When MOE Meets LLMs: Parameter Efficient Fine-tuning for Multi-task Medical Applications

Qidong Liu
Xi'an Jiaotong University &
City University of Hong Kong
Xi'an, China
liuqidong@stu.xjtu.edu.cn

Xian Wu ✉
Jarvis Research Center,
Tencent YouTu Lab
Shenzhen, China
kevinxwu@tencent.com

Xiangyu Zhao ✉
City University of Hong Kong
Hong Kong, Hong Kong
xianzhao@cityu.edu.hk

Yuanshao Zhu
Southern University of Science and Technology &
City University of Hong Kong
Shenzhen, China
zhuys2019@mail.sustech.edu.cn

Derong Xu
University of Science and Technology of China &
City University of Hong Kong
Hefei, China
derongxu@mail.ustc.edu.cn

Feng Tian ✉
Xi'an Jiaotong University
Xi'an, China
fengtian@mail.xjtu.edu.cn

Yefeng Zheng
Jarvis Research Center,
Tencent YouTu Lab
Shenzhen, China
yefengzheng@tencent.com

ABSTRACT

The recent surge in Large Language Models (LLMs) has garnered significant attention across numerous fields. Fine-tuning is often required to fit general LLMs for a specific domain, like the web-based healthcare system. However, two problems arise during fine-tuning LLMs for medical applications. One is the task variety problem, which involves distinct tasks in real-world medical scenarios. The variety often leads to sub-optimal fine-tuning for data imbalance and seesaw problems. Besides, the large amount of parameters in LLMs leads to huge time and computation consumption by fine-tuning. To address these two problems, we propose a novel parameter efficient fine-tuning framework for multi-task medical applications, dubbed as **MOELoRA**. The designed framework aims to absorb both the benefits of mixture-of-expert (MOE) for multi-task learning and low-rank adaptation (LoRA) for parameter efficient fine-tuning. For unifying MOE and LoRA, we devise multiple experts as the trainable parameters, where each expert consists of a pair of low-rank matrices to retain the small size of trainable parameters. Then, a task-motivated gate function for all MOELoRA layers is proposed, which can control the contributions of each expert and produce distinct parameters for various tasks. We conduct experiments on a multi-task medical dataset, indicating MOELoRA outperforms the existing parameter efficient fine-tuning methods. The code is available online [1].

CCS CONCEPTS

• **Applied computing** → **Health informatics**.

KEYWORDS

Medical Application; Large Language Model; Multi-task Learning;

ACM Reference Format:
Qidong Liu, Xian Wu ✉, Xiangyu Zhao ✉, Yuanshao Zhu, Derong Xu, Feng Tian ✉, and Yefeng Zheng. 2024. When MOE Meets LLMs: Parameter Efficient Fine-tuning for Multi-task Medical Applications. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '24), July 14–18, 2024, Washington, DC, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3626772.3657722

1 INTRODUCTION

Due to the impressive capabilities in language understanding and generation, the Large Language Models (LLMs), such as ChatGPT [25] and ChatGLM [55], have gained extensive interest from both academia and industry. Many efforts have been devoted to investigating the potential applications of LLMs across various domains [9, 12, 43]. One particularly suitable domain for LLMs is the medical domain, as the application of LLMs can benefit both patients and doctors. For patients, the LLM-enabled online Chatbot can provide convenient access to medical knowledge; For doctors, the LLM-enabled Clinical Decision Supporting Systems (CDSS) can relieve heavy workloads and improve diagnosis efficiency.

However, the majority of LLMs are trained for general purposes and are not customized for the medical domain. As a result, the

✉ Corresponding authors.

SIGIR '24, July 14–18, 2024, Washington, DC, USA

ACM ISBN 979-8-4007-0431-4/24/07
https://doi.org/10.1145/3626772.3657722

[1] https://github.com/Applied-Machine-Learning-Lab/MOELoRA-peft

general LLMs often fall short in medical tasks due to a lack of specialized medical knowledge [53]. To empower LLMs with medical capabilities, a straightforward manner is to fine-tune LLMs with medical tasks. For large LLMs with more than 100 billion parameters, they are usually closed-source and extremely costly for fine-tuning [30]. Therefore, in this paper, we focus on the open-source LLMs and fine-tuning them with medical knowledge and clinical tasks [42, 53].

Fine-tuning LLMs for the medical domain usually involves two primary challenges: (i) **Task Variety Problem**: In real-world clinics, LLMs can be applied to a large range of tasks, like doctor recommendation [61], diagnosis prediction [32], medicine recommendation [57, 62], medical named entity recognition [33, 59], clinical report generation [29] and *etc.* Since the input and output of these tasks are quite different, it is difficult to fine-tune a unified model for all tasks. Given the diversity of these tasks, fine-tuning a single model for each specific task is feasible but demands extensive expertise and labor. An integrated multi-task learning framework could potentially address this issue. However, much of the existing research on LLMs, as seen in studies like [37, 42], predominantly centers on medical dialogue. Such over-attention ignores the variety of tasks, resulting in multi-task fine-tuning remains underexplored. (ii) **High Tuning Cost**: While fine-tuning all model parameters was a standard approach during the era of Bert [17], it becomes challenging for LLMs due to their sheer size. The vast number of parameters in LLMs can lead to prohibitive time and computational expenses in practice [52]. As such, there is an urgent need for parameter efficient fine-tuning methodologies. To address these two challenges, the community urgently calls for developing a multi-task parameter efficient fine-tuning framework for LLM-driven medical applications.

For the task variety problem, several multi-task learning frameworks have been proposed [5, 8, 26, 46, 58]. A standout among these is Mixture-of-Experts (MOE) [35], which designs multiple separate experts to learn task-shared and -specific knowledge, and integrates a gate function to modulate the contributions of each expert. While existing frameworks adeptly consolidate multiple tasks for classical neural network architectures, they are primarily compatible with full fine-tuning, which is associated with high tuning costs. Correspondingly, the emergence of parameter efficient fine-tuning (PEFT) methods has offered a potential solution to the problem of high tuning costs. These methods typically tune a limited number of parameters, keeping the pre-trained LLMs parameters frozen. However, the existing PEFT is limited to fine-tuning either multiple sets of parameters for each task separately or a singular set across all tasks. Though separate training can fit each task well, this strategy is laborious and lacks task-shared knowledge. While fine-tuning a set of parameters is feasible, it may hurt performance due to issues such as data imbalance and seesaw effects [5, 20]. For illustration, we analyze the data distribution of a Chinese medical dataset, PromptCBLUE [63], in Figure 1. Our analysis reveals significant disparities: while some tasks boast nearly $5,000$ samples, others have fewer than $2,000$. This imbalance can skew the uniquely fine-tuned parameters towards tasks with more samples, inadvertently undermining the performance on tasks with fewer samples. *Therefore, parameter efficient fine-tuning of separate parameters for multi-task by a unique training process can alleviate both problems.*

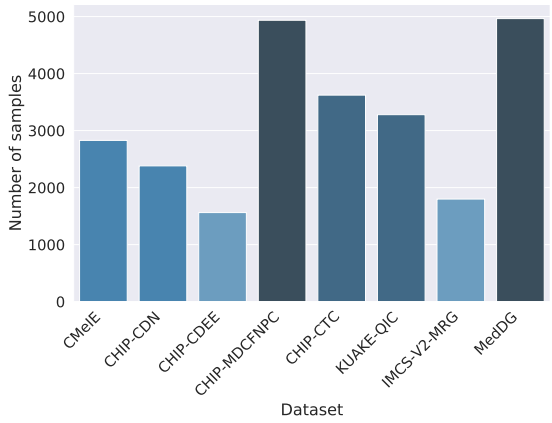


Figure 1: The illustration for data imbalance problem.

To address the challenges of task variety and high tuning costs, we propose a unified parameter efficient fine-tuning framework to learn separate parameters for various tasks, dubbed as MOELoRA. Our framework follows the basic scheme of LoRA for the parameter efficiency, *i.e.,* only fine-tuning small size of parameters parallel to the dense layers in LLMs. However, as discussed previously, existing unified LoRA fine-tuning faces the challenge of **a singular set of parameters across all tasks**. Thus, in our approach, we first design several experts as the trainable part rather than a singular pair of low-rank matrices. On the one hand, inspired by MOE [35], separate experts can help learn task-specific knowledge under one unique training process. On the other hand, such a design gives the chance to produce several distinct sets of parameters. Besides, for parameter efficiency, we devise each expert as two low-rank matrices. Then, to learn separate sets of parameters for each task, we propose a task-motivated gate function. In specific, the gate function absorbs the task identity and outputs corresponding expert weights. By the expert weights for one specific task and the parameters of multiple experts, we can get the unique updated parameters for this task.

