Plug-in and Fine-tuning: Bridging the Gap between Small Language Models and Large Language Models

Kyeonghyun Kim[1*], Jinhee Jang[1*], Juhwan Choi[2*†], Yoonji Lee[1], Kyohoon Jin[3†], YoungBin Kim[1]

[1]Chung-Ang University [2]AITRICS [3]DATUMO

[1]{khyun8072, jinheejang, pioneer0305, ybkim85}@cau.ac.kr

[2]jhchoi@aitrics.com [3]kyohoon.jin@selectstar.ai

Abstract

Large language models (LLMs) are renowned for their extensive linguistic knowledge and strong generalization capabilities, but their high computational demands make them unsuitable for resource-constrained environments. In contrast, small language models (SLMs) are computationally efficient but often lack the broad generalization capacity of LLMs. To bridge this gap, we propose PiFi, a novel framework that combines the strengths of both LLMs and SLMs to achieve high performance while maintaining efficiency. PiFi integrates a single frozen layer from an LLM into a SLM and fine-tunes the combined model for specific tasks, boosting performance without a significant increase in computational cost. We show that PiFi delivers consistent performance improvements across a range of natural language processing tasks, including both natural language understanding and generation. Moreover, our findings demonstrate PiFi's ability to effectively leverage LLM knowledge, enhancing generalization to unseen domains and facilitating the transfer of linguistic abilities.

1 Introduction

Language models (LMs) based on transformer architecture have demonstrated impressive performance across a wide range of natural language processing (NLP) tasks, primarily due to the linguistic knowledge they acquire from training on large-scale datasets (Zhao et al., 2023). In particular, large language models (LLMs), such as GPT-3 (Brown et al., 2020), GPT-4 (Achiam et al., 2023), and Llama-3 (Dubey et al., 2024), stand out for their significantly larger number of parameters compared to small language models (SLMs), which typically contain between 100M and 5B parameters (Lu et al., 2024). Notable examples of SLMs include BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and BART (Lewis et al., 2020). The extensive parameterization of LLMs allows them to capture a broader range of knowledge, leading to enhanced generalizability across novel tasks, whereas SLMs often require fine-tuning for specific tasks or domains (Brown et al., 2020; Wei et al., 2022; Ye, 2024).

Despite the notable strengths and exceptional performance of LLMs, they come with inherent limitations. A major constraint is the trade-off between performance and inference cost, primarily due to the computational resources required to deploy these large models (Shashidhar et al., 2023). As the number of parameters increases, so does the demand for computational power and memory, making LLMs less suitable for environments with limited resources, such as mobile devices (Nityasya et al., 2020; Lin et al., 2023). Similarly, fine-tuning LLMs for domain-specific applications involves a substantial computational overhead (Dettmers et al., 2023; Zhang et al., 2024a).

In such resource-constrained scenarios, SLMs often provide a more viable solution. Studies have shown that fine-tuned SLMs can outperform LLMs in specific tasks such as sentiment analysis, semantic textual similarity evaluation, and named entity recognition (Yu et al., 2023; Lepagnol et al., 2024). Their lightweight architecture makes them an appealing choice for scenarios where computational efficiency and reduced memory usage are critical (Gao et al., 2023b; Lepagnol et al., 2024).

To address these challenges, we propose the plug-in and fine-tuning (PiFi) framework, which aims to harness the extensive knowledge and strengths of LLMs, such as linguistic ability and domain generalizability, while retaining the computational efficiency of SLMs. PiFi achieves this by extracting a single layer from a designated LLM and integrating it into a SLM, followed by fine-tuning the combined model on the target task. By

[*]Equal contribution.

[†]Work was done at Chung-Ang University.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 5434–5452

incorporating a single LLM layer rather than the full model, PiFi enhances the SLM without compromising its lightweight structure. Furthermore, PiFi optimizes fine-tuning by freezing the extracted LLM layer, thereby minimizing the number of additional parameters to be trained.

To validate the effectiveness of the PiFi framework, we conducted extensive experiments across various datasets, involving diverse natural language understanding (NLU) and natural language generation (NLG) tasks. Additionally, our evaluations in settings such as domain adaptation and multilingual classification demonstrate that PiFi can impart LLMs' benefits to SLMs, such as improved domain generalizability. In particular, our multilingual classification experiments show that PiFi can significantly enhance SLMs performance by leveraging a layer from an LLM pre-trained in the desired language, illustrating that even a straightforward integration of a single LLM layer can substantially assist SLM training through the knowledge transfer from the LLM. We also conducted a comprehensive comparison of different LLMs, evaluated the impact of varying model sizes, and assessed the effect of integrating instruction-tuned LLMs, highlighting the versatility and robustness of the PiFi framework.

2 Related Work

2.1 Small Language Models

Language models (LMs) have progressed significantly from early models like BERT (Devlin et al., 2019) and GPT-1 (Radford and Narasimhan, 2018), both of which are based on the transformer architecture (Vaswani et al., 2017), to more advanced models such as RoBERTa (Liu et al., 2019), DeBERTa DeBERTa (He et al., 2021), and BART (Lewis et al., 2020). These models utilize larger datasets and advanced strategies, achieving greater performance. With each iteration, these models have enhanced their performance and applicability across diverse tasks and languages.

Currently, open-sourced LMs are available in a range of parameter sizes, from extremely small models with 14.5 million parameters (Jiao et al., 2020) to large models with billions of parameters (Jiang et al., 2023; Almazrouei et al., 2023; Dubey et al., 2024; Yang et al., 2024; Team et al., 2024). In this study, we focus on specifically on SLMs with millions of parameters. Recent research has introduced several strategies to boost the performance and expand the utility of SLMs while maintaining computational efficiency (Gururangan et al., 2020; Gao et al., 2023b). Building upon these works, we propose the PiFi framework, which leverages the strengths of LLMs to supplement SLMs, effectively bridging the gap between SLMs and their larger counterparts.

2.2 Employment of Large Language Models

LLMs have achieved state-of-the-art performance across a broad range of NLP tasks, owing to their vast number of parameters. Models like GPT-4 (Achiam et al., 2023), Llama-3 (Dubey et al., 2024), and Mistral (Jiang et al., 2023) excel in generative tasks, particularly under zero-shot and few-shot settings. Additionally, LLMs demonstrate strong domain generalizability, making them applicable to a variety of domains without additional fine-tuning (Minaee et al., 2024).

These strengths have inspired researchers to explore novel strategies for leveraging LLMs beyond simple downstream task applications. One prominent approach is knowledge distillation, where the knowledge of LLMs is transferred to smaller models. There are two main categories of knowledge distillation methods: parametric and non-parametric. Parametric methods involve using white-box LLMs as teacher models and training student models using the output distribution or intermediate features of the teacher model (Zhong et al., 2024; Timiryasov and Tastet, 2023; Gu et al., 2024). Non-parametric methods, on the other hand, generate synthetic training data using LLMs, which is then used to train smaller student models, thereby achieving knowledge distillation from a data-centric perspective (Ye et al., 2022; West et al., 2022; Gao et al., 2023a; Choi et al., 2024).

In recent years, LLMs have also been integrated into multi-modal tasks, extending their utility beyond traditional NLP applications. For instance, some studies have used LLMs' linguistic capabilities to enhance the pre-training of vision-language models such as CLIP (Radford et al., 2021), by rewriting the textual descriptions in image-text pairs to improve the model's understanding of the relationships between images and text (Fan et al., 2023). Other research has explored using LLMs in computer vision, demonstrating that a combination of a vision transformer (ViT) model (Dosovitskiy et al., 2021) with an LLM layer can enhance performance on image classification and other vision

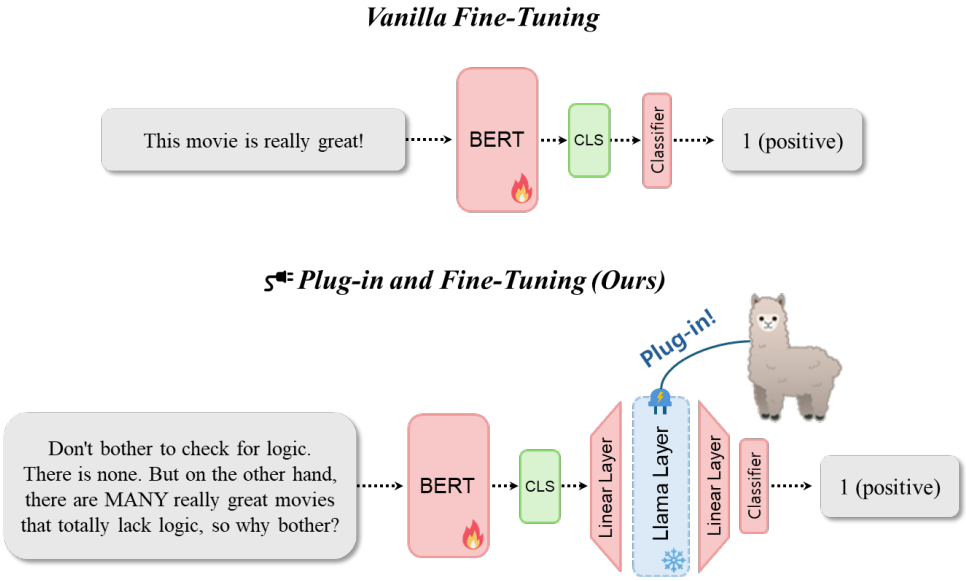

Figure 1: The comparison between vanilla fine-tuning of SLMs and our proposed PiFi architecture.

tasks (Pang et al., 2024). The study showed that LLM layers can serve as visual encoders, helping to identify patterns among image tokens and highlight salient features.

