh, and Avishek Anand. 2021. Bertnesia: Investigating the capture and forgetting of knowledge in BERT. *CoRR*, abs/2106.02902.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

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

Alex Warstadt, Yu Cao, Ioana Grosu, Wei Peng, Hagen Blix, Yining Nie, Anna Alsop, Shikha Bordia, Haokun Liu, Alicia Parrish, Sheng-Fu Wang, Jason Phang, Anhad Mohananey, Phu Mon Htut, Paloma Jeretic, and Samuel R. Bowman. 2019. 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, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2877–2887. Association for Computational Linguistics.

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

Taesun Whang, Dongyub Lee, Chanhee Lee, Kisu Yang, Dongsuk Oh, and Heuiseok Lim. 2020. An effective domain adaptive post-training method for BERT in response selection. In *Interspeech 2020, 21st Annual*

Conference of the International Speech Communication Association, Virtual Event, Shanghai, China, 25-29 October 2020, pages 1585–1589. ISCA.

Tianyu Xu, Wen Hua, Jianfeng Qu, Zhixu Li, Jiajie Xu, An Liu, and Lei Zhao. 2022. Evidence-aware document-level relation extraction. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, October 17-21, 2022*, pages 2311–2320. ACM.

Yichong Xu, Chenguang Zhu, Ruochen Xu, Yang Liu, Michael Zeng, and Xuedong Huang. 2021. Fusing context into knowledge graph for commonsense question answering. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 1201–1207. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

Yunzhi Yao, Shaohan Huang, Li Dong, Furu Wei, Huajun Chen, and Ningyu Zhang. 2022. Kformer: Knowledge injection in transformer feed-forward layers. In *Natural Language Processing and Chinese Computing - 11th CCF International Conference, NLPCC 2022, Guilin, China, September 24-25, 2022, Proceedings, Part I*, volume 13551 of *Lecture Notes in Computer Science*, pages 131–143. Springer.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 9051–9062.

Rong Zhang, Revanth Gangi Reddy, Md Arafat Sultan, Vittorio Castelli, Anthony Ferritto, Radu Florian, Efsun Sarioglu Kayi, Salim Roukos, Avi Sil, and Todd Ward. 2020. Multi-stage pre-training for low-resource domain adaptation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5461–5468, Online. Association for Computational Linguistics.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: enhanced language representation with informative entities. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1441–1451. Association for Computational Linguistics.

Zexuan Zhong, Tao Lei, and Danqi Chen. 2022. Training language models with memory augmentation. *CoRR*, abs/2205.12674.

A PlugLM Pretraining Details

The details of PlugLM pre-training is shown in Table 6

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 6: Hyperparameters for PlugLM pretraining.

B Data for Domain Adaptive Post-Training

The detailed statistics of domain corpora for post-training is listed in the Table 7 and downstream tasks in Table 8.

C Latency

In Table 9, we show the detailed latency of WikiBERT and PlugLM.

D Case Study

We show three concrete examples from QNLI and ACL-ARC in Table 13,14,15.

E More Experiments for Tuning PlugLM

In Table 10, we show more results in Section 5.4 on `STS-b`, `MRPC` and `QNLI`.

	WikiBERT	PlugLM $_{\text{All}}$	PlugLM $_{\text{Fuse}}$	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 10: Experimental Results as in Section 5.4 on `STS-b`, `MRPC` and `QNLI`. The evaluation metrics are Spearman correlation, F1 score and Accuracy respectively.

F Details for Wikipedia and Pubmed

The source and size of Wikipedia and Pubmed are shown in Table 11. And hyper-parameters for WikiBERT and PubmedBERT pre-training is shown in Table 12.

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 12: Hyperparameters for WikiBERT and PubmedBERT pretraining.

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 and McAuley, 2016)	2.11B	11GB

Table 7: 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 8: Specifications of the various target task datasets. † indicates high-resource settings. Sources: CHEMPROT (Kringelum et al., 2016), RCT (Dernoncourt and 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).

	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
Metrics	Accuracy	Matthews	F1	Spearman	Accuracy	Accuracy	Accuracy	Accuracy
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 9: 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.

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

Table 11: List of the PubMed and Wikipedia.

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.

ACL 2023 Responsible NLP Checklist

A For every submission:

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

☐ A2. Did you discuss any potential risks of your work?
Not applicable. Left blank.

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

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

B ☑ Did you use or create scientific artifacts?

code will be released when published

☑ B1. Did you cite the creators of artifacts you used?
section 5

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Not applicable. Left blank.

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
section 5

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

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

C ☑ Did you run computational experiments?

section 5

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
section 5

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

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
section 5

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
section 5

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
section 5

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

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.