In summary, the contributions of this paper are as follows:

- We introduce **MOELoRA**, a novel multi-task PEFT framework that combines the strengths of both MOE and LoRA. Additionally, we design a task-motivated gate function to facilitate the tuning of distinct parameter sets for each task.
- We conduct comprehensive experiments on a public multi-task Chinese medical dataset, with the results underscoring the superiority of the proposed MOELoRA framework.
- To our knowledge, this research represents the first endeavor to delve into multi-task parameter efficient fine-tuning techniques for LLM-driven medical applications.

2 PRELIMINARY

2.1 LLMs for Medical Applications

Intelligent medical systems have become increasingly prevalent in contemporary web-based healthcare settings. Numerous studies have sought to standardize medical tasks by defining consistent

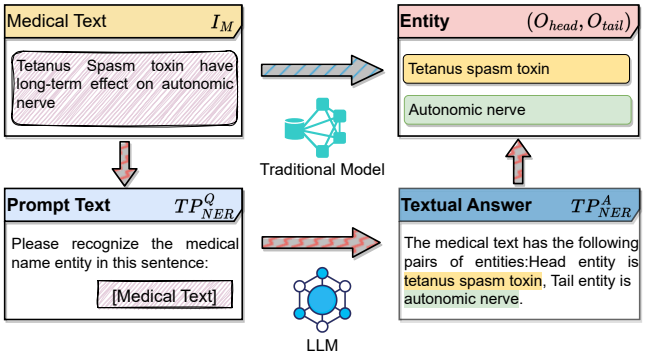


Figure 2: The medical name entity recognition example for illustration of how to use LLMs to complete medical tasks.

input and output patterns, thereby streamlining the model design process. As the example of medical named entity recognition (NER) [33, 59] illustrated in Figure 2, traditional models typically process medical texts, denoted as I_M, to produce entities O_{head} and O_{tail}. However, the integration of LLMs into medical tasks introduces a distinct paradigm. Given that both the input and output of LLMs are typically linguistic in nature, there is a necessity to reformulate medical tasks to be compatible with LLMs.

To adapt medical tasks for LLMs, we need to restructure both the input and output patterns. **Input Modification**: We incorporate instruction templates into the original medical texts to guide LLMs in executing the relevant tasks [56]. Taking medical NER as an example, as depicted in Figure 2, we employ the template: *Please recognize the medical name entity in this sentence: "[Medical Text]"*, where "[Medical Text]" serves as a placeholder for the raw medical text I_M. **Output Modification**: Instead of using plain targets, we format LLMs outputs into linguistic texts. For the NER task, the recognized head entity O_{head} and tail entity O_{tail} are integrated into the template: "*The medical text has the following pairs of entities: head entity is [head entity] and tail entity is [tail entity]*". For ease of reference, we label the input and output instruction templates for NER as TP^Q_{NER} and TP^A_{NER}, respectively. With these modifications in place, the process by which LLMs undertake the NER task can be described as follows:

$$I_M \rightarrow TP^Q_{NER}(I_M) \xrightarrow{LLM} TP^A_{NER}(O_{head}, O_{tail}) \rightarrow O_{head}, O_{tail} \tag{1}$$

After the task reformulation for LLMs, we can use the purely lingual data to fine-tune the foundation large language models, such as LlaMA [40], ChatGLM [7] and etc. The fine-tuned model completes the medical tasks by generating the regulated answers.

2.2 Multi-task Fine-tuning

As previously mentioned, medical applications often encompass a variety of tasks, such as name entity recognition, medical inquiry, etc. Our goal is to fine-tune LLMs to gain robust performance for each task and thus can also benefit the whole healthcare system. For multi-task fine-tuning, we consider a set of medical tasks represented as $\mathbb{T} = \{\mathcal{T}_1, \ldots, \mathcal{T}_j, \ldots, \mathcal{T}_M\}$. The structured data corresponding to task $\mathcal{T}_j$ can be represented as $\mathcal{D}_j = \{(LI_k^{\mathcal{T}_j}, LO_k^{\mathcal{T}_j})\}_{k=1}^{|\mathcal{D}_j|}$, where LI and LO represent the template-formatted linguistic input and output, respectively. For the sake of brevity, we omit the superscript $\mathcal{T}_j$ in subsequent discussions. Assuming the parameters of LLMs are represented by Φ, the multi-task fine-tuning challenge can be articulated as: Given the dataset of all medical tasks $\mathcal{D} = \{\mathcal{D}_j\}_{j=1}^M$, optimize the parameters Φ of LLMs to ensure optimal performance across each task $\mathcal{T}_j$.

Since the data from diverse tasks are standardized into a consistent linguistic format, we straightforwardly employ the conditional language modeling objectives [7] for all training instances. Furthermore, with the intent to assimilate shared medical knowledge and be free from the labor of fine-tuning for various tasks separately, data from all tasks are incorporated into the unique optimization process. Consequently, the objective function for multi-task fine-tuning can be formulated as follows:

$$\max_{\Phi} \sum_{j \in [M]} \sum_{(x,y) \in \mathcal{D}_j} \sum_{t=1}^{|y|} log(P_\Phi(y_t | x, y_{\leq t})) \tag{2}$$

3 METHOD

In this section, we provide a comprehensive description of our proposed framework. We begin with an overview of the proposed method. Then, the devised MOELoRA and task-motivated gate are addressed. Finally, we detail fine-tuning and inference processes.

3.1 Overview

Figure 3 provides a visual representation of the parameter efficient fine-tuning and inference process of LLMs using MOELoRA. In the realm of parameter efficient fine-tuning, LoRA [14] introduces the concept of training only two low-rank matrices as a substitute for updates in dense layers. Building on this, our approach integrates MOELoRA layers into each dense layer, enabling them to acquire *keys*, *queries*, and *values*, as well as facilitating the feed-forward network (*FNN*). In Figure 3, we take *FNN* as the example for illustration. A significant advantage of our method is that we only fine-tune the parameters of the MOELoRA layers for various tasks, keeping the rest parameters of the original LLMs frozen.

Furthermore, each MOELoRA layer incorporates multiple experts, which are designed to capture diverse knowledge across various medical tasks, a concept we will delve deeper into in Section 3.2. Then, we introduce a task-motivated gate function to ensure that unique parameter sets are learned for each task. This function determines the contribution weights of experts across all MOELoRA layers, enabling the generation of distinct updated parameters tailored to different tasks. In particular, we employ a single gate function for all MOELoRA layers, rather than having a one-to-one correspondence between gates and MOELoRA layers. For the fine-tuning process, we update the MOELoRA layers on the mixture of the data from all tasks. Then, the MOELoRA can derive distinct fine-tuned weights for each task during the inference.