Inspired by these findings, we extend the use of LLMs to various NLP tasks. Specifically, we investigate whether key strengths of LLMs—such as generalizability to unseen domains and their rich linguistic knowledge—can be effectively distilled into SLMs, enabling them to benefit from these capabilities. Our work thus aims to bridge the performance gap between small and large models by transferring these strengths through a novel integration and fine-tuning framework.

3 Method

This section presents the PiFi framework, which integrates a single layer from an LLM into SLMs to leverage the extensive knowledge of LLMs while maintaining the efficiency of smaller models. We first describe the methodology for incorporating an LLM layer into encoder-based LMs, such as BERT, and encoder-decoder LMs like T5 (Raffel et al., 2020). Subsequently, we detail the fine-tuning process for SLMs with the integrated LLM layer. The overall procedure of the PiFi framework is illustrated in Figure 1.

3.1 Plug-in of LLM Layer to Encoder-based LMs

An encoder-based model, denoted as *Enc*, generates a hidden representation h_{enc} for an input sequence x, which is then passed to a classification head *Head* to produce the final prediction $\hat{y}$. This process is formulated as:

$$h_{enc} = Enc(x), \hat{y} = Head(h_{enc})$$

To extend this process, PiFi introduces a single LLM layer, denoted as L_{LLM}, into the pipeline between *Enc* and *Head*. Since the hidden representation size of SLMs (e.g., 768 for BERT) may differ from that of LLMs (e.g., 4096 for Llama-3), we employ two additional transformation layers: L_{in} and L_{out}. Specifically, L_{in} projects h_{enc} into a compatible dimension for L_{LLM}, which then processes the projected representation. Subsequently, L_{out} converts the output of L_{LLM} back to the original hidden representation size of h_{enc}. The final transformed feature is then fed into *Head* to generate the prediction $\hat{y}$. The overall process can be expressed as:

$$h_{enc} = Enc(x), h_{LLM} = L_{LLM}(L_{in}(h_{enc}))$$
$$\hat{y} = Head(L_{out}(h_{LLM}))$$

By introducing L_{LLM}, PiFi enables SLMs to benefit from the rich knowledge encoded within LLMs, thereby improving performance on various downstream tasks.

3.2 Plug-in of LLM Layer to Encoder-decoder LMs

Encoder-decoder models are widely used for sequence-to-sequence tasks such as machine translation and text summarization. An encoder-decoder model consists of an encoder, *Enc*, and a decoder, *Dec*. The encoder *Enc* processes the input sequence x to produce hidden representations h_{enc}, which are then passed to *Dec* to generate the target sequence $\hat{y}$. This process can be represented as:

$$h_{enc} = Enc(x), \hat{y}_t = Dec(h_{enc}, \hat{y}_{<t})$$

where $\hat{y}_{<t}$ denotes the tokens generated in previous time steps. The decoder *Dec* takes both $henc$ and $\hat{y}_{<t}$ as inputs to predict the next token $\hat{y}_t$.

In the PiFi framework, an LLM layer, L_{LLM}, is inserted between *Enc* and *Dec*. Similar to the procedure for encoder-based models, the hidden representation h_{enc} is first transformed using L_{in} to match the input size of L_{LLM}. After processing by L_{LLM}, the output is projected back to the original size using L_{out} before being fed into the decoder *Dec*. This updated procedure for predicting $\hat{y}_t$ can be formulated as:

$$h_{enc} = Enc(x), h_{LLM} = L_{LLM}(L_{in}(h_{enc}))$$
$$\hat{y}_t = Dec(L_{out}(h_{LLM}), \hat{y}_{<t})$$

By incorporating L_{LLM} in this manner, PiFi utilizes the LLM's knowledge to improve the performance of encoder-decoder models in generating high-quality target sequences.

3.3 Fine-tuning of SLMs with Additional Layer

During the fine-tuning stage, PiFi trains only the parameters of the original SLM, L_{in}, L_{out}, and the classification head. Importantly, the parameters of L_{LLM} are kept frozen and remain unchanged. This approach offers two key advantages: (1) it minimizes the number of additional parameters to be trained, and (2) it preserves the knowledge encoded in L_{LLM} during its pre-training stage. If we were to also update the parameters of L_{LLM}, the model might suffer from catastrophic forgetting, a phenomenon where previously learned knowledge is lost when the model adapts to new data (Luo et al., 2023; Wang et al., 2023). By freezing the LLM layer, PiFi mitigates this risk and maintains the effectiveness of L_{LLM} throughout fine-tuning.

We evaluate the impact of freezing L_{LLM} through an ablation study, as presented in Appendix B.4, demonstrating the effectiveness of this strategy in preventing catastrophic forgetting while ensuring optimal performance of the PiFi framework.

4 Experiment

In this section, we present a comprehensive evaluation of our proposed PiFi framework through various experiments.

4.1 Experimental Setup

We conduct experiments using Llama-3.1-8B (Meta, 2024) as the default LLM to extract L_{LLM}. Unless specified otherwise, the last layer of Llama-3.1-8B is integrated into a smaller LM as L_{LLM}.

We performed our experiments on various tasks across NLU tasks and NLG tasks. For NLU tasks, we adopted text classification, natural language inference (NLI), and question answering (QA) tasks. Specifically, we used SST-2 (Socher et al., 2013), IMDB (Maas et al., 2011), Tweet for sentiment classification and offensive language identification (Rosenthal et al., 2017; Zampieri et al., 2019; Barbieri et al., 2020), and CoLA (Warstadt et al., 2019) datasets for text classification tasks. For NLI tasks, we used MNLI (Williams et al., 2018) and SNLI (Bowman et al., 2015). For these two tasks, we measured the performance of the model through accuracy and F1-score. We used SQuAD v1.1 (Rajpurkar et al., 2016) for QA tasks, where the performance was measured by exact match and F1-score.

For NLG tasks, we evaluate each model through a machine translation task with Multi30k dataset (Elliott et al., 2016) and a text summarization task with CNN/DailyMail dataset (Nallapati et al., 2016). For both tasks, we used BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), METEOR (Banerjee and Lavie, 2005), BERTScore (Zhang et al., 2020), and BARTScore (Yuan et al., 2021) as a metric for measuring the performance of each model. All models were trained with five different random seeds, and we report the average performance for each experimental setup.

4.2 Performance Improvement of PiFi on NLU Tasks

We assess PiFi's effectiveness on NLU tasks using several SLMs: BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), ELECTRA (Clark

	Classification					NLI		QA	Average
	SST2	IMDB	Tweet (Sentiment)	Tweet (Offensive)	CoLA	MNLI	SNLI	SQuAD	
BERT$_{base}$	89.41	85.1	86.9	83.15	80.10	82.00	89.10	63.81	82.45
	0.8907	0.4733	0.862	0.7727	0.7398	0.8131	0.8892	0.7606	0.7752
+PiFi (Llama-3.1-8B)	**91.5**	**87.09**	**92.95**	**86.03**	**82.07**	**82.74**	**89.48**	**66.17**	**84.75**
	0.9125	**0.4800**	**0.9224**	**0.8026**	**0.7523**	**0.8185**	**0.8934**	**0.7809**	**0.7953**
RoBERTa$_{base}$	91.65	87.36	90.12	83.60	80.55	84.00	**88.73**	68.09	84.32
	0.9137	0.4890	0.894	0.7682	0.7192	0.8332	**0.8856**	0.8006	0.7879
+PiFi (Llama-3.1-8B)	**92.54**	**88.68**	**91.85**	**85.35**	**82.29**	**84.25**	88.49	**68.97**	**85.42**
	0.9228	**0.4878**	**0.9115**	**0.8083**	**0.7526**	**0.8346**	0.8837	**0.8089**	**0.7980**
ELECTRA$_{base}$	93.42	88.31	90.58	83.52	83.99	85.41	90.11	44.44	82.00
	0.9324	0.4974	0.8989	0.7751	0.7757	0.8472	0.8995	0.5706	0.7662
+PiFi (Llama-3.1-8B)	**94.13**	**89.40**	**0.9331**	**84.99**	**86.26**	**86.47**	**90.48**	**67.99**	**86.71**
	0.9393	**0.4994**	**0.9270**	**0.7848**	**0.8081**	**0.8618**	**0.9037**	**0.8045**	**0.8076**
DeBERTa$_{base}$	92.60	87.98	88.22	82.67	80.04	83.72	89.61	67.87	84.40
	0.9236	0.4882	0.8755	0.7729	0.7216	0.8312	0.8945	0.8094	0.7943
+PiFi (Llama-3.1-8B)	**93.04**	**88.85**	**92.17**	**85.47**	**80.87**	**84.87**	**90.62**	**69.65**	**85.63**
	0.9283	**0.4928**	**0.9152**	**0.8065**	**0.7347**	**0.8419**	**0.8982**	**0.8152**	**0.8037**
DeBERTa-V3$_{base}$	93.74	89.45	91.29	83.60	84.75	87.52	90.94	69.40	86.34
	0.9355	0.4951	0.9066	0.7915	0.8015	0.8687	0.9079	0.8249	0.8162
+PiFi (Llama-3.1-8B)	**95.01**	**89.83**	**93.80**	**85.60**	**86.07**	**87.98**	**91.05**	**69.87**	**87.40**
	0.9481	**0.5014**	**0.9325**	**0.8056**	**0.8118**	**0.8932**	**0.9095**	**0.8283**	**0.8287**

Table 1: Experimental results of PiFi on various NLU tasks and datasets. We used the last layer from Llama-3.1-8B as L_{LLM} in this experiment. For classification and NLI tasks, we report accuracy and F1-score in the upper row and lower row of each cell. For the QA task, we report the exact match and F1-score in the upper row and lower row of each cell.

et al., 2020), DeBERTa (He et al., 2021), and DeBERTaV3 (He et al., 2023).

