Review of Fine-Tuning Techniques for Large Language Models

Jianhua Guo

***Abstract*—With the rapid advancement of artificial intelligence technology, large language models (LLMs) have gradually become a hot research topic and are actively applied in various fields. LLMs are pre-trained language models with over tens of billions of parameters, trained on large-scale corpora, showcasing excellent natural language processing capabilities and demonstrating strong application potential in vertical domains such as healthcare, programming, and data generation. However, as model scales increase, the consumption of computing resources rises, interpretability declines, and the generation of inaccurate or harmful content raises concerns about reliability.**

To address these issues, aligning model behavior with human values has become crucial. This article reviews the research progress in parameter fine-tuning, focusing on classical parameter fine-tuning methods as well as recent efficient fine-tuning strategies, including prompt tuning and reinforcement learning fine-tuning. It also looks ahead to future research directions in the field of parameter fine-tuning.

***Index Terms*—Large Language Models, Alignment, Techniques, Reinforcement Learning**

I. Introduction

The rise of large language models marks a new milestone in the field of deep learning, with fine-tuning serving as the bridge connecting pre-trained large models to specific application scenarios. As artificial intelligence continues to evolve, fine-tuning has become crucial for enhancing model applicability, efficiency, and accuracy [1].

This technique involves refining a model that has been pre-trained on a large-scale dataset by further adjusting it on a specific dataset for a particular task, thereby transferring the broad knowledge gained during pre-training into specific applications [2]. This strategy can be seen as the core of transfer learning, seamlessly integrating general knowledge with specific tasks, which is especially important for dealing with tasks requiring specialized knowledge or specific context.

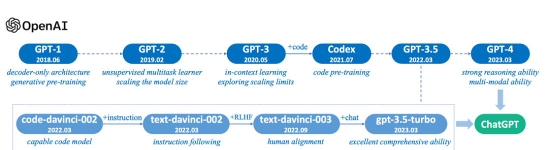

Fig. 1. The Evolution of the OpenAI GPT Series of Language Models

Fine-tuning holds significant value for the development of natural language processing (NLP) technology and the improvement of large language model performance [3]. Firstly, it avoids the enormous costs associated with training large models from scratch, reducing the consumption of substantial computational resources. This makes it possible for researchers and developers to effectively train models even with limited resources [4].

Moreover, since the model has already learned a substantial amount of general knowledge during the pre-training phase, fine-tuning enables the model to more effectively learn the features of a specific task, even with a limited amount of labeled data [5].

For example, OpenAI's work has further advanced the traditional "pre-training + fine-tuning" paradigm for language models, beginning with GPT-1 (Generative Pre-trained Transformer) [6], which paved the way for the era of fine-tuning large language models. The GPT series continued to demonstrate the scaling laws for large language model parameters [7]. GPT-3 introduced in-context learning, which led to the development of prompt tuning techniques; InstructGPT, based on instruction tuning, and GPT-4, fine-tuned using reinforcement learning from human feedback (RLHF) [8], achieved remarkable results.

In China, many leading language models have actively adopted advanced fine-tuning methods. For instance, the knowledge-enhanced model "Ernie Bot" integrates supervised fine-tuning, RLHF, and prompt learning techniques [9]. Additionally, excellent language models like iFlytek's Xinghuo and Tencent's Hunyuan have reached or even surpassed the level of international counterparts [10].

In recent years, there has been rapid innovation and development in combining fine-tuning with large language models. This evolution spans from traditional fine-tuning strategies involving hyperparameters such as learning rate to emerging strategies like instruction tuning [11]. Zhang et al. have delved into these processes in depth, while Han et al. and Qiu et al [12]. have provided comprehensive reviews on the development of pre-trained models. Liu et al. compared and summarized the rapid advancements in prompt strategies, while Ding et al [13]. explored optimizing fine-tuning without adding parameters.