3.2 MOELoRA

Low-rank Adaptation (LoRA) [14] has demonstrated both its effectiveness and efficiency in fine-tuning LLMs. It is inspired by the low intrinsic dimension characteristic [1], which reformulates the parameter fine-tuning process in LLMs as a low-rank decomposition. Specifically, the equation $\mathbf{W}_0 + \Delta\mathbf{W} = \mathbf{W} + \mathbf{BA}$ captures this decomposition. Here, $\mathbf{W}_0 \in \mathbb{R}^{d_{in} \times d_{out}}$ represents the

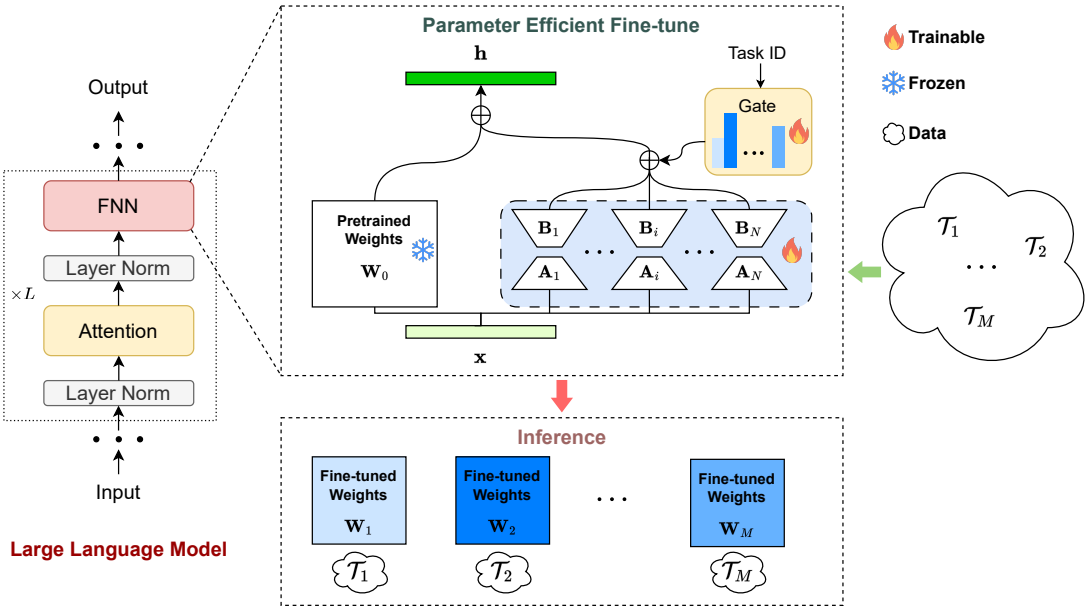


Figure 3: The overview of parameter efficient fine-tuning and inference process using MOELoRA.

parameter matrix of the pre-trained LLMs, while $\Delta\mathbf{W} \in \mathbb{R}^{d_{in} \times d_{out}}$ denotes the matrix updated during fine-tuning. The matrices $\mathbf{B} \in \mathbb{R}^{d_{in} \times r}$ and $\mathbf{A} \in \mathbb{R}^{r \times d_{out}}$ are low-rank and trainable. Given this setup, the forward process of a linear layer paired with a LoRA layer can be expressed as:

$$\mathbf{h} = \mathbf{W}_0\mathbf{x} + \frac{\alpha}{r} \cdot \Delta\mathbf{W}\mathbf{x} = \mathbf{W}_0\mathbf{x} + \frac{\alpha}{r} \cdot \mathbf{B}\mathbf{A}\mathbf{x} \tag{3}$$

where $\mathbf{x}$ represents the input vector of dimension d_{in}, and $\mathbf{h}$ is the output vector with dimension d_{out}. The rank of the trainable low-rank matrices is denoted by r, which determines the number of trainable parameters. The constant hyper-parameter α facilitates the tuning of rank r [14]. During the LoRA fine-tuning process, all parameters in LLMs remain frozen. Only the low-rank matrices, $\mathbf{A}$ and $\mathbf{B}$, undergo fine-tuning. Given that $r \ll d_{in}$ and $r \ll d_{out}$, the combined number of parameters in $\mathbf{A}$ and $\mathbf{B}$ is significantly smaller than the ones in $\mathbf{W_0}$. Such characteristics result in achieving parameter efficiency for the fine-tuning process.

However, the integral parameters are fine-tuned for all tasks in the original LoRA, which causes difficulty in learning the various aspects of medical knowledge. A potential solution is to segment the entire parameter set into several parts and derive various combinations for various tasks. The Mixture-of-Expert (MOE) model [35] suggests employing multiple expert networks to capture different facets of multi-task information, aligning with the combination concept. This insight leads us to design MOELoRA, which seamlessly integrates the advantages of both LoRA and MOE. To harmonize the distinct forward processes of LoRA and MOE, we introduce a set of experts, denoted as $\{E_i\}_{i=1}^{N}$, to learn the updated matrix $\Delta\mathbf{W}$. As MOELoRA fine-tunes the experts using data from all tasks, it inherently captures shared task knowledge. Moreover, to maintain a compact parameter size, every expert in the MOELoRA layer is constructed as two decomposed low-rank matrices. Based on this structure, the forward process of a linear layer paired with a MOELoRA layer for samples from task $\mathcal{T}_j$ is expressed as:

$$\begin{aligned}\mathbf{h}_j &= \mathbf{W}_0\mathbf{x}_j + \frac{\alpha}{r} \cdot \Delta\mathbf{W}_j\mathbf{x}_j \\ &= \mathbf{W}_0\mathbf{x}_j + \frac{\alpha}{r} \cdot \sum_{i=1}^{N} \omega_{ji} \cdot E_i(\mathbf{x}_j) \\ &= \mathbf{W}_0\mathbf{x}_j + \frac{\alpha}{r} \cdot \sum_{i=1}^{N} \omega_{ji} \cdot \mathbf{B}_i\mathbf{A}_i\mathbf{x}_j\end{aligned} \tag{4}$$

where $\mathbf{h}_j$ and $\mathbf{x}_j$ represent the input and output of intermediate LLM layers for samples from $\mathcal{T}_j$. The matrices $\mathbf{B}_i \in \mathbb{R}^{d_{in} \times \frac{r}{N}}$ and $\mathbf{A}_i \in \mathbb{R}^{\frac{r}{N} \times d_{out}}$ form the expert E_i. The hyper-parameter N denotes the number of experts in MOELoRA, and for each expert, the rank of matrices A and B is $\frac{r}{N}$. In Equation (4), the term ω_{ji} modulates these contribution weights for task $\mathcal{T}_j$. This weight is determined by our proposed gate function, which will be detailed later.

Here, we will discuss the number of trainable parameters for LoRA and MOELoRA. In terms of LoRA, the two low-rank matrices $\mathbf{B} \in \mathbb{R}^{d_{in} \times r}$ and $\mathbf{A} \in \mathbb{R}^{r \times d_{out}}$ contain all trainable parameters. Thus, the number of trainbale parameters of LoRA is $d_{in} \times r + r \times d_{out} = r \times (d_{in} + d_{out})$. As for MOELoRA, there are N trainable experts and each expert owns $\frac{r}{N} \times (d_{in} + d_{out})$, so total number is calculated as $N \times \frac{r}{N} \times (d_{in} + d_{out}) = r \times (d_{in} + d_{out})$. In conclusion, MOELoRA has the same number of trainable parameters as LoRA, which indicates high efficiency.

3.3 Task-Motivated Gate Function

In this section, we detail the intricacies of our task-motivated gate function. As previously emphasized, the contribution of each expert should be tailored to specific tasks. To regulate these contributions, we introduce a gate function. Since these weights are inherently task-specific, our gate function is designed to take the task identity as input. To facilitate this, we employ a task embedding matrix, denoted as $\mathbf{E} \in \mathbb{R}^{|\mathbb{T}| \times d_T}$, where d_T represents

the dimension of the task embedding. Upon identifying a task $\mathcal{T}_j$, we extract the j-th column of $\mathbf{E}$, which serves as the representation vector for that task, symbolized as $\mathbf{e}_j \in \mathbb{R}^{d_T}$. To determine the contribution weights for task $\mathcal{T}_j$, we apply a linear transformation. This computation is captured by the following equation:

$$\boldsymbol{\omega}_j = \text{Softmax}(\mathbf{W}_T \mathbf{e}_j) \tag{5}$$

Here, $\boldsymbol{\omega}_j \in \mathbb{R}^N$ represents the contribution weight vector tailored for task $\mathcal{T}_j$. The transformation matrix is denoted as $\mathbf{W}_T \in \mathbb{R}^{N \times d_T}$. To prevent any disproportionately large weights, we employ a softmax operation to normalize the contribution weights.