Table 1 shows the results, where PiFi consistently outperforms vanilla fine-tuned SLMs across all datasets. For example, integrating PiFi into BERT resulted in a 2.3%p increase in average accuracy compared to standard fine-tuning. Similar gains are observed for other models, demonstrating PiFi's compatibility and effectiveness across a diverse set of NLU tasks and architectures.

4.3 Performance Improvement of PiFi on NLG Tasks

To validate PiFi on NLG tasks, we employed encoder-decoder models such as T5$_{base}$ (Raffel et al., 2020) and BART$_{base}$ (Lewis et al., 2020). These models were evaluated on machine translation and text summarization tasks, thereby validating the advantage of PiFi on NLG tasks beyond NLU tasks.

The experimental results on two NLG tasks are displayed in Table 2. We found that the SLM trained with PiFi exhibits higher performance compared to vanilla fine-tuned SLM in most cases. Particularly, for the text summarization task, both models trained with PiFi achieved higher scores across all evaluation metrics, indicating that PiFi effectively transfers knowledge from the LLM to SLMs, thus enhancing linguistic capabilities in NLG tasks.

4.4 Generalizability of PiFi under Domain Shift

Since LLMs are pre-trained on much larger and more diverse corpora compared to SLMs, integrating LLMs into SLMs through PiFi is expected to improve the generalizability of the model, particularly in unseen domains. To test this hypothesis, we conducted an experiment comparing the performance of the PiFi model against a vanilla fine-tuned model under conditions of domain shift. For this experiment, we used three text classification datasets. Specifically, we used IMDB, CR (Ding et al., 2008), and Tweet for sentiment classification. While all three datasets involve sentiment classification, they represent different domains such as movie review, electronics product review, and tweet messages. We trained the models on each dataset and evaluated their performance across all three datasets.

The results, shown in Table 3, indicate that PiFi

	Multi30K (Translation)					CNN/Daily Mail (Summarization)				
	BLEU	**ROUGE**	**METEOR**	**BERTS.**	**BARTS.**	**BLEU**	**ROUGE**	**METEOR**	**BERTS.**	**BARTS.**
$T5_{base}$	0.5301	0.6195	**0.3605**	0.8724	**-4.634**	0.2175	0.2323	0.1731	0.7409	-5.784
+ PiFi (Llama-3.1-8B)	**0.5413**	**0.6536**	0.3534	**0.8978**	-4.669	**0.2242**	**0.2357**	**0.1752**	**0.7412**	**-5.777**
$BART_{base}$	0.4580	0.5864	0.3331	**0.8635**	**-4.513**	0.2270	0.2348	0.1782	0.7424	-5.665
+ PiFi (Llama-3.1-8B)	**0.4695**	**0.5908**	**0.3364**	0.8617	-4.515	**0.2331**	**0.2355**	**0.1799**	**0.7425**	**-5.652**

Table 2: Experimental results of PiFi on machine translation and text summarization tasks. We used the last layer from Llama-3.1-8B as L_{LLM} in this experiment.

Train Dataset		**Test Dataset**		
		IMDB	**Tweet (Sentiment)**	**CR**
IMDB	$BERT_{base}$	85.1	70.40	74.56
		0.4773	0.6918	0.7301
	+PiFi (Llam-3-8B)	**87.09**	**83.68**	**79.86**
		0.4800	**0.8176**	**0.788**
	$BERT_{large}$	86.88	77.39	76.10
		0.4836	0.7647	0.7419
Tweet (Sentiment)	$BERT_{base}$	74.70	86.90	85.46
		0.4306	0.862	0.8357
	+PiFi (Llam-3-8B)	**77.28**	**92.95**	**87.25**
		0.4351	**0.9224**	**0.8596**
	$BERT_{large}$	75.91	89.26	86.50
		0.4331	0.8853	0.8526
CR	$BERT_{base}$	75.72	82.52	89.60
		0.4301	0.8161	0.8857
	+PiFi (Llam-3-8B)	**77.49**	**84.80**	**90.90**
		0.4362	**0.8365**	**0.9015**
	$BERT_{large}$	76.77	83.90	90.64
		0.4341	0.8284	0.8989

Table 3: Experimental results of PiFi under domain shift. We report accuracy and F1-score in the upper row and lower row of each cell.

consistently outperforms vanilla fine-tuning across all domain shifts. Notably, the PiFi model trained on IMDB showed significant performance improvements when tested on the Tweet and CR datasets, with gains of 13.28%p and 5.3%p, respectively, over the vanilla fine-tuned model. To verify that this improved generalization is due to leveraging LLM layer knowledge rather than merely a simple increase in parameters, we compared PiFi's fine-tuning performance against $BERT_{large}$, which is slightly larger than the PiFi model. The results showed that PiFi outperformed $BERT_{large}$, confirming that incorporating the LLM layer into an SLM via PiFi allows the model to effectively leverage the extensive knowledge stored in the LLM. This enhancement offers benefits beyond parameter scaling, thereby improving its ability to generalize to unseen domains.

	NSMC (Korean)	**Filmstarts (German)**
$mBERT_{base}$	83.62	86.77
	0.8318	0.8279
+ PiFi (Llama-3-8B English)	84.04	87.92
	0.8362	0.8362
+ PiFi (Llama-3-8B Korean)	**85.61**	87.09
	0.8522	0.8337
+ PiFi (Llama-3-8B German)	83.85	**88.11**
	0.8341	**0.8411**

Table 4: Experimental results on Korean and German text classification datasets. For this experiment, we used Llama-3 model trained in English, Korean, and German. We report accuracy and F1-score in the upper row and lower row of each cell.

4.5 Transferring of Linguistic Ability of LLMs to SLMs

In this section, we examine how the primary language of large language models (LLMs) influences the performance of the PiFi model when applied to smaller language models (SLMs). Specifically, we explore this effect by training an mBERT model (Pires et al., 2019) on two distinct datasets: NSMC (Park, 2016), a Korean sentiment classification dataset for movie reviews, and Filmstarts (Guhr et al., 2020), a German movie review sentiment dataset. For our experiments, we use Llama-3-8B[1] along with Llama-3-8B variants further fine-tuned for Korean and German (Lee, 2024; DiscoResearch, 2024) to extract L_{LLM}, which is then integrated into the mBERT model.

The results of our experiments are presented in

[1]Note that Llama 3, not Llama 3.1, was used in this experiment to ensure a fair comparison across English, German, and Korean models.

	SST-2	IMDB	Tweet (Sentiment)	Tweet (Offensive)	CoLA
BERT$_{base}$	89.41	85.10	86.90	83.15	80.10
	0.8907	0.4733	0.862	0.7727	0.7398
+ PiFi-Random	90.28	86.43	89.49	83.50	80.01
(Llama-3.1-8B)	0.8997	0.4879	0.9095	0.7564	0.7046
+ PiFi-Random-Full	90.50	86.46	90.95	83.95	80.39
(Llama-3.1-8B)	0.9017	0.4767	0.9024	0.7812	0.7376
+ PiFi	**91.50**	**87.09**	**92.95**	**86.03**	**82.07**
(Llama-3.1-8B)	**0.9125**	**0.48**	**0.9224**	**0.8026**	**0.7523**

Table 5: Experimental results of PiFi under different configurations of L_{LLM} initialization and fine-tuning. For this experiment, we compared PiFi-Random (frozen randomly initialized L_{LLM}), PiFi-Random-Full (fine-tuned randomly initialized L_{LLM}), and PiFi (pre-trained L_{LLM}). We report accuracy and F1-score in the upper row and lower row of each cell.

Table 4. We find that while PiFi demonstrates performance improvements on non-English datasets using the default English Llama-3 model, utilizing language-specific Llama-3 variants (e.g., Korean or German) leads to further performance gains for their respective languages. For example, the PiFi model, when combined with the Korean Llama-3 model, achieved an additional 1.57%p improvement over the English Llama-3 model on the Korean dataset. Conversely, when a PiFi model is trained on a different language than the target downstream task, the performance gains diminish. For instance, training PiFi on the Filmstarts (German) dataset with the Korean Llama-3 model resulted in a performance increase that was 0.83%p lower than that obtained with the English Llama-3 model and 1.02%p lower than that with the German Llama-3 model.