II. Development Overview of Large Language Models

From early analytical predictive language models to today's large language models with billions to trillions of parameters, significant progress has been made. The development of large language models can be divided into the following four stages:

A. Early Language Models and Neural Network Models

In the 1990s and early 2000s, language models were primarily based on statistical methods, such as n-gram models and hidden Markov models (HMMs), which predicted text

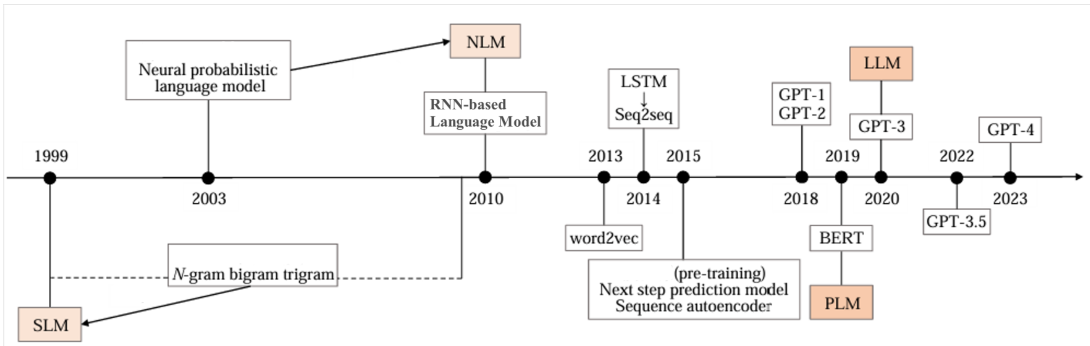

Fig. 2. Fine-tune the language model using supervised data

by analyzing word frequencies and co-occurrence patterns. As computational power improved and more complex and unpredictable processing challenges emerged, neural networks were gradually introduced into language processing tasks.

The origin of neural network language models (NLMs) can be traced back to the neural probabilistic language model proposed by Bengio et al., although at that time, the role of neural networks was limited to providing a single feature for existing statistical language models (SLMs). It wasn't until the introduction of RNN-based language models in 2010 that their performance was shown to surpass that of traditional n-gram models, marking the official transition of language models into the NLM phase. Various neural network-based word embedding methods, such as Word2Vec and GloVe, have also been shown to significantly enhance the accuracy of word representations.

The rapid development of neural network language models has led researchers to explore more complex model architectures, such as Long Short-Term Memory networks (LSTMs) and Gated Recurrent Units (GRUs). These models effectively capture long-distance dependencies, resulting in superior performance when processing contextual information. The release of BERT in 2018 marked an important milestone in the field of natural language processing. BERT employs a bidirectional encoding approach, pre-training on large-scale unlabeled text and then fine-tuning for specific tasks, which significantly improves performance across multiple natural language processing tasks.

Since then, many Transformer-based models have emerged, such as the GPT series, RoBERTa, and XLNet, demonstrating exceptional capabilities in tasks like text generation, text comprehension, and question-answering systems. Currently, language models are not only attracting widespread attention in academia but also showcasing their strong potential in practical applications, such as machine translation, sentiment analysis, and dialogue systems.

B. The Introduction of Sequence-to-Sequence Models and Attention Mechanisms

In 2014, Ilya Sutskever, Oriol Vinyals, and others from Google introduced the sequence-to-sequence (Seq2Seq) model, an innovative architecture specifically designed for machine translation tasks. This model can translate entire sentences rather than relying on word-by-word translation. The Seq2Seq architecture employs an encoder-decoder framework, where the encoder processes the input sequence and compresses it into a fixed-length context vector that serves as a summary of the input. The decoder then takes this context vector and generates the corresponding output sequence. This structure enables the model to handle varying lengths of input and output, making it suitable for translation between languages with different sentence structures.

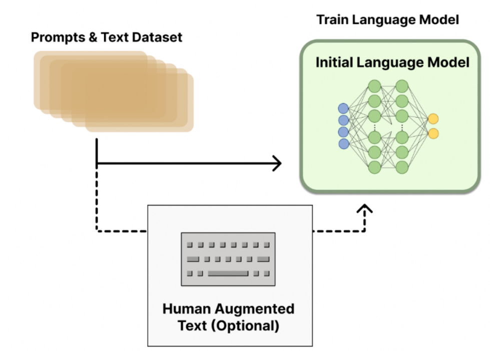

Fig. 3. Fine-tune the language model using supervised data

To further enhance the performance of the Seq2Seq model, Dzmitry Bahdanau, Kyunghyun Cho, and others introduced the attention mechanism. This mechanism allows large language models to dynamically focus on different parts of the input sequence during the decoding process, rather than solely relying on the fixed-length context vector. By attending to specific words or phrases in the input, the model can generate more accurate and contextually relevant translations. The introduction of the Seq2Seq model and the attention mechanism laid a critical foundation for the subsequent development of large-scale pre-trained models.