The gate mentioned in Equation (5) is naturally a dense design to combine all of the experts. Recently, some works [34, 35] have focused on another form of gate, *i.e.,* sparse gate. It has the benefit of alleviating the optimization interference across various tasks [11]. Thus, we also design a sparse version of the task-motivated gate to explore which design is more effective. The devised sparse gate is formulated as follows:

$$\boldsymbol{\omega}_j = \text{Softmax}(\text{Top}(\mathbf{W}_T \mathbf{e}_j, K)) \tag{6}$$

$$\text{Top}(\mathbf{x}, K) = \begin{cases} x_i & \text{if } x_i \text{ in top } K \text{ elements of } \mathbf{x}. \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

While the conventional design of MOE directly feeds the input vector $\mathbf{x}$ into the gate function, our approach diverges. We exclusively input the task identity into the gate function, as Figure 3 shows, aiming to yield a unique set of model parameters for each task. To illustrate, if one wishes to recover the fine-tuned parameters for task $\mathcal{T}_j$, the process can be articulated as:

$$\begin{aligned} \mathbf{W}_j &= \mathbf{W}_0 + \frac{\alpha}{r} \cdot \Delta \mathbf{W}_j \\ &= \mathbf{W}_0 + \frac{\alpha}{r} \cdot \sum_{i=1}^{N} \omega_{ji} \cdot E_i \\ &= \mathbf{W}_0 + \frac{\alpha}{r} \cdot \sum_{i=1}^{N} \omega_{ji} \cdot \mathbf{B}_i \mathbf{A}_i \end{aligned} \tag{8}$$

If the gate function is driven by the input vector $\mathbf{x}$, the weight vector would differ across samples. This means each sample would possess its unique $\boldsymbol{\omega}_j$, leading to a sample-specific fine-tuned parameter matrix. This design would render the parameters non-recoverable on a per-task basis. The ability to recover parameters for each task offers two primary advantages: 1) *Customization for Task*: Each task is fine-tuned with a set of parameters, which can help learn more task-specific information and alleviate the problem of data imbalance. 2) *Efficiency in Inference*: The recovered, fine-tuned LLMs exhibit reduced inference latency. This is attributed to the elimination of the need for the additional forward computation associated with the MOELoRA layer.

3.4 Fine-tune and Inference

In this section, we refer to the fine-tuning and inference process of MOELoRA. For better readability, we also conclude the whole procedure in Algorithm 1.

Fine-tuning. We first configure the MOELoRA according to the specified layers in LLMs and several hyper-parameters (line 1-3). Then, for the parameter efficient fine-tuning, all pre-trained parameters in LLMs (line 4) are frozen. During the fine-tuning, we randomly sample a batch of data from all tasks iteratively, instead of grouping the samples from the same task into one batch as some multi-task researches [36, 39] do. We choose the random sampling for batch by the performance comparison in experiments. Using the batch of data, we can conduct the forward process and compute the loss for fine-tuning (line 6-7). For parameter update, it is worth noting that we only fine-tune the parameters of MOELoRA and task-motivated gate function, *i.e.,* $\{\mathbf{A}_i, \mathbf{B}_i\}_{i=1}^N$ and $\{\mathbf{E}, \mathbf{W}_T\}$.

Inference. As discussed in Section 3.3, the MOELoRA can recover the fine-tuned parameter matrices for each task by Equation (8). For inference, we first recover the fine-tuned parameters in LLMs for each task (line 10-13), which indicates that each task has its own LLMs parameters. Then, we can apply the corresponding LLMs to complete the specified task.

Algorithm 1 Fine-tuning and inference process of MOELoRA

1: Indicate the LLMs and the layers that need MOELoRA fine-tuning.
2: Indicate the rank value r and scale value α.
3: Indicate the number of experts N of MOELoRA.

Fine-tuning Process

4: Freeze all parameters in pre-trained LLMs, *e.g.,* $\mathbf{W}_q$, $\mathbf{W}_k$, $\mathbf{W}_v$.
5: **for** a batch of samples B in $\mathcal{D}$ **do**
6: Conduct forward process for LLMs accompanied with MOELoRA by Equation (4).
7: Compute the loss function by Equation (2).
8: Update the parameters of MOELoRA $\{\mathbf{A}_i, \mathbf{B}_i\}_{i=1}^N$ and the parameters of gate function $\{\mathbf{E}, \mathbf{W}_T\}$
9: **end for**

Inference Process

10: **for** $\mathcal{T}_j$ in $\mathbb{T}$ **do**
11: Calculate the contribution weights $\boldsymbol{\omega}_j$ for each experts by Equation (5).
12: Recover the MOELoRA fine-tuned parameters by Equation (8) for each task.
13: **end for**
14: For specific task $\mathcal{T}_j$, apply the corresponding parameters of LLMs for prediction.

4 EXPERIMENT

In this section, we seek to address the following Research Questions (**RQ**):

- **RQ1**: How does MOELoRA compare to other parameter-efficient fine-tuning strategies and cross-task generalization methods in terms of performance?
- **RQ2**: What impact do the MOE architecture and the gate function have on the fine-tuning process? How do different training strategies influence MOELoRA's performance?
- **RQ3**: How do the number of experts and the rank of MOELoRA influence performance outcomes?
- **RQ4**: Is the proposed MOELoRA efficient in the process of fine-tuning and inference?

Table 1: The brief description and statistics of the dataset PromptCBLUE.

Task	Description	# Train	# Validation	# Test
CMeIE	Name Entity Recognition	2,828	600	600
CHIP-CDN	Normalization	2,381	600	600
CHIP-CDEE	Attribute Extraction	1,562	600	600
CHIP-MDCFNPC	Clinic Entity Discovery	4,935	600	600
CHIP-CTC	Medical Text Classification	3,622	1,100	1,100
KUAKE-QIC	Query Intention	3,279	660	660
IMCS-V2-MRG	Report Generation	1,799	600	600
MedDG	Doctor Dialogue	4,964	600	600

- **RQ5**: Are the experts specialized in capturing specific aspects of knowledge for various tasks?

4.1 Experimental Settings

4.1.1 ***Dataset.*** Our experiments are conducted on the PromptCBLUE dataset[2] [63], a multi-task Chinese medical dataset recently made available on the Tianchi Competition Platform[3]. This dataset encompasses 16 distinct medical tasks, each of which has been transformed into pure text format using specific prompts, ensuring compatibility with LLMs. To the best of our knowledge, PromptCBLUE is the only multi-task medical dataset tailored for LLMs. Due to computational constraints, we have chosen 8 tasks at random for our experiments. For pre-processing, we eliminate duplicate samples in the original dataset. Since the test set used in the competition remains unreleased, we opt to use the development set as our test set. Then, the validation set for the experiment is derived from the training set in competition, with its size matching that of the test set. The statistics of the dataset are concluded in Table 1.

4.1.2 ***Baselines.*** In our experiments, we benchmark against four distinct groups of baselines:

LLMs without Fine-tuning: We employ In-Context Learning [6] to guide LLMs in accomplishing the relevant tasks.

- **ChatGPT** [3]. ChatGPT is one of the most popular LLMs. To inspire task-relevant ability, we randomly sample 3 to 10 input-output pairs from the training data of one specific task to get the demonstration, filling the input length to the maximum.
- **Huatuo** [42]. Huatuo first constructs a Chinese medical instruction dataset by a medical database. In our experiment, we use the version of ChatGLM-6B for fair comparison and the same in-context learning method as the ChatGPT baseline.

LLMs with Fine-tuning: This group encompasses various strategies of the fine-tuning.

- **P-Tuning** [25]. *P-Tuning* designs a trainable prompt encoder to produce continuous prompt vectors, which are inserted into the input sequence. We implement it by fine-tuning the prompt encoder on the data of all tasks.
- **LoRA (Full)** [14]. LoRA designs two low-rank matrices as the trainable parameters for dense layers while freezing all parameters of pre-trained LLMs. *LoRA (Full)* trains a unique set of LoRA parameters for all tasks.
- **LoRA (Single)** [14]. We implement *LoRA (Single)* by separately training LoRA for each task. For the time and resource limitation, we adopt the same set of hyper-parameters for all tasks and select the model according to the best average score.
- **LoRA (Full+TP)** [14]. We add simple task demonstration to input texts, which aims to prompt LLMs with the distinctions between tasks. As for the implementation, we conduct the same training process as *LoRA (Full)*.

Model Editing: One model editing method can be adapted to multi-task fine-tuning for LLMs.

- **Task-Arithmetic** [16]. *Task-Arithmetic* defines a novel task vector, which can be operated for merging or unlearning tasks. However, it is for full fine-tuning, so we modify the method for fair comparison: the task vector is calculated by $\tau_t = \mathbf{B}_t \cdot \mathbf{A}_t$, where $\mathbf{A}$ and $\mathbf{B}$ are the low-rank matrices in LoRA layers. According to the adding operation for multi-task, we add all task vectors together and adjust the scale factor on the validation set.

Cross-task Generalization: To assess the applicability of cross-task generalization to multi-task fine-tuning, we also evaluate two recent approaches, *LoRAHub* [15] and *MoLoRA* [54].