These findings indicate that the PiFi model benefits primarily from effectively leveraging knowledge from the LLM through a single layer, rather than simply from an increase in the number of model parameters. In conclusion, aligning the language of the downstream task with the language of the LLM used for L_{LLM} extraction is critical for maximizing the effectiveness of PiFi, as this alignment ensures optimal utilization of the LLM's linguistic capabilities.

4.6 Probing the Role of Intrinsic Knowledge in LLM Layer

To verify that the performance improvement achieved by plugging in L_{LLM} is not simply due to an increase in the number of parameters but rather a result of leveraging the inherent knowledge embedded in the LLM, we designed a series of experiments.

	Params (M)	SST-2	IMDB	Tweet (Sentiment)	Tweet (Offensive)	CoLA
BERT$_{base}$	110	89.41	85.10	86.90	83.15	80.10
		0.8907	0.4733	0.8620	0.7727	0.7398
+PiFi	334	**91.50**	**87.09**	**92.95**	**86.03**	**82.07**
(Llama-3.1-8B)		**0.9125**	**0.48**	**0.9224**	**0.8026**	**0.7523**
BERT$_{large}$	336	90.73	86.88	89.26	83.95	80.96
		0.9040	0.4836	0.8853	0.7887	0.7327

Table 6: Experimental results comparing the performance of PiFi and BERT$_{large}$ with similar parameter counts. We report accuracy and F1-score in the upper row and lower row of each cell.

	SST-2	**IMDB**	**Tweet (Sentiment)**	**Tweet (Offensive)**	**CoLA**
BERT$_{base}$	89.41	85.10	86.90	83.15	80.10
	0.8907	0.4733	0.862	0.7727	0.7398
+ PiFi	**91.50**	**87.09**	**92.95**	**86.03**	**82.07**
(Llama-3.1-8B)	**0.9125**	**0.48**	**0.9224**	**0.8026**	**0.7523**
+ PiFi	90.89	86.47	90.12	84.99	81.65
(Mistral-7B-v0.1)	0.9061	0.4794	0.8943	0.8055	0.7324
+ PiFi	91.65	87.22	92.57	85.33	81.72
(Mistral-7B-v0.3)	0.9136	0.4797	0.9177	0.802	0.7502
+ PiFi	91.17	86.68	92.48	85.70	81.62
(Qwen2 -7B)	0.9092	0.4774	0.917	0.809	0.7475
+ PiFi	91.39	87.1	92.34	84.29	80.74
(Gemma-2-9B)	0.9111	0.4849	0.9163	0.7866	0.7598
+ PiFi	91.44	86.63	92.51	84.85	80.99
(Falcon-7B)	0.9115	0.4779	0.9178	0.7907	0.7518

Table 7: Experimental results of PiFi across various LLMs, such as Llama-3.1-8B, Mistral-7B-v0.1 and v0.3, Qwen-7B, Gemma-2-9B, and Falcon-7B. We extracted the last layer from each LLM as L_{LLM} and compared their performance. We report accuracy and F1-score in the upper row and lower row of each cell.

First, Table 5 shows the results of experiments conducted with randomly initialized L_{LLM} under two configurations. PiFi-Random refers to a model where the randomly initialized L_{LLM} is frozen during fine-tuning, while PiFi-Random-Full is a model where the entire network, including the randomly initialized L_{LLM}, is fine-tuned. Although PiFi-Random and PiFi-Random-Full exhibited slight improvements over BERT$_{base}$, their performance fell short of PiFi, which utilizes the pre-trained weights of the L_{LLM}. These results demonstrate that the benefits of simply increasing the number of parameters are limited.

In addition to the comparison with randomly initialized baselines, Table 6 compares the performance of PiFi with BERT$_{large}$, which has a similar number of parameters. BERT$_{large}$ has approximately 336M parameters, while PiFi applied to BERT$_{base}$ has a comparable parameter count of approximately 334M. Despite this similarity in

	SST-2	IMDB	Tweet (Sentiment)	Tweet (Offensive)	CoLA
BERT$_{base}$	89.41	85.1	86.90	83.15	80.10
	0.8907	0.4733	0.862	0.7727	0.7398
+ PiFi	90.13	86.09	89.67	84.52	81.15
(Qwen2-0.5B)	0.8977	0.4843	0.8899	0.7959	0.7453
+ PiFi	91.06	86.29	92.35	85.91	81.54
(Qwen2-1.5B)	0.9079	0.4668	0.9164	0.8096	0.7453
+ PiFi	**91.17**	**86.68**	**92.48**	**86.04**	**81.62**
(Qwen2-7B)	**0.9092**	**0.4774**	**0.917**	**0.8068**	**0.7475**

Table 8: Experimental results of PiFi on different sizes of LM within same model family. For this experiment, we extracted the last layer of 0.5B, 1.5B, 7B version of Qwen2 as L_{LM}. We report accuracy and F1-score in the upper row and lower row of each cell.

parameters and native architecture of BERT$_{large}$, BERT$_{base}$ with PiFi outperformed BERT$_{large}$ in terms of performance.

This experimental finding suggests that the performance improvement of PiFi is not merely an outcome of increasing the number of parameters but is primarily due to effectively utilizing the pretrained knowledge embedded in the LLM.

4.7 Comparison of PiFi between Various LLMs

LLMs vary significantly in architecture and pretraining data, which can influence their performance as backbones for the PiFi framework. In this section, we evaluate the downstream task performance of PiFi when using different LLMs as sources for extracting L_{LLM}. Specifically, we incorporate the following LLMs: Llama-3.1-8B (Dubey et al., 2024), Mistral-7B-v0.1 and v0.3 (Jiang et al., 2023), Qwen2-7B (Yang et al., 2024), Gemma-2-9B (Team et al., 2024), and Falcon-7B (Almazrouei et al., 2023). These LLMs serve as backbones for generating L_{LLM}, which is subsequently integrated into a BERT model for the fine-tuning on text classification tasks.

Table 7 presents the results of these experiments. Our findings reveal several key insights. First, across all LLM variants, PiFi consistently outperforms the BERT model with vanilla fine-tuning, demonstrating the effectiveness of utilizing L_{LLM}. Second, Llama-3.1-8B and Mistral-7B-v0.3 yield the highest performance across multiple datasets, indicating their strong suitability as backbones for PiFi. Additionally, the comparison between PiFi models using Mistral-7B-v0.1 and v0.3 shows that even incremental improvements within the same model family lead to enhanced downstream performance. This result suggests that PiFi is highly responsive to the advancements in LLM capabilities, and we expect that future advancements in LLM development will further augment the effectiveness of PiFi.

4.8 Impact on PiFi Depending on LM Size

Currently, LMs are available in various sizes to cater to different computational and performance needs. In this section, we evaluate how the size of the LM used to extract LM layer affects the overall performance of PiFi. For this experiment, we utilize three versions of the Qwen2 model (Yang et al., 2024): Qwen2-0.5B, Qwen2-1.5B, and Qwen2-7B.

Table 8 demonstrates the results. We observe that integrating LM layer from any LM size into a SLM results in improved performance compared to traditional fine-tuning methods. Notably, the usage of 7B model yields the highest performance gains, surpassing both the 0.5B and 1.5B models. This suggests that larger models, with their enhanced capacity for capturing and storing extensive knowledge, provide more informative and effective representations for PiFi, leading to better downstream task performance.

5 Conclusion

In this paper, we introduced PiFi, plug-in and fine-tuning, a novel framework designed to leverage the intrinsic knowledge of LLMs while maintaining the efficiency and lightweight nature of SLMs. PiFi incorporates a frozen LLM layer into a SLM and fine-tunes the resulting model, enabling the effective utilization of LLM knowledge without significantly increasing model complexity.

We validated the applicability of PiFi across a diverse range of NLU and NLG tasks, demonstrating its compatibility with various SLMs and LLMs. Notably, the results from our domain shift experiment in Section 4.4 showed that PiFi can significantly improve the performance of SLMs on unseen domains. Similarly, the analysis in Section 4.5 confirmed that PiFi effectively leverages the linguistic capabilities of LLMs by incorporating even a single LLM layer.

In future work, we aim to extend the usability of PiFi to more tasks and languages. We will also explore advanced strategies to further optimize PiFi's effectiveness, such as automatically selecting the number and position of LLM layers to be integrated, allowing for more flexible and task-specific

knowledge transfer.

Limitations

In this study, we proposed PiFi, an efficient framework for integrating the knowledge of LLMs into SLMs. Despite its effectiveness, there are several limitations in our current approach that warrant further exploration.

One limitation is that we selected L_{LLM} as the last layer of each LLM based on a heuristic approach. While our analysis in Appendix B.1 indicates that using the last layer generally yields the best performance, it is possible that different layers may be optimal depending on the specific downstream task. Future research could explore methods for automatically selecting the most suitable LLM layer as L_{LLM}, akin to techniques used in neural architecture search (Elsken et al., 2019; White et al., 2023).

Another limitation is that our experiments primarily focused on relatively straightforward tasks, as this manuscript's primary goal is to propose PiFi and clearly demonstrate its effectiveness. We did not extend our evaluation to more complex benchmarks, such as the massive multitask language understanding dataset (Hendrycks et al., 2021). Testing PiFi on such diverse and challenging tasks in future work could provide deeper insights into its generalizability and highlight areas for further improvement.