The attention mechanism revolutionized the capabilities of Seq2Seq models, leading to more contextually aware outputs. By allowing the model to weigh the importance of various input elements, it can generate translations that better reflect the nuances and complexities of language. This development laid the groundwork for the evolution of large-scale pre-trained models, which leverage the combined strengths of Seq2Seq architectures and attention mechanisms. Today, these advanced models power a wide array of applications, from real-time translation services to sophisticated conversational agents, underscoring the profound impact of the Seq2Seq framework and attention mechanisms on the future of NLP.

C. The Proposal of Large-Scale Pre-Training and Fine-Tuning

With the expansion of computing resources and the exponential growth in datasets, pre-training followed by fine-tuning has become the dominant approach in natural language processing. In 2018, Jacob Devlin and his team at Google introduced BERT (Bidirectional Encoder Representations from Transformers), which used a bidirectional training strategy to consider the context on both sides of a token within a sentence. This approach significantly improved language understanding by capturing the nuances of meaning based on surrounding words.

Meanwhile, OpenAI developed the Generative Pre-trained Transformer (GPT), which generates text using an autoregressive strategy, predicting the next word based on preceding words. This architecture enables GPT to produce coherent and relevant content, showcasing strong text generation capabilities.

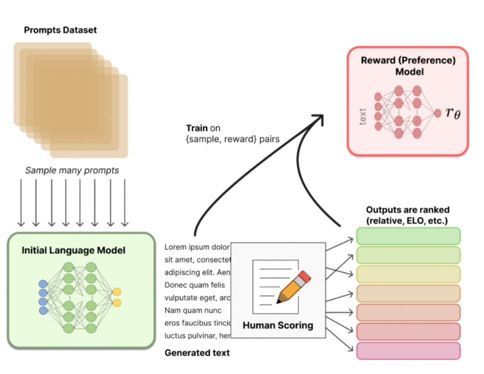

Fig. 4. Train the reward model

The release of GPT-2 in 2019 brought substantial improvements over its predecessor, delivering high-quality text generation across diverse topics and styles. In 2020, GPT-3 took this even further by expanding the model's parameter count to 175 billion, up from GPT-2's 1.5 billion. This massive increase in parameters greatly enhanced GPT-3's generative abilities and language comprehension, allowing it to tackle a wide range of tasks with minimal fine-tuning.

GPT-3 has demonstrated impressive few-shot and zero-shot learning capabilities, understanding and responding to tasks it has never been explicitly trained on.

D. Development of Large Language Models in China

The rapid development of large language models in China has benefited from the increase in computing resources and the continuous maturation of technology. The transition from early machine learning and deep learning methods to innovations based on the Transformer architecture has enabled domestic research institutions and enterprises to quickly catch up with international advancements. Following the release of the BERT model, several teams developed BERT-based variants, such as Huawei's "PanGu Model." These models have achieved significant progress in text generation and understanding capabilities, thereby advancing the field of natural language processing.

As technology iterates and application scenarios expand, domestic companies have successively launched their independently developed large language models, such as Alibaba's "Tongyi Qianwen," Baidu's "Wenxin," iFlytek's "Xinghuo Cognitive Model," and Tencent's "Hunyuan Model." These models not only demonstrate strong capabilities in knowledge questioning and dialogue generation but also find widespread applications in various industries, including education, healthcare, and finance. Meanwhile, the diversification of fine-tuning techniques has supported the adaptability and efficiency of these models, further promoting in-depth exploration of large language models in practical applications.

At the same time, China's large language models have begun to incorporate regional linguistic characteristics and cultural nuances, making them more relevant and effective for local users. This localization is particularly important in addressing the unique challenges posed by the Chinese language, including its tonal nature and complex character system. A continued focus on local needs ensures that these technologies are not only powerful but also user-friendly.

Looking ahead, the future of large language models in China appears promising, with ongoing investments in research, the emergence of new startups, and a vibrant ecosystem that fosters innovation. The continuous development of these models is likely to play a crucial role in transforming industries, enhancing productivity, and driving economic growth across the nation, ultimately contributing to a more interconnected and intelligent society.