- **LoRAHub** [15]. *LoRAHub* proposes an assembling method to compose LoRA parameters fine-tuned on source tasks and seek the generalization to unseen target tasks. In the experiment, we LoRA fine-tune each task. Then, use the validation of one specified task to learn the composing weight and test the performance for this task.
- **MoLoRA** [54]. *MoLoRA* is a relatively recent work, which adopts the MOE structure to the LoRA. However, the gate in MoLoRA takes the intermediate embedding of the tokens to derive expert weights. In our experiments, we adapt it to our multi-task setting, *i.e.,* training and testing on same set of tasks.

4.1.3 ***Implementation Details.*** Our experiments are simulated by PyTorch 1.12.0 and Python 3.9.5. We run the code on Tesla V100 32G GPUs for acceleration. The LLM ChatGLM-6B [7], recognized for its proficiency in Chinese language processing, serves as the foundational model for fine-tuning. For all LoRA fine-tuning baselines and the proposed MOELoRA, we designate trainable layers for the layers identified as "query_key_value", "dense", "dense_h_to_4h", and "dense_4h_to_h". The maximum input and output lengths were configured to $1,024$ and 196, respectively. We set the batch size to 64 with a maximum of $8,000$ training steps. The LoRA rank r was fixed at 16, with a LoRA dropout $\alpha = 0.1$. For MOELoRA, the number of experts is set to 8. K is searched from 1 to 7 for the sparse gate version of MOELoRA, and we find 2 is the optimal. Besides, during the testing, we set the temperature as 0.95 for generation. Our MOELoRA implementation[4] is compatible with the PEFT package[5], which can facilitate easier adoption and utilization of the proposed MOELoRA.

4.1.4 ***Evaluation Metrics.*** For our evaluations, we employ a variety of metrics tailored to the nature of each task. For example, CMeIE is a task of name entity recognition (NER), where there are too many entity classes (**1262** classes for CMeIE), so we apply commonly used Micro-F1 for this task [47]. CHIP-CDN (**579** classes), CHIP-CDEE (**998** classes) and CHIP-MDCFNPC (**2065** classes) are all tasks that own too many categories, so Micro-F1 is

[2]https://tianchi.aliyun.com/competition/entrance/532084/information
[3]https://tianchi.aliyun.com/competition/activeList
[4]https://github.com/Applied-Machine-Learning-Lab/MOELoRA-peft
[5]https://github.com/huggingface/peft

Table 2: The overall results of competing baselines and MOELoRA on PromptCBLUE. The boldface refers to the highest score and the underline indicates the best result of the methods. "*" indicates the statistically significant improvements (*i.e.,* two-sided t-test with $p < 0.05$) over the best baseline.

Model	CMeIE	CHIP-CDN	CHIP-CDEE	CHIP-MDCFNPC	CHIP-CTC	KUAKE-QIC	IMCS-V2-MRG	MedDG	Avg.
ChatGPT	0.3058	0.6069	0.2838	0.5854	0.5253	0.4851	0.3253	**0.1361**	0.4067
Huatuo	0.1826	0.3610	0.1658	0.3487	0.1909	0.1454	0.2401	0.1308	0.2207
P-Tuning	0.4552	0.8687	0.5256	0.7423	0.8275	0.8377	0.3155	0.0901	0.5828
LoRA (Full)	0.5089	0.8748	0.5464	0.7780	**0.8758**	0.8615	0.3678	0.1113	0.6155
LoRA (Single)	0.4984	0.8882	0.5528	0.7765	0.8694	0.8524	0.3583	0.1143	0.6138
LoRA (Full+TP)	0.4933	0.8814	0.5450	0.7705	0.8755	0.8664	0.3556	0.1160	0.6130
Task-Arithmetic	0.3928	0.7533	0.3216	0.6619	0.8091	0.6596	0.3163	0.1147	0.5037
LoRAHub	0.4411	0.8442	0.5041	0.7177	0.8564	0.8502	0.3061	0.1192	0.5799
MoLoRA	0.5081	0.8850	0.5656	0.7850	0.8749	0.8605	0.3590	0.1067	0.6181
MOELoRA(D)	**0.5193***	0.8928*	0.5697*	**0.7933***	0.8691	**0.8675**	**0.3681**	0.1089	**0.6236***
MOELoRA(S)	0.5110*	**0.8980***	**0.5719***	0.7872*	0.8682	0.8633	0.3558	0.1080	0.6204*

used to evaluate them too. By comparison, CHIP-CTC (**44** classes) and QUAKE-QIC (**7** classes) tasks both have fewer classes, which requires considering the equal importance of each class [31], so we apply Macro-F1 for them. As for text generation tasks, *i.e.,* IMCS-V2-MRG and MedDG, the Rouge-L [22] is applied. Also, the average score across all tasks is used for evaluating overall performance. To ensure the robustness and reproducibility of our results, tests are run thrice by random seeds $\{42, 43, 44\}$, with average scores reported in the following experimental results.

4.2 Overall Performance (RQ1)

The comprehensive experimental results of MOELoRA and the competing baselines are presented in Table 2. MOELoRA(D) and MOELoRA(S) represent the dense and sparse gate design for the MOELoRA, respectively. Analyzing the average scores across all tasks, it is evident that MOELoRA(D) consistently outperforms all other methods. To respond **RQ1**, the detailed analysis is given:

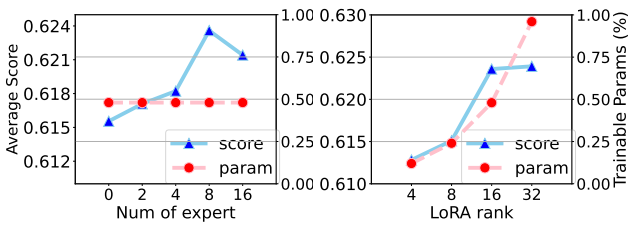
- **LLMs without Fine-tuning**: The first group of baselines, which are LLMs without any fine-tuning, significantly lag behind the other groups. This highlights the importance of fine-tuning LLMs to incorporate task-specific medical knowledge.
- **Parameter Efficient Fine-tuning Strategies**: Among the parameter efficient fine-tuning strategies, LoRA-based methods clearly surpass P-Tuning. While LoRA (Full) and LoRA (Full+TP) both utilize data from all tasks, LoRA (Full+TP) slightly underperforms. This might be attributed to the addition of task prompts, which extend the input texts, leading to the potential truncation of informative words due to input length constraints. LoRA (Single), which fine-tunes for individual tasks, also does not match the performance of LoRA (Full), underscoring the value of shared knowledge across tasks.
- **Model Editing**: The Task-Arithmetic evidently underperforms all tuning competitors. The reason may be that the model editing method is not suitable to the parameter efficient fine-tuning.
- **Cross-task Generalization**: We benchmark against two recent cross-task generalization methods. Despite their impressive performance in cross-task generalization settings, they require a vast amount of task data, which conflicts with the multi-task setting. In our experiments, we only consider 8 tasks, which might explain its relative underperformance.
- **Input-related Gate V.S. Task-related Gate**: MoLoRA can be considered as an input-related gate variant of our MOELoRA. We can find that MOELoRA outperforms MoLoRA. The reason could lie in that MoLoRA needs different gate functions for each MoLoRA layer, which leads to many redundant parameters and difficulty in training. Such performance comparison validates the effectiveness of task-motivated gate design.
- **Dense Gate V.S. Sparse Gate**: From the Table 2, we find that MOELoRA(S) performs the best on two tasks. The reason lies in that the sparse gate can help alleviate the optimization interference across various tasks [11], which can help learn some specific tasks better. However, for multi-task medical applications, shared medical knowledge is more vital. The dense gate can benefit the model learning shared knowledge by utilizing all experts, so such a design shows a superior performance on most tasks.
- **Task-specific Observations**: Performance variations are evident across tasks. For instance, LoRA (Full) and LoRA (Full+TP) excel in tasks with larger datasets, while LoRA (Single) shines in tasks with fewer samples, highlighting the data imbalance issue. MOELoRA consistently achieves optimal performance in most tasks, demonstrating its ability to effectively address this imbalance. For MedDG tasks, the inherent dialog capability of ChatGPT and Huatuo gives them an advantage over other approaches.