Ethics Statement

We acknowledge the potential for inherent biases in LLMs, and the integration of an LLM layer into a SLM may introduce such biases (Liu et al., 2022). While our experiments did not reveal any explicit evidence of bias, future work will carefully consider the possibility of bias transfer from LLMs to SLMs when employing PiFi, as well as its broader implications.

Acknowledgements

This work was supported by the Institute of Information Communications Technology Planning Evaluation (IITP) grant funded by the Korea government (MSIT) [RS-2021-II211341, Artificial Intelligent Graduate School Program (Chung-Ang University)] and by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2025-00556246).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, et al. 2025. Smollm2: When smol goes big–data-centric training of a small language model. *arXiv preprint arXiv:2502.02737*.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, et al. 2023. The falcon series of open language models. *arXiv preprint arXiv:2311.16867*.

Amit Artzy and Roy Schwartz. 2024. Attend first, consolidate later: On the importance of attention in different llm layers. In *Proceedings of EMNLP 2024 BlackboxNLP Workshop*, pages 177–184.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of ACL 2005 Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72.

Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa Anke, and Leonardo Neves. 2020. TweetEval: Unified benchmark and comparative evaluation for tweet classification. In *Findings of EMNLP*, pages 1644–1650.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of EMNLP*, pages 632–642.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Proceedings of NeurIPS*, pages 1877–1901.

Juhwan Choi, Yeonghwa Kim, Seunguk Yu, Jungmin Yun, and Youngbin Kim. 2024. Unigen: Universal domain generalization for sentiment classification via zero-shot dataset generation. In *Proceedings of EMNLP*, pages 1–14.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *Proceedings of ICLR*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning

of quantized llms. In *Proceedings of NeurIPS*, pages 10088–10115.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*, pages 4171–4186.

Xiaowen Ding, Bing Liu, and Philip S Yu. 2008. A holistic lexicon-based approach to opinion mining. In *Proceedings of WSDM*, pages 231–240.

DiscoResearch. 2024. Llama-3-german-8b. Hugging Face Repository.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of ICLR*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Desmond Elliott, Stella Frank, Khalil Sima'an, and Lucia Specia. 2016. Multi30k: Multilingual english-german image descriptions. In *Proceedings of ACL 2016 Workshop on Vision and Language*, pages 70–74.

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. 2019. Neural architecture search: A survey. *Journal of Machine Learning Research*, 20(55):1–21.

Lijie Fan, Dilip Krishnan, Phillip Isola, Dina Katabi, and Yonglong Tian. 2023. Improving clip training with language rewrites. In *Proceedings of NeurIPS*, pages 35544–35575.

Jiahui Gao, Renjie Pi, Lin Yong, Hang Xu, Jiacheng Ye, Zhiyong Wu, Weizhong Zhang, Xiaodan Liang, Zhenguo Li, and Lingpeng Kong. 2023a. Self-guided noise-free data generation for efficient zero-shot learning. In *Proceedings of ICLR*.

Ze-Feng Gao, Kun Zhou, Peiyu Liu, Wayne Xin Zhao, and Ji-Rong Wen. 2023b. Small pre-trained language models can be fine-tuned as large models via over-parameterization. In *Proceedings of ACL*, pages 3819–3834.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024. Minillm: Knowledge distillation of large language models. In *Proceedings of ICLR*.

Oliver Guhr, Anne-Kathrin Schumann, Frank Bahrmann, and Hans Joachim Böhme. 2020. Training a broad-coverage German sentiment classification model for dialog systems. In *Proceedings of LREC*, pages 1627–1632.

Suchin Gururangan, Ana Marasović, 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 ACL*, pages 8342–8360.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2023. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. In *Proceedings of ICLR*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. In *Proceedings of ICLR*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *Proceedings of ICLR*.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *Proceedings of ICLR*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for natural language understanding. In *Findings of EMNLP*, pages 4163–4174.

Tianjie Ju, Weiwei Sun, Wei Du, Xinwei Yuan, Zhaochun Ren, and Gongshen Liu. 2024. How large language models encode context knowledge? a layer-wise probing study. In *Proceedings of LREC-COLING*, pages 8235–8246.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR*.

Junbum Lee. 2024. Llama-3-open-ko. Hugging Face Repository.

Pierre Lepagnol, Thomas Gerald, Sahar Ghannay, Christophe Servan, and Sophie Rosset. 2024. Small language models are good too: An empirical study of zero-shot classification. In *Proceedings of LREC-COLING*, pages 14923–14936.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of ACL*, pages 7871–7880.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proceedings of ACL 2004 Workshop Text Summarization Branches Out*, pages 74–81.

Zheng Lin, Guanqiao Qu, Qiyuan Chen, Xianhao Chen, Zhe Chen, and Kaibin Huang. 2023. Pushing large language models to the 6g edge: Vision, challenges, and opportunities. *CoRR*.

Ruibo Liu, Chenyan Jia, Jason Wei, Guangxuan Xu, and Soroush Vosoughi. 2022. Quantifying and alleviating political bias in language models. *Artificial Intelligence*, 304:103654.

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 preprint arXiv:1907.11692*.

Zhu Liu, Cunliang Kong, Ying Liu, and Maosong Sun. 2024. Fantastic semantics and where to find them: Investigating which layers of generative LLMs reflect lexical semantics. In *Findings of ACL*, pages 14551–14558.

Zhenyan Lu, Xiang Li, Dongqi Cai, Rongjie Yi, Fangming Liu, Xiwen Zhang, Nicholas D Lane, and Mengwei Xu. 2024. Small language models: Survey, measurements, and insights. *arXiv preprint arXiv:2409.15790*.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. 2023. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *arXiv preprint arXiv:2308.08747*.

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 *Proceedings of ACL*, pages 142–150.

Meta. 2019. fvcore. GitHub Repository.

Meta. 2024. Llama-3.1-8b. Hugging Face Repository.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint arXiv:2402.06196*.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gùlçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of SIGNLL*, pages 280–290.

Made Nindyatama Nityasya, Haryo Akbarianto Wibowo, Radityo Eko Prasojo, and Alham Fikri Aji. 2020. Costs to consider in adopting nlp for your business. *arXiv preprint arXiv:2012.08958*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. In *Proceedings of NeurIPS*, pages 27730–27744.

Ziqi Pang, Ziyang Xie, Yunze Man, and Yu-Xiong Wang. 2024. Frozen transformers in language models are effective visual encoder layers. In *Proceedings of ICLR*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of ACL*, pages 311–318.

Lucy Park. 2016. Naver sentiment movie corpus. GitHub Repository.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019a. Pytorch: An imperative style, high-performance deep learning library. In *Proceedings of NeurIPS*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019b. Pytorch: An imperative style, high-performance deep learning library. In *Proceedings of NeurIPS*.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of ACL*, pages 4996–5001.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of ICML*, pages 8748–8763.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training. *OpenAI Blog*.

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. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of EMNLP*, pages 2383–2392.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. Semeval-2017 task 4: Sentiment analysis in twitter. In *Proceedings of ACL 2017 Workshop on Semantic Evaluation*, pages 502–518.

Sumuk Shashidhar, Abhinav Chinta, Vaibhav Sahai, Zhenhailong Wang, and Heng Ji. 2023. Democratizing llms: An exploration of cost-performance trade-offs in self-refined open-source models. In *Findings of EMNLP*, pages 9070–9084.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP*, pages 1631–1642.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Inar Timiryasov and Jean-Loup Tastet. 2023. Baby llama: Knowledge distillation from an ensemble of teachers trained on a small dataset with no performance penalty. In *Proceedings of CoNLL 2024 BabyLM Challenge*, pages 279–289.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NeurIPS*, volume 30.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. 2023. Orthogonal subspace learning for language model continual learning. In *Findings of EMNLP*, pages 10658–10671.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022. Emergent abilities of large language models. *Transactions on Machine Learning Research*.

Peter West, Chandra Bhagavatula, Jack Hessel, Jena Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. 2022. Symbolic knowledge distillation: From general language models to commonsense models. In *Proceedings of NAACL*, pages 4602–4625.

Colin White, Mahmoud Safari, Rhea Sukthanker, Binxin Ru, Thomas Elsken, Arber Zela, Debadeepta Dey, and Frank Hutter. 2023. Neural architecture search: Insights from 1000 papers. *arXiv preprint arXiv:2301.08727*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of NAACL*, pages 1112–1122.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of EMNLP (Demo Track)*, pages 38–45.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Jiacheng Ye, Jiahui Gao, Qintong Li, Hang Xu, Jiangtao Feng, Zhiyong Wu, Tao Yu, and Lingpeng Kong. 2022. Zerogen: Efficient zero-shot learning via dataset generation. In *Proceedings of EMNLP*, pages 11653–11669.

Qinyuan Ye. 2024. Cross-task generalization abilities of large language models. In *Proceedings of NAACL 2024 Student Research Workshop*, pages 255–262.

Hao Yu, Zachary Yang, Kellin Pelrine, Jean-François Godbout, and Reihaneh Rabbany. 2023. Open, closed, or small language models for text classification? *arXiv preprint arXiv:2308.10092*.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. In *Proceedings of NeurIPS*, pages 27263–27277.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). In *Proceedings of NAACL 2019 Workshop on Semantic Evaluation*, pages 75–86.