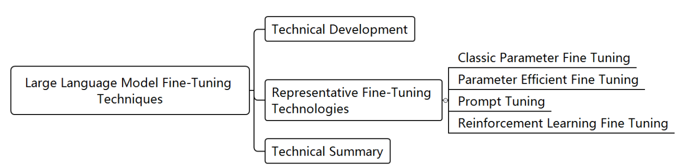

Fig. 5. Content Architecture Diagram

III. Advancements in Fine-Tuning Techniques

Fine-tuning techniques have significantly advanced, driving the widespread adoption and performance improvements of large language models. Traditionally, fine-tuning involved adjusting all parameters of the pre-trained model to adapt it to specific tasks. However, as the size of these models increased, the computational costs and resource requirements of this approach became prohibitive. Consequently, efficient parameter fine-tuning methods have emerged, such as fine-tuning selective layers or leveraging low-rank matrix decomposition techniques to reduce the complexity and resource demands of parameter updates. Simultaneously, novel strategies like prompt tuning and instruction tuning have gained popularity. These methods improve task adaptability by adjusting input prompts or fine-tuning models to better follow given instructions.

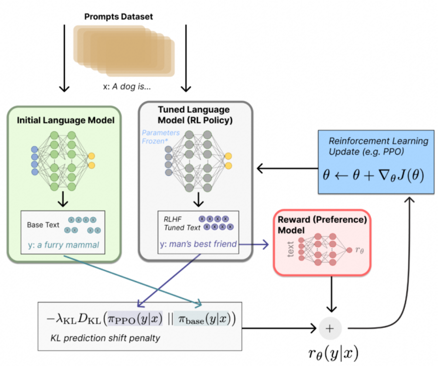

Fig. 6. Train the reinforcement learning model

Moreover, reinforcement learning, particularly reinforcement learning from human feedback (RLHF), has been incorporated into the fine-tuning process. This approach enables models to iteratively refine their outputs based on user feedback, thereby enhancing the quality of generated content and user satisfaction. These methods allow large language models to exhibit greater flexibility and efficiency across diverse tasks, facilitating the development of practical applications such as chatbots, text generation, and intelligent question answering. The diversification of fine-tuning techniques has become a critical pathway for enhancing large language model performance and is leading the ongoing technological innovation in the field of natural language processing.

IV. Methods of Fine-Tuning

A. Classic Parameter Fine-Tuning

Classic parameter fine-tuning [14], often referred to as full parameter tuning, is a model optimization technique in deep learning, especially used in transfer learning or domain adaptation scenarios. It involves fine-tuning all the parameters of a pre-trained model to adapt it to a specific task or dataset [15]. This approach allows the model to retain the knowledge gained during pre-training [16] while optimizing for a particular task. However, it comes with higher computational resource requirements.

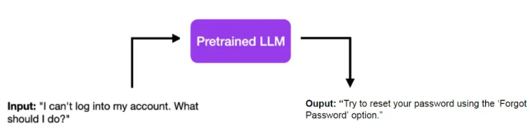

Fig. 7. Unsupervised fine-tuning

Full parameter fine-tuning primarily refers to supervised fine-tuning applied in model transfer for downstream subtasks [17], while unsupervised fine-tuning is often used in the pre-training phase of models. The applications of unsupervised fine-tuning [18] and self-supervised fine-tuning [19] in downstream tasks are still in development, and researchers are also focusing on the impact of both on the model's inductive biases, categorizing them as behavior fine-tuning and adaptive fine-tuning.

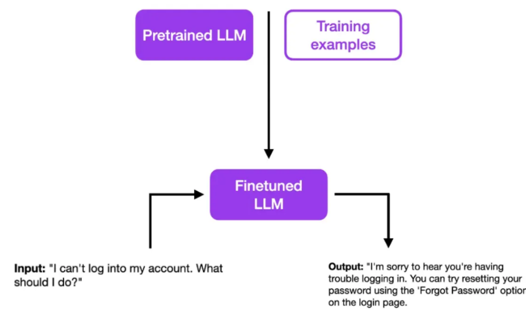

Fig. 8. Supervised fine-tuning

Supervised fine-tuning is based on the idea of fine-tuning in deep learning, where large language models are first pre-trained, followed by fine-tuning on labeled datasets for specific tasks to minimize the cross-entropy loss function [20].

In 2018, BERT introduced a method of optimizing model performance through pre-training on large text corpora followed by fine-tuning for specific tasks, laying the groundwork for subsequent research on full parameter fine-tuning. Howard and Ruder proposed a universal language model fine-tuning (ULMFiT) approach based on transfer learning [21], significantly improving fine-tuning efficiency.