4.3 Ablation Study (RQ2)

To delve deeper into **RQ2** and understand the contributions of each component in MOELoRA, we present the results of our ablation study in Table 3. The variant *w/o MOE* (essentially reverts to LoRA (Full)) excludes the MOE architecture. It demonstrates inferior performance compared to the full-fledged MOELoRA, underscoring the significance of the MOE architecture. Similarly, the *w/o gate* variant, which employs uniform expert weights bypassing the gate function, also lags behind MOELoRA, highlighting the gate function's effectiveness. The *w multiple gate*

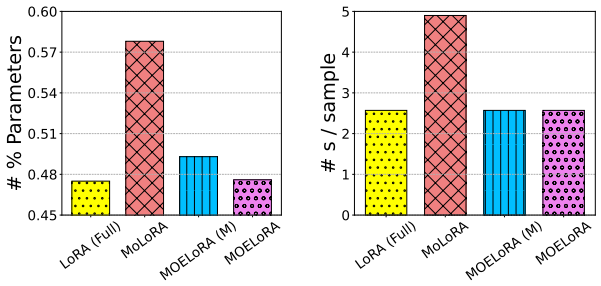
Table 3: The experimental results of ablation study for MOELoRA. The boldface refers to the highest score and the underline indicates the best result of the methods. "*" indicates the statistically significant improvements (*i.e.,* two-sided t-test with $p < 0.05$) over the best baseline.

Model	CMeIE	CHIP-CDN	CHIP-CDEE	CHIP-MDCFNPC	CHIP-CTC	KUAKE-QIC	IMCS-V2-MRG	MedDG	Avg.
w/o MOE	0.5089	0.8748	0.5464	0.7780	0.8758	0.8615	0.3678	0.1113	0.6155
w/o gate	0.5015	0.8840	0.5378	0.7789	**0.8818**	0.8699	0.3709	**0.1140**	0.6174
w multiple gate	0.4994	0.8840	0.5692	0.7842	0.8764	0.8675	0.3632	0.1130	0.6196
w BT	0.4817	0.8806	**0.5712**	0.7713	0.8682	0.8643	0.3522	0.1110	0.6126
w RBT	0.4769	0.8830	0.5600	0.7741	0.8636	**0.8795**	0.3541	0.1135	0.6131
LoRA (Full)-QKV	0.4666	0.8605	0.4997	0.7703	0.8264	0.7161	0.3636	0.1123	0.5770
MOELoRA(D)-QKV	0.4897	0.8733	0.5227	0.7854	0.8213	0.7295	0.3640	0.1099	0.5869
MOELoRA(D)	**0.5193***	**0.8928***	0.5697	**0.7933***	0.8691	0.8675	**0.3681**	0.1089	**0.6236***



(a) (b)

Figure 4: The results of experiments for hyper-parameters, *i.e.,* expert number N and LoRA rank r.



(a) Ratio of Parameters **(b) Inference Latency**

Figure 5: The results of experiments for comparing training and inference efficiency.

variant, uses a unique gate function for each MOELoRA layer. We can see that it achieves comparable results on several tasks and is slightly outperformed by the single gate function design due to over-parameterization. Besides, multiple gate functions also incur a higher count of trainable parameters, leading to diminished efficiency compared to a single gate function setup.

Additionally, we analyze the impact of different training strategies. Specifically, the *w BT* method [36] consolidates samples from the same task into one batch. The *w RBT* approach [39] randomly selects a task for each batch of data. Both of them prove to be less conducive for MOELoRA, resulting in performance degradation. This performance comparison underscores the influence of specific training patterns.

For verification of the robustness of the proposed MOELoRA, We have conducted the experiments of only imposing LoRA layers on the attention layers, denoted as *LoRA (Full)-QKV* and *MOELoRA(D)-QKV*. From the results, we find that *MOELoRA(D)-QKV* can outperform *LoRA (Full)-QKV* on most tasks, which aligns the performance comparison between *MOELoRA(D)* and *LoRA (Full)* in Table 2. Besides, *MOELoRA(D)* is better than *MOELoRA(D)-QKV*, which illustrates more MOELoRA layers can raise the performance of fine-tuning consistently.

4.4 Hyper-parameter Analysis (RQ3)

To answer **RQ3**, we delve into the impact of hyper-parameters on the performance of MOELoRA(D). Specifically, we examine how variations in the expert number N and LoRA rank r influence the results, as depicted in Figure 4. Our observations reveal that the performance of MOELoRA improves as N increases from 0 to 8, while fixing the overall LoRA rank r as 16. This enhancement can be attributed to the fact that a greater number of experts facilitate the learning of a broader spectrum of knowledge [35]. However, when N escalates to 16, we notice a marginal decline in performance. This can be explained by a large expert number leading to a small LoRA rank for each expert, which degrades the learning ability of low-rank matrices. Subsequently, we set the rank of each expert, *i.e.,* $\frac{r}{N}$, as 2. We can observe from Figure 4b that while an increase in r consistently boosts performance, it also leads to a proportionate surge in the size of trainable parameters. Given the need to strike a balance between efficiency and performance, a practical choice for r would be 16.

4.5 Efficiency Analysis (RQ4)

To evaluate the training and inference efficiency, we compare the ratio of tunable parameters and inference latency in Figure 5. Inference latency is calculated by averaging the inference time on the number of inference samples. MOELoRA(M) denotes the variant of MOELoRA, where every MOELoRA layer is accompanied by a task-motivated gate. The results show that MOELoRA achieves as high training and inference efficiency as LoRA (Full), which can save resources by training no more than 0.48% parameters of LLMs. MoLoRA and MOELoRA(M) need more trainable parameters, because they set the extra gates for each trainable low-rank layer. In terms of inference, all models need the same inference latency, except MoLoRA. The reason lies in that MoLoRA cannot recover the fine-tuned parameters as Equation (8), because the expert weights vary from samples. Therefore, it needs to accompany the MoLoRA layers when inference, which

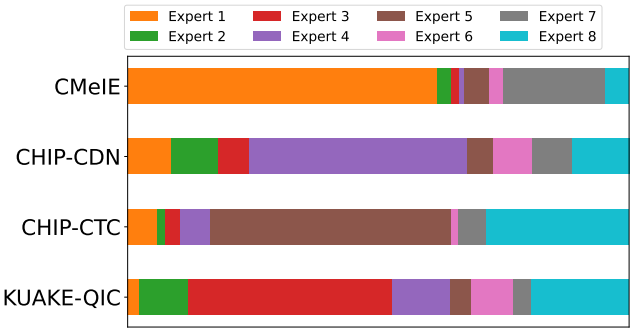


Figure 6: The visualization of expert weights for various tasks. In each task, the length of the bar in different colors represents the weights for the corresponding expert.

leads to more inference latency caused by the additional forward computation. This comparison indicates the benefit of the design of the task-motivated gate. As a response to the **RQ4**, the designed MOELoRA achieves high training and inference efficiency, and avoids efficiency degradation by the task-motivated gate.

4.6 Case Study (RQ5)

For **RQ4**, we present a visualization of the expert weights across four tasks in Figure 6. For each task, the length of the bar in different colors represents the weights for the corresponding expert. Since the expert weights are normalized to 1, the lengths of the bar for each task are the same. At a macro level, it is evident that the contributions from each expert vary significantly, underscoring the idea that different experts specialize in distinct facets of medical knowledge. Moreover, the pronounced disparities in weights across tasks highlight the diverse nature of medical applications. Taking a closer look at the tasks CHIP-CDN and KUAKE-QIC, we observe that their expert weights are largely congruent, with exceptions in experts 3 and 4. Considering diagnostic word normalization can bolster inquiry classification, the similarity in expert weights suggests that MOELoRA is adept at harnessing shared knowledge to benefit related tasks.