Renrui Zhang, Jiaming Han, Chris Liu, Aojun Zhou, Pan Lu, Yu Qiao, Hongsheng Li, and Peng Gao. 2024a. LLaMA-adapter: Efficient fine-tuning of large language models with zero-initialized attention. In *Proceedings of ICLR*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *Proceedings of ICLR*.

Yang Zhang, Yanfei Dong, and Kenji Kawaguchi. 2024b. Investigating layer importance in large language models. In *Proceedings of EMNLP 2024 BlackboxNLP Workshop*, pages 469–479.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Ming Zhong, Chenxin An, Weizhu Chen, Jiawei Han, and Pengcheng He. 2024. Seeking neural nuggets: Knowledge transfer in large language models from a parametric perspective. In *Proceedings of ICLR*.

A Implementation Detail

We implemented PiFi using the PyTorch (Paszke et al., 2019b) and Transformers (Wolf et al., 2020) libraries. For encoder-based LMs, we used the representation of the *CLS* token as h_{enc}. Similarly, for encoder-decoder models, we used the hidden states of the last encoder layer for each token as h_{enc}. Each model was trained for three epochs with a learning rate of 5e-5 and a batch size of 32 using the Adam optimizer (Kingma and Ba, 2015). Table 20 provides the number of parameters used for each PiFi configuration across various SLMs and LLMs. Please refer to attached source code for further details.[2]

B Ablation Study

In this section, we present the results of ablation studies to supplement the main experiments conducted in Section 4.

B.1 Effect of Selection of Layers within LLM

For the experiments in Section 4, we consistently used the last layer of each LLM as L_{LLM}. However, recent studies have shown that different layers of LLMs capture distinct types of information and serve different roles (Artzy and Schwartz, 2024; Zhang et al., 2024b; Liu et al., 2024). To explore this, we analyzed the effectiveness of PiFi when extracting L_{LLM} from layers other than the last. Specifically, we experimented with layers $1, 4, 8, 12, 16, 20, 24, 28, 32$ of the LLMs used in Section 4.7.

Figure 2 presents the experimental results on the SST-2 dataset. The results clearly show that using the last layer yields the best performance compared to other layers, which aligns with previous findings suggesting that the upper layers of LLMs contain more contextual knowledge (Ju et al., 2024).

In addition to classification tasks, we also evaluated the effectiveness of PiFi in generation tasks. Specifically, we conducted a layer-wise performance analysis on the Multi30K translation task. As shown in Figure 3, the last layer consistently achieved the best performance in generation tasks. However, we acknowledge that for certain tasks, intermediate or other layers may be more suitable. This highlights the importance of task-specific optimization in layer selection. Therefore, in future research, we plan to explore optimization methods for selecting the most suitable layers for different tasks. Additionally, we aim to extend our work by leveraging automated techniques, such as Neural Architecture Search, to identify the optimal layers more efficiently.

[2] https://github.com/khyun8072/PiFi

	SST-2	IMDB	Tweet (Sentiment)	Tweet (Offensive)	CoLA
SmolLM2-135M	92.48	87.70	90.47	84.42	78.93
	0.9224	0.4817	0.8980	0.7673	0.7111
+PiFi	**93.42**	**88.69**	**92.20**	**86.04**	**81.22**
(Llama-3.1-8B)	**0.9319**	**0.4955**	**0.9160**	**0.7967**	**0.7359**

Table 9: Experimental results of PiFi on a decoder-based model. For this experiment, we used SmolLM2-135B model with Llama-3.1-8B. We report accuracy and F1-score in the upper row and lower row of each cell.

B.2 Performance of PiFi on Decoder-Based Models

The PiFi framework has primarily been tested on encoder-based and encoder-decoder models. However, it is crucial to verify whether it exhibits the same effectiveness in decoder-based models (e.g., GPT-style architectures). To investigate its applicability and effectiveness further, we evaluated the performance of PiFi on a decoder-based model, SmolLM2-135M (Allal et al., 2025). SmolLM2-135M is a relatively small decoder-based model, comparable in size to BERT-base. In this experiment, we added a classification head at the end of the model to enable it to perform various classification tasks.

As shown in Table 9, the decoder-based PiFi model outperforms the base SmolLM2-135M model across all tasks. This result indicates that PiFi is not limited to encoder-based and encoder-decoder models but can also be effectively applied to decoder-based architectures. These findings highlight the high versatility and scalability of PiFi, providing strong evidence for its applicability across a wide range of language model architectures.

B.3 Comparison between Text Representation Methods in Encoder-based LMs

In this section, we explore different methods for representing text in encoder-based LMs. In our

BERT$_{base}$ + PiFi (Llama-3.1-8B)	SST-2	IMDB	Tweet (Sentiment)
CLS	**91.50** **0.9125**	**87.09** **0.4800**	**92.95** **0.9224**
Avearge Pooling (w/o padding)	91.06 0.9079	86.25 0.4724	92.70 0.9197
Avearge Pooling (w/ padding)	90.40 0.9008	86.56 0.4758	92.73 0.9191

Table 10: Experimental results comparing the method for representing the input sequence for encoder-based LMs with PiFi. For this experiment, we used BERT$_{base}$ model with Llama-3.1-8B. We report accuracy and F1-score in the upper row and lower row of each cell.

	SST-2	IMDB	Tweet (Sentiment)
BERT$_{base}$	89.41 0.8907	85.10 0.4733	86.90 0.862
+ PiFi (Llama-3.1-8B)	**91.50** **0.9125**	**87.09** **0.4800**	**92.95** **0.9224**
+ PiFi-full (Llama-3.1-8B)	91.27 0.9106	86.00 0.4726	90.20 0.8951

Table 11: Experimental results to validate the effectiveness of fully fine-tuning L_{LLM} rather than keeping it frozen. We report accuracy and F1-score in the upper row and lower row of each cell.

original PiFi framework, we used the *CLS* token representation as h_{enc}. To evaluate alternative approaches, we conducted experiments with two additional methods: (1) representing h_{enc} as the average of all token representations in the input sequence, including padding tokens, and (2) representing h_{enc} as the average of all token representations excluding padding tokens.

Table 10 presents the experimental results. The results reveal distinct differences between these configurations, indicating that using the *CLS* token representation as h_{enc} is the most effective choice for ensuring optimal model performance.

B.4 Full fine-tuning of L_{LLM} in PiFi

In PiFi framework, we fine-tune the model while keeping L_{LLM} frozen. To validate the impact of this approach, we conducted an experiment where the entire PiFi model, including L_{LLM}, was fully fine-tuned.

Table 11 shows the results of this experiment. Freezing L_{LLM} during fine-tuning leads to better performance compared to fully fine-tuning the model. We hypothesize that freezing L_{LLM} preserves the extensive knowledge of the LLM, while full fine-tuning may cause catastrophic forgetting of this knowledge. Thus, it is crucial to keep L_{LLM} frozen to retain the LLM's vast knowledge, optimize the SLM for the desired task, and reduce training costs.

	SST-2	IMDB	Tweet (Sentiment)	Tweet (Offensive)	CoLA
BERT$_{base}$	89.41 0.8907	85.10 0.4733	86.90 0.8620	83.15 0.7727	80.10 0.7398
+ PiFi (Llama-3.1-8B)	91.50 0.9125	**87.09** **0.4800**	**92.95** **0.9224**	**86.03** **0.8026**	82.07 0.7523
+ PiFi (Llama-3.1-8B-Instruct)	**91.98** **0.9174**	86.72 0.4882	92.81 0.9206	85.57 0.7985	**82.45** **0.7450**
p-value	0.7352	0.3237	0.4422	0.2129	0.5308

Table 12: Experimental results comparing the usage of base LLM and instruction-tuned LLM for PiFi. For this experiment, we used Llama-3.1-8B and Llama-3.1-8B-Instruct. The p-value denotes the statistical significance between the distribution of accuracy of PiFi with Llama-3.1-8B and Llama-3.1-8B-Instruct. We report accuracy and F1-score in the upper row and lower row of each cell.

B.5 Effectiveness of Instruction-tuned LLM for PiFi

Recent innovations in LLMs involve additional training to enable these models to better follow instructions provided in prompts, leading to improved performance on various downstream tasks (Ouyang et al., 2022). In this section, we explore the impact of using instruction-tuned LLMs for extracting L_{LLM} within the PiFi framework. Specifically, we compare the performance of PiFi when using Llama-3.1-8B, our default model, against Llama-3.1-8B-Instruct, an instruction-tuned version of the Llama-3.1-8B.

The results are presented in Table 12. Our findings indicate that using instruction-tuned LLMs with PiFi can enhance the performance of SLMs, but the performance gains are marginal. In fact, there is no consistent trend showing that instruction-tuned models outperform their base counterparts across all datasets. To better understand this observation, we conducted a statistical significance test by comparing the results obtained from five different random seeds. The p-values for accuracy distributions between the instruction-tuned and base LLMs were greater than 0.05, indicating that these performance differences are not statistically significant.

This suggests that incorporating instruction-

	SST-2	IMDB	Tweet (Sentiment)	Tweet (Offensive)	CoLA
BERT-base	89.41	85.10	86.90	83.15	80.77
	0.8907	0.4733	0.8620	0.7727	0.7296
+PiFi	**91.50**	87.09	**92.95**	**86.03**	**82.07**
(Llama-3.1-8B)	**0.9125**	0.4800	**0.9224**	**0.8026**	**0.7523**
+PiFi	91.49	**87.11**	92.73	85.00	82.00
(Llama-3.1-70B)	0.9120	**0.4875**	0.9198	0.7784	0.761
+PiFi	91.05	86.56	91.90	85.14	81.65
(Qwen2.5-7B)	0.9080	0.4765	0.9117	0.8007	0.7562
+PiFi	91.12	86.27	91.69	83.75	80.74
(Qwen2.5-32B)	0.9086	0.4718	0.9096	0.7569	0.7463
+PiFi	91.38	86.75	92.21	84.51	82.45
(Qwen2.5-72B)	0.9115	0.4788	0.9150	0.7796	0.7450

Table 13: Experimental results comparing PiFi performance across various large language models. We evaluated PiFi with Llama-3.1 models (8B, 70B) and Qwen2.5 models (7B, 32B, 72B) across five downstream tasks. We report accuracy and F1-score in the upper row and lower row of each cell.

	SST-2	IMDB	Tweet (Sentiment)	Tweet (Offensive)	CoLA
BERT_{base}	89.41	85.10	86.90	83.15	80.77
	0.8907	0.4733	0.8620	0.7727	0.7296
+PiFi	91.50	87.09	**92.95**	**86.03**	82.07
(Llama-3.1-8B)	0.9125	0.4800	**0.9224**	**0.8026**	0.7523
+PiFi	91.49	87.11	92.73	85.00	82.00
(Llama-3.1-70B)	0.9120	0.4875	0.9198	0.7784	0.761
+PiFi-2 layers	**91.76**	**87.20**	92.95	85.66	**82.22**
(Llama-3.1-70B)	**0.9154**	**0.4901**	0.9217	0.8119	**0.7488**

Table 14: Experimental results of PiFi with multiple layers. This experiment compares the performance of using a single layer versus two layers from Llama-3.1-70B. We report accuracy in the upper row and F1-score in the lower row of each cell.

tuned LLMs does not necessarily lead to significant improvements for PiFi. Instead, it highlights that the intrinsic knowledge encoded within the LLMs is more crucial for PiFi's success than their capacity to follow human instructions.

B.6 Impact of Larger-Scale LLM on PiFi Performance

Previously, as analyzed in Section 4.8, we examined the performance of PiFi using the Qwen model across different scales (0.5B, 1.5B, and 7B parameters). However, it is necessary to further examine how performance changes when applying larger-scale LLM layers. To this end, we conducted experiments applying PiFi to various large-scale models, including the 8B and 70B versions of Llama-3.1 and the 7B, 32B, and 72B versions of Qwen2.5. We selected Qwen2.5 as it was newly released at the time of writing this paper, allowing us to evaluate PiFi's scalability using state-of-the-art models with cutting-edge performance.

	SST-2	IMDB	Tweet (Sentiment)	Tweet (Offensive)	CoLA
BERT_{base}	89.41	85.10	86.90	83.15	80.10
	0.8907	0.4733	0.8620	0.7727	0.7398
+PiFi	**91.50**	**87.09**	**92.95**	**86.03**	**82.07**
(Llama-3.1-8B)	**0.9125**	**0.4800**	**0.9224**	**0.8026**	**0.7523**
Single layer: First	80.09	77.63	80.14	78.59	68.12
(Llama-3.1-8B)	0.7970	0.4383	0.7913	0.6735	0.3812
Single layer: Last	80.67	78.31	80.28	80.47	69.31
(Llama-3.1-8B)	0.8010	0.4392	0.7939	0.6983	0.4076

Table 15: Experimental results on the usage of a single frozen L_{LLM} without SLM. For this experiment, we extracted the first and last layers of Llama-3.1-8B as L_{LLM}. We report accuracy and F1-score in the upper row and lower row of each cell.

As shown in Table 13, when comparing various large language models, performance improvements did not consistently correlate with model size. In particular, the use of the Llama-3.1-70B model layer in PiFi resulted in comparable or slightly lower performance than when using the 8B model layer. Similarly, the performance differences between Qwen2.5-7B, Qwen2.5-32B, and Qwen2.5-72B did not show a clear linear relationship with model size. We interpret this outcome as potentially influenced by a factor we refer to as layer knowledge density. Specifically, the 8B model comprises 32 layers, while the 70B model consists of 80 layers (Dubey et al., 2024). This suggests that a single layer in the 70B model may encapsulate relatively less dense knowledge compared to a single layer in the 8B model.

To mitigate the limitations imposed by this lower density, we conducted additional experiments incorporating the last two layers of the 70B model, with results presented in Table 14. This experiment directly compares the performance when using a single layer versus two layers from the 70B model. However, the observed performance improvement was marginal. This suggests that the selected layer combination may not be optimal, and alternative configurations could lead to greater performance gains. Nevertheless, the large number of possible combinations poses a challenge in identifying the most effective configuration. Future research will explore more efficient methods for identifying effective layer combinations, aiming to enhance PiFi's scalability while optimizing performance.

	FLOPs (GFLOPs)	GPU Memory (GB)
BERT$_{base}$	272.12	1.33
+ PiFi (Llama-3.1-8B)	279.3	2.27

Table 16: Efficiency-performance trade-off analysis results. This analysis was conducted by measuring FLOPs and GPU memory consumption.

B.7 Evaluating the Usage of a Single LLM Layer

To evaluate the performance of a single frozen LLM Layer without SLM, we conducted experiments by attaching classification heads to the first and last layers of Llama-3.1-8B.

As shown in the Table 15, there is a clear performance gap between the PiFi framework and the approach using a single LLM Layer. This demonstrates that PiFi effectively enhances performance through integration with an SLM (e.g., BERT$_{base}$).

Notably, the single LLM Layer shows limitations in achieving high performance independently, highlighting the necessity of combining it with an SLM for effective knowledge transfer. Furthermore, despite having fewer parameters than PiFi, the single frozen LLM Layer exhibited significantly degraded performance across all tasks. These findings reaffirm that PiFi can enhance the performance of the SLM by effectively integrating knowledge from a single LLM Layer while preserving the efficiency of the SLM.

B.8 Analyzing Efficiency-Performance Trade-offs in PiFi

As the number of parameters increases, it is necessary to analyze the trade-offs between efficiency and performance. To validate this, we measured FLOPs during the inference process for each model using the fvcore library (Meta, 2019) and GPU memory consumption using the PyTorch Profiler (Paszke et al., 2019a). The results are presented in Table 16.

The measurements revealed that PiFi adds approximately 2.6% to the baseline inference cost in terms of FLOPs. By using only the CLS token as input to the L_{LLM}, PiFi significantly reduces the sequence length processed in the L_{LLM}s to a single token, thereby greatly reducing computational overhead. Although GPU memory consumption showed a relatively larger increase, this trade-off is deemed reasonable for tasks that require performance improvement in environments where additional memory resources are available. Moreover, PiFi's modular design allows for flexible scalability according to system constraints, making it adaptable even in resource-constrained environments.

	SST-2	IMDB	Tweet (Sentiment)	Tweet (Offensive)	CoLA
BERT$_{base}$	89.41	85.10	86.90	83.15	80.10
	0.8907	0.4733	0.8620	0.7727	0.7398
+PiFi	**91.50**	87.09	**92.95**	**86.03**	**82.07**
(Llama-3.1-8B)	**0.9125**	0.4800	**0.9224**	**0.8026**	**0.7523**
BERT-base	79.48	**87.11**	81.05	27.89	69.31
+ ZEROGEN (GPT2-XL)	0.7885	**0.4875**	0.7748	0.2146	0.4076
BERT-base	81.56	68.26	87.99	25.29	30.69
+ ZEROGEN (Llama-3.1-8B)	0.8106	0.4343	0.8637	0.2061	0.2306

Table 17: Experimental results comparing PiFi and ZEROGEN for knowledge transfer. For ZEROGEN, 200,000 synthetic examples were generated per domain using GPT-2-XL and Llama-3.1-8B as data generators. We report accuracy and F1-score in the upper row and lower row of each cell.

	SST-2	IMDB	Tweet (Sentiment)	Tweet (Offensive)	CoLA
BERT$_{base}$	89.41	85.10	86.90	83.15	80.77
	0.8907	0.4733	0.8620	0.7727	0.7296
+PiFi	**91.50**	**87.09**	**92.95**	**86.03**	**82.07**
(Llama-3.1-8B)	**0.9125**	**0.4800**	**0.9224**	**0.8026**	**0.7523**
BERT$_{base}$+LoRA	82.93	75.22	81.37	73.07	69.98
	0.8207	0.4287	0.7677	0.4490	0.4252

Table 18: Experimental results comparing PiFi and LoRA for parameter-efficient fine-tuning. For LoRA, we used rank=8, scaling factor α=16, and dropout=0.1 with the same BERT$_{base}$ backbone. We report accuracy and F1-score in the upper row and lower row of each cell.

B.9 Comparison with Knowledge Distillation and Parameter-Efficient Fine-tuning Methods

We also conducted a comparison with ZEROGEN (Ye et al., 2022), which leverages synthetic data generated by an LLM to enhance the SLM as an alternative method for distilling the inherent knowledge of an LLM. Following the settings of prior studies, we generated 200,000 synthetic examples per domain using GPT-2-XL. Additionally, to account for the fact that ZEROGEN's performance depends on the quality of the data generator, we conducted more comprehensive experiments by generating additional synthetic examples using the latest model, Llama-3.1-8B.