5 RELATED WORKS

5.1 LLM for Medical Applications

Recently, the powerful capabilities of LLMs have been proven in many fields [9, 10, 18, 19, 27, 28, 44, 48, 50, 60], including the medical domain. For instance, Med-PaLM [37] proposes a new benchmark called MultiMedQA, which combines seven medical question-answering datasets to address the challenges of evaluating the clinical knowledge of LLM. Med-PaLM2 [38] has further improved upon Med-PaLM by introducing a new prompting strategy called ensemble refinement. This strategy is based on CoT [49] and self-consistency [45] and has shown significant improvements in MedQA. Then, ChatDoctor [53] trains medical LLMs by 100,000 patient-doctor dialogues collected from a widely used online medical consultation platform. Besides, HuaTuo [42] is initially based on LlaMA [40] and fine-tuned using Chinese medical knowledge from CMeKG [4]. For more specific medical applications, Liu *et al.* [23] proposes an LLM-based medication recommendation model, while Xu *et al.* [51] designs a model editing method to resolve hallucination in medical LLMs. However, most previous works tend to focus on medical dialogue or one specific medical task while neglecting multiple important tasks simultaneously. Besides, they usually demand a significant fine-tuning cost for achieving generalization ability.

5.2 Parameter Efficient Fine-tuning

Parameter efficient fine-tuning (PEFT) aims to improve the performance of LLMs on new tasks by minimizing the number of fine-tuning parameters and computational complexity. Adapter Tuning [13] first introduces a lightweight adapter module, which has only a few trainable parameters. Prefix-tuning [21] and P-Tuning [24] both construct a task-specific virtual token that adds trainable, continuous prompts or embeddings to the original text sequence. However, using prompts can be challenging for training and can also limit the available sequence length of the model. LoRA [14] introduces two trainable low-rank matrices into each dense layer. It has been shown to achieve comparable performance to full fine-tuning while requiring no additional computation during inference. Nevertheless, the LoRA fine-tuning performs inferiorly for multi-task medical applications. In recent times, a thread of research named cross-task generalization [2, 15, 39, 41] emerges, which proposes various parameter efficient fine-tuning strategies. However, different from the multi-task setting in this paper, they first train the model on too many tasks and aim to transfer the ability to unseen tasks. Due to the distinct setting, their method is difficult to be adapted to our problem. In a word, the multi-task parameter efficient fine-tuning for LLM-driven medical applications is still underexplored, and we take the first step.

6 CONCLUSION

In this paper, we take the first step to explore the multi-task parameter efficient fine-tuning method for LLM-driven medical applications. To satisfy the requirements of efficiency for fine-tuning and effectiveness for multi-task, we propose a novel multi-task fine-tuning framework. Specifically, we design the MOELoRA architecture, which consists of several low-rank experts as the trainable parameters to learn task-related knowledge and retain high efficiency. Besides, a task-motivated gate function is devised to produce distinct fine-tuned parameters for various tasks. By the comprehensive experiments on a multi-task Chinese medical dataset, we verify the effectiveness of the proposed MOELoRA. In the future, we will further explore how to combine explicit medical knowledge, such as knowledge graphs, with LLMs by fine-tuning.

ACKNOWLEDGMENTS

This research was supported by Tencent (CCF-Tencent Open Fund), Research Impact Fund (No.R1015-23), APRC - CityU New Research Initiatives (No.9610565, Start-up Grant for New Faculty of CityU), CityU - HKIDS Early Career Research Grant (No.9360163), Hong Kong ITC Innovation and Technology Fund Midstream Research Programme for Universities Project (No.ITS/034/22MS), Hong Kong Environmental and Conservation Fund (No. 88/2022), SIRG - CityU Strategic Interdisciplinary Research Grant (No.7020046, No.7020074).

REFERENCES

[1] Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. 2021. Intrinsic Dimensionality Explains the Effectiveness of Language Model Fine-Tuning. 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)*. 7319–7328.

[2] Akari Asai, Mohammadreza Salehi, Matthew E Peters, and Hannaneh Hajishirzi. 2022. Attempt: Parameter-efficient multi-task tuning via attentional mixtures of soft prompts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. 6655–6672.

[3] 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. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[4] Odma Byambasuren, Yunfei Yang, Zhifang Sui, Damai Dai, Baobao Chang, Sujian Li, and Hongying Zan. 2019. Preliminary study on the construction of Chinese medical knowledge graph. *Journal of Chinese Information Processing* 33, 10 (2019), 1–9.

[5] Michael Crawshaw. 2020. Multi-task learning with deep neural networks: A survey. *arXiv preprint arXiv:2009.09796* (2020).

[6] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022. A survey for in-context learning. *arXiv preprint arXiv:2301.00234* (2022).

[7] Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. GLM: General Language Model Pretraining with Autoregressive Blank Infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 320–335.

[8] Wenqi Fan, Xiangyu Zhao, Xiao Chen, Jingran Su, Jingtong Gao, Lin Wang, Qidong Liu, Yiqi Wang, Han Xu, Lei Chen, et al. 2022. A comprehensive survey on trustworthy recommender systems. *arXiv preprint arXiv:2209.10117* (2022).

[9] Wenqi Fan, Zihuai Zhao, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Jiliang Tang, and Qing Li. 2023. Recommender systems in the era of large language models (llms). *arXiv preprint arXiv:2307.02046* (2023).

[10] Zichuan Fu, Xiangyang Li, Chuhan Wu, Yichao Wang, Kuicai Dong, Xiangyu Zhao, Mengchen Zhao, Huifeng Guo, and Ruiming Tang. 2023. A Unified Framework for Multi-Domain CTR Prediction via Large Language Models. *arXiv preprint arXiv:2312.10743* (2023).

[11] Shashank Gupta, Subhabrata Mukherjee, Krishan Subudhi, Eduardo Gonzalez, Damien Jose, Ahmed H Awadallah, and Jianfeng Gao. 2022. Sparsely activated mixture-of-experts are robust multi-task learners. *arXiv preprint arXiv:2204.07689* (2022).

[12] Muhammad Usman Hadi, R Qureshi, A Shah, M Irfan, A Zafar, MB Shaikh, N Akhtar, J Wu, and S Mirjalili. 2023. A survey on large language models: Applications, challenges, limitations, and practical usage. *TechRxiv* (2023).

[13] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning*. PMLR, 2790–2799.

[14] Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2021. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*.

[15] Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. 2023. LoraHub: Efficient Cross-Task Generalization via Dynamic LoRA Composition. *arXiv preprint arXiv:2307.13269* (2023).

[16] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2022. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*.

[17] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT*. 4171–4186.

[18] Xinhang Li, Chong Chen, Xiangyu Zhao, Yong Zhang, and Chunxiao Xing. 2023. E4SRec: An elegant effective efficient extensible solution of large language models for sequential recommendation. *arXiv preprint arXiv:2312.02443* (2023).

[19] Xiaopeng Li, Lixin Su, Pengyue Jia, Xiangyu Zhao, Suqi Cheng, Junfeng Wang, and Dawei Yin. 2023. Agent4Ranking: Semantic Robust Ranking via Personalized Query Rewriting Using Multi-agent LLM. *arXiv preprint arXiv:2312.15450* (2023).

[20] Xiaoya Li, Xiaofei Sun, Yuxian Meng, Junjun Liang, Fei Wu, and Jiwei Li. 2020. Dice Loss for Data-imbalanced NLP Tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 465–476.

[21] Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. 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)*. Association for Computational Linguistics, Online, 4582–4597. https://doi.org/10.18653/v1/2021.acl-long.353

[22] Chin-Yew Lin and Franz Josef Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*. 605–612.

[23] Qidong Liu, Xian Wu, Xiangyu Zhao, Yuanshao Zhu, Zijian Zhang, Feng Tian, and Yefeng Zheng. 2024. Large Language Model Distilling Medication Recommendation Model. *arXiv preprint arXiv:2402.02803* (2024).

[24] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-Tuning: Prompt Tuning Can Be Comparable to Fine-tuning Across Scales and Tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Dublin, Ireland, 61–68. https://doi.org/10.18653/v1/2022.acl-short.8

[25] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2023. GPT understands, too. *AI Open* (2023).

[26] Ziru Liu, Jiejie Tian, Qingpeng Cai, Xiangyu Zhao, Jingtong Gao, Shuchang Liu, Dayou Chen, Tonghao He, Dong Zheng, Peng Jiang, et al. 2023. Multi-task recommendations with reinforcement learning. In *Proceedings of the ACM Web Conference 2023*. 1273–1282.

[27] Sichun Luo, Bowei He, Haohan Zhao, Yinya Huang, Aojun Zhou, Zongpeng Li, Yuanzhang Xiao, Mingjie Zhan, and Linqi Song. 2023. RecRanker: Instruction Tuning Large Language Model as Ranker for Top-k Recommendation. *arXiv preprint arXiv:2312.16018* (2023).