As shown in the Table 17, ZEROGEN proved less effective than PiFi in most tasks, which we attribute to noise introduced during the synthetic

	SST-2	IMDB	Tweet (Sentiment)	Tweet (Offensive)	CoLA
BERT$_{base}$	89.41	85.10	86.90	83.15	80.77
	0.8907	0.4733	0.8620	0.7727	0.7296
+PiFi	**91.50**	**87.09**	**92.95**	**86.03**	**82.07**
(Llama-3.1-8B)	**0.9125**	**0.4800**	**0.9224**	**0.8026**	**0.7523**
	+2.09%p	+1.99%p	+6.05%p	+2.88%p	+1.97%p
BERT$_{base}$-10%	87.32	82.32	85.40	81.07	75.14
	0.8691	0.4541	0.8463	0.7420	0.6866
+PiFi-10%	**89.96**	**84.12**	**91.21**	**84.31**	**77.20**
(Llama-3.1-8B)	**0.8967**	**0.4688**	**0.9030**	**0.7807**	**0.6527**
	+2.64%p	+1.80%p	+5.81%p	+3.24%p	+2.06%p

Table 19: Experimental results of PiFi under limited training data conditions. We used only 10% of the total training data for each dataset, and report accuracy in the upper row of each cell and F1-score in the lower row.

data generation process. These results suggest that directly and efficiently utilizing the inherent knowledge of an LLM, rather than indirectly leveraging it through synthetic data generation, offers a more advantageous approach.

In addition to knowledge distillation methods, we compared PiFi with parameter-efficient fine-tuning using LoRA (Hu et al., 2022), which has gained popularity for adapting language models with minimal computational overhead. For our experiments, we applied LoRA to the BERT-base backbone using standard settings (rank=8, scaling factor α=16, dropout=0.1) and evaluated its performance on the same set of downstream tasks.

As shown in Table 18, PiFi consistently outperforms LoRA across all downstream tasks. This demonstrates that PiFi achieves superior performance improvements by effectively integrating LLM layers, enabling direct transfer of rich linguistic knowledge and generalization capabilities encoded in large-scale models to SLMs.

B.10 Performance Evaluation of PiFi under Limited Training Data

To assess whether the PiFi framework operates effectively in a limited training data environment, experiments were conducted using only 10% of the total training data for each dataset. Specifically, the following sample counts were used: SST-2 (6,920 $\rightarrow$ 692), IMDB (20,000 $\rightarrow$ 2,000), Tweet for sentiment classification (45,615 $\rightarrow$ 4,562), Tweet for offensive classification (11,916 $\rightarrow$ 1,192), and CoLA (5,536 $\rightarrow$ 554).

As shown in Table 19, even when trained on restricted data, the model with PiFi consistently outperformed the BERT$_{base}$. In particular, the SST-2, Tweet for offensive classification, and CoLA datasets exhibited relatively greater performance improvements compared to when the full dataset was used, indicating that PiFi can deliver significant benefits even under constrained data conditions. In contrast, the IMDB and Tweet for sentiment classification datasets have a considerably larger amount of total data (20,000 and 45,615 samples, respectively), so even with only 10% of the data, a sufficient number of training samples was available. Consequently, the relative performance gains were smaller; however, this can be viewed as a positive example of PiFi's flexible applicability even without large-scale data. These results underscore that PiFi offers stable and consistent performance advantages in limited training data environments, thereby enhancing its potential for real-world low-resource scenarios.

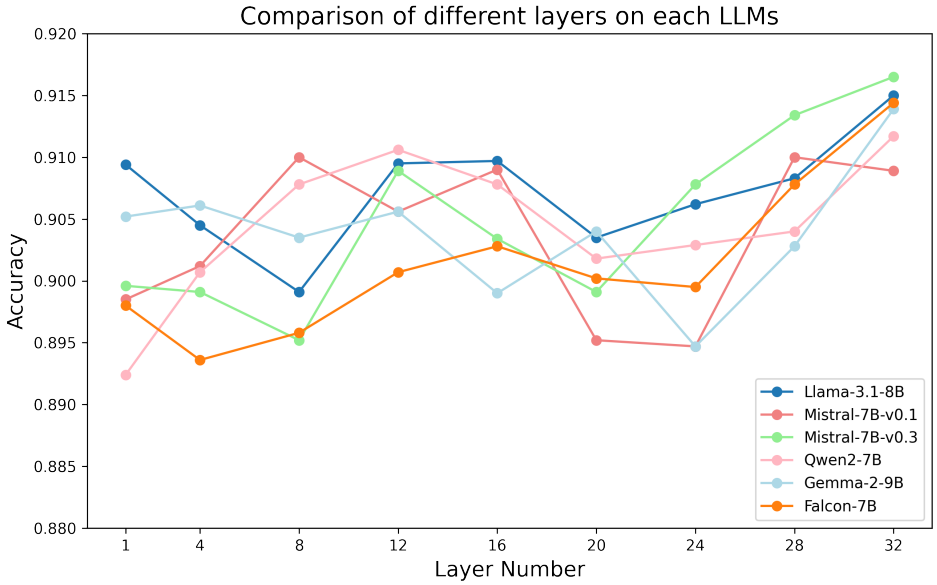

Figure 2: Comparison of the performance of PiFi models with different position of L_{LLM} on SST-2 dataset.

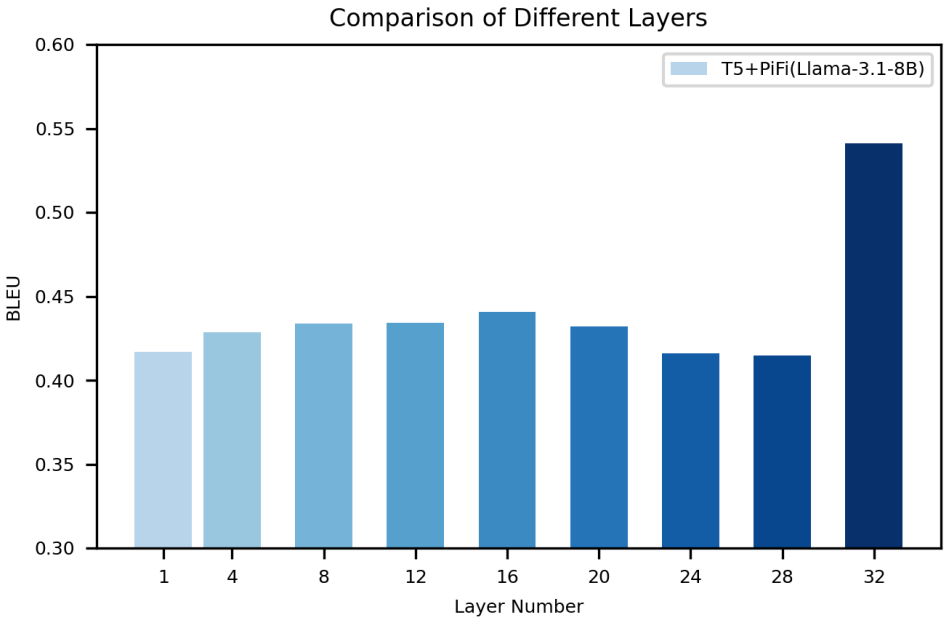

Figure 3: Comparison of the performance of PiFi models with different position of L_{LLM} on Multi30K dataset.

	SLM Params(M)	L_{LLM} Params(M)	L_{in} Params(M)	L_{out} Params(M)	*Head* Params(M)
BERT$_{base}$ **+ PiFi (Llama-3.1-8B)**	109.482	218.112	3.146	3.146	0.592
RoBERTa$_{base}$ **+ PiFi (Llama-3.1-8B)**	124.646				
ELECTRA$_{base}$ **+ PiFi (Llama-3.1-8B)**	108.892				
DeBERTa$_{base}$ **+ PiFi (Llama-3.1-8B)**	138.602				
DeBERTa-V3$_{base}$ **+ PiFi (Llama-3.1-8B)**	183.832				
BERT$_{base}$ **+ PiFi (Mistral-7B-v0.1)**	109.482				
BERT$_{base}$ **+ PiFi (Mistral-7B-v0.3)**	109.482				
BERT$_{base}$ **+ PiFi (Qwen2-0.5B)**	109.482	14.912	0.688	0.688	
BERT$_{base}$ **+ PiFi (Qwen2-1.5B)**		46.798	1.180	1.180	
BERT$_{base}$ **+ PiFi (Qwen2-7B)**		233.058	3.146	3.146	
BERT$_{base}$ **+ PiFi (Gemma-2-9B)**		198.195	3.146	3.146	
BERT$_{base}$ **+ PiFi (Falcon-7B)**		207.070	3.146	3.146	
BERT$_{base}$ **+ PiFi (Llama-3.1-70B)**		855.654	6.291	6.291	
T5$_{base}$ **+ PiFi (Llama-3.1-8B)**	222.904	218.112	3.146	3.146	-
BART$_{base}$ **+ PiFi (Llama-3.1-8B)**	139.420				

Table 20: The parameters of each module for PiFi, across various setup regarding SLMs and LLMs.