[28] Sichun Luo, Yuxuan Yao, Bowei He, Yinya Huang, Aojun Zhou, Xinyi Zhang, Yuanzhang Xiao, Mingjie Zhan, and Linqi Song. 2024. Integrating Large Language Models into Recommendation via Mutual Augmentation and Adaptive Aggregation. *arXiv preprint arXiv:2401.13870* (2024).

[29] Yasuhide Miura, Yuhao Zhang, Emily Bao Tsai, Curtis P Langlotz, and Dan Jurafsky. 2020. Improving factual completeness and consistency of image-to-text radiology report generation. *arXiv preprint arXiv:2010.10042* (2020).

[30] OpenAI. 2023. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774* (2023).

[31] Juri Opitz and Sebastian Burst. 2019. Macro f1 and macro f1. *arXiv preprint arXiv:1911.03347* (2019).

[32] Zhi Qiao, X. Wu, Shen Ge, and Wei Fan. 2019. MNN: Multimodal Attentional Neural Networks for Diagnosis Prediction. In *International Joint Conference on Artificial Intelligence*. https://api.semanticscholar.org/CorpusID:199466261

[33] Saed Rezayi, Haixing Dai, Zhengliang Liu, Zihao Wu, Akarsh Hebbar, Andrew H Burns, Lin Zhao, Dajiang Zhu, Quanzheng Li, Wei Liu, et al. 2022. Clinicalradiobert: Knowledge-infused few shot learning for clinical notes named entity recognition. In *International Workshop on Machine Learning in Medical Imaging*. Springer, 269–278.

[34] Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. 2021. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems* 34 (2021), 8583–8595.

[35] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538* (2017).

[36] Xiang-Rong Sheng, Liqin Zhao, Guorui Zhou, Xinyao Ding, Binding Dai, Qiang Luo, Siran Yang, Jingshan Lv, Chi Zhang, Hongbo Deng, et al. 2021. One model to serve all: Star topology adaptive recommender for multi-domain ctr prediction. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 4104–4113.

[37] Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, et al. 2023. Large language models encode clinical knowledge. *Nature* 620, 7972 (2023), 172–180.

[38] Karan Singhal, Tao Tu, Juraj Gottweis, Rory Sayres, Ellery Wulczyn, Le Hou, Kevin Clark, Stephen Pfohl, Heather Cole-Lewis, Darlene Neal, et al. 2023. Towards expert-level medical question answering with large language models. *arXiv preprint arXiv:2305.09617* (2023).

[39] Tianxiang Sun, Zhengfu He, Qin Zhu, Xipeng Qiu, and Xuan-Jing Huang. 2023. Multitask Pre-training of Modular Prompt for Chinese Few-Shot Learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 11156–11172.

[40] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971* (2023).

[41] Ahmet Üstün, Arianna Bisazza, Gosse Bouma, Gertjan van Noord, and Sebastian Ruder. 2022. Hyper-X: A unified hypernetwork for multi-task multilingual transfer. *arXiv preprint arXiv:2205.12148* (2022).

[42] Haochun Wang, Chi Liu, Nuwa Xi, Zewen Qiang, Sendong Zhao, Bing Qin, and Ting Liu. 2023. HuaTuo: Tuning LLaMA Model with Chinese Medical Knowledge. arXiv:2304.06975 [cs.CL]

[43] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2023. A survey on

large language model based autonomous agents. *arXiv preprint arXiv:2308.11432* (2023).

[44] Maolin Wang, Yao Zhao, Jiajia Liu, Jingdong Chen, Chenyi Zhuang, Jinjie Gu, Ruocheng Guo, and Xiangyu Zhao. 2023. Large Multimodal Model Compression via Efficient Pruning and Distillation at AntGroup. *arXiv preprint arXiv:2312.05795* (2023).

[45] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171* (2022).

[46] Yuhao Wang, Ha Tsz Lam, Yi Wong, Ziru Liu, Xiangyu Zhao, Yichao Wang, Bo Chen, Huifeng Guo, and Ruiming Tang. 2023. Multi-task deep recommender systems: A survey. *arXiv preprint arXiv:2302.03525* (2023).

[47] Yu Wang, Hanghang Tong, Ziye Zhu, and Yun Li. 2022. Nested named entity recognition: a survey. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 16, 6 (2022), 1–29.

[48] Yuhao Wang, Xiangyu Zhao, Bo Chen, Qidong Liu, Huifeng Guo, Huanshuo Liu, Yichao Wang, Rui Zhang, and Ruiming Tang. 2023. PLATE: A Prompt-Enhanced Paradigm for Multi-Scenario Recommendations. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1498–1507.

[49] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems* 35 (2022), 24824–24837.

[50] Derong Xu, Ziheng Zhang, Zhenxi Lin, Xian Wu, Zhihong Zhu, Tong Xu, Xiangyu Zhao, Yefeng Zheng, and Enhong Chen. 2024. Multi-perspective Improvement of Knowledge Graph Completion with Large Language Models. *arXiv preprint arXiv:2403.01972* (2024).

[51] Derong Xu, Ziheng Zhang, Zhihong Zhu, Zhenxi Lin, Qidong Liu, Xian Wu, Tong Xu, Xiangyu Zhao, Yefeng Zheng, and Enhong Chen. 2024. Editing Factual Knowledge and Explanatory Ability of Medical Large Language Models. *arXiv preprint arXiv:2402.18099* (2024).

[52] Aiyuan Yang, Bin Xiao, Bingning Wang, Borong Zhang, Chao Yin, Chenxu Lv, Da Pan, Dian Wang, Dong Yan, Fan Yang, et al. 2023. Baichuan 2: Open Large-scale Language Models. *arXiv preprint arXiv:2309.10305* (2023).

[53] Li Yunxiang, Li Zihan, Zhang Kai, Dan Ruilong, and Zhang You. 2023. Chatdoctor: A medical chat model fine-tuned on llama model using medical domain knowledge. *arXiv preprint arXiv:2303.14070* (2023).

[54] Ted Zadouri, Ahmet Üstün, Arash Ahmadian, Beyza Ermiş, Acyr Locatelli, and Sara Hooker. 2023. Pushing mixture of experts to the limit: Extremely parameter efficient moe for instruction tuning. *arXiv preprint arXiv:2309.05444* (2023).

[55] Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, et al. 2022. GLM-130B: An Open Bilingual Pre-trained Model. In *The Eleventh International Conference on Learning Representations*.

[56] Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. 2023. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792* (2023).

[57] Yingying Zhang, Xian Wu, Quan Fang, Shengsheng Qian, and Changsheng Xu. 2023. Knowledge-Enhanced Attributed Multi-Task Learning for Medicine Recommendation. *ACM Trans. Inf. Syst.*, Article 17 (jan 2023), 24 pages.

[58] Yu Zhang and Qiang Yang. 2021. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering* 34, 12 (2021), 5586–5609.

[59] Sendong Zhao, Ting Liu, Sicheng Zhao, and Fei Wang. 2019. A neural multi-task learning framework to jointly model medical named entity recognition and normalization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 817–824.

[60] Zhi Zheng, Wenshuo Chao, Zhaopeng Qiu, Hengshu Zhu, and Hui Xiong. 2024. Harnessing Large Language Models for Text-Rich Sequential Recommendation. *arXiv preprint arXiv:2403.13325* (2024).

[61] Zhi Zheng, Zhaopeng Qiu, Hui Xiong, Xian Wu, Tong Xu, Enhong Chen, and Xiangyu Zhao. 2022. DDR: Dialogue Based Doctor Recommendation for Online Medical Service. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Washington DC, USA) *(KDD '22)*. Association for Computing Machinery, New York, NY, USA, 4592–4600. https://doi.org/10.1145/3534678.3539201

[62] Zhi Zheng, Chao Wang, Tong Xu, Dazhong Shen, Penggang Qin, Xiangyu Zhao, Baoxing Huai, Xian Wu, and Enhong Chen. 2023. Interaction-aware drug package recommendation via policy gradient. *ACM Transactions on Information Systems* 41, 1 (2023), 1–32.

[63] Wei Zhu, Xiaoling Wang, Huanran Zheng, Mosha Chen, and Buzhou Tang. 2023. PromptCBLUE: A Chinese Prompt Tuning Benchmark for the Medical Domain. *arXiv preprint arXiv:2310.14151* (2023).