To optimize resource usage, Malladi et al [22]. introduced a memory-efficient zeroth-order optimizer (MeZO), reducing memory consumption to one-twelfth of its original size, while Lv et al.'s low memory optimization (LOMO) reduced memory usage to 10.8%.

In unsupervised fine-tuning, models learn the structure and patterns of data through self-supervised tasks, particularly useful when labeled data is scarce. Li et al [23]. proposed strategies for sparse source data replay and data mixing to enhance the effectiveness of unsupervised fine-tuning, and some researchers focused on improvements in weakly supervised fine-tuning. Yu et al [24]. proposed fine-tuning pre-trained LLMs under weak supervision, while Huang et al [25]. developed a semi-supervised learning approach combining limited labeled data, and Tanwisuth et al [26]. introduced prompt-oriented unsupervised fine-tuning methods using unlabeled target data. Feedback-based reinforcement learning fine-tuning methods are also seen as a promising direction for unsupervised fine-tuning.

Let the model be $f(\cdot)$, with pre-trained parameters $\theta \in \mathbb{R}^p$, learning rate α, training dataset $D_T = \{(X_i, y_i)\}_{i=1}^N$, and loss function L (whose specific form depends on the model). The fine-tuning objective is:

$$\min_{\theta_T} \frac{1}{N} \sum_{i=1}^{N} L\left(f\left(X_i; \theta_T\right), y_i\right) + \lambda R\left(\theta_T\right) \tag{1}$$

where $R(\theta_T)$ is a regularization term. To minimize the loss function for the new task, fine-tuning can be performed based on the original pre-trained parameters. Optimization algorithms, such as gradient descent, are typically used, with a small or dynamically changing learning rate, to update the fine-tuning parameters:

$$\theta_T = \theta_T - \alpha \nabla C\left(\theta_T\right) \tag{2}$$

B. Parameter Efficient Fine-Tuning

Parameter Efficient Fine-Tuning (PEFT) [27] is a technique used in natural language processing (NLP) to enhance the performance of pre-trained language models on specific downstream tasks. It involves reusing the parameters of a pre-trained model and fine-tuning it on a smaller dataset, saving computational resources and time compared to training the entire model from scratch [28].

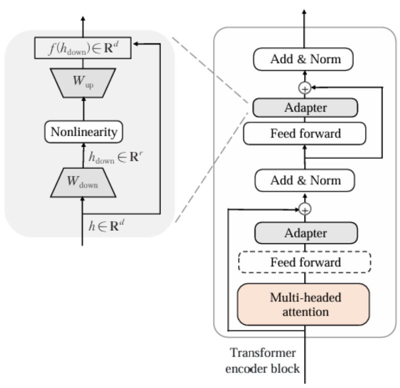

Fig. 9. Principle architecture diagramofAdapter

PEFT achieves this efficiency by freezing certain layers of the pre-trained model and fine-tuning only the last few layers specific to the downstream task. This approach allows the model to adapt to new tasks with less computational overhead and fewer labeled examples. Although PEFT is a relatively new concept, the practice of updating the final layer of models has been present in the field of computer vision since the introduction of transfer learning [29].

In 2024, the research team led by Maosong Sun from the Department of Computer Science at Tsinghua University published a significant study on ”Parameter-Efficient Fine-Tuning for Large-Scale Pre-trained Language Models” in Nature Machine Intelligence [30]. This work defines and explores the Delta Tuning problem, providing a unified framework to review and synthesize previous research. The Delta Tuning methods are categorized into three groups: addition-based, specification-based, and reparameterization-based approaches [31].

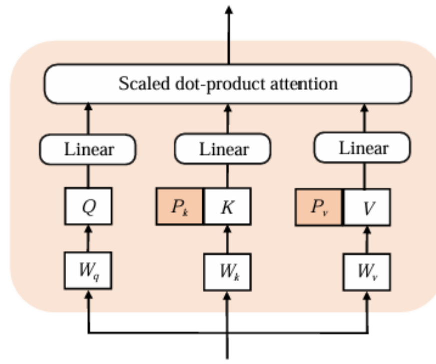

Fig. 10. Principle architecture diagram of prefix-tuning

The research team also proposed theoretical frameworks from the perspectives of optimization and optimal control to guide future structural and algorithmic designs. Their comprehensive experiments and performance analyses were conducted across more than 100 NLP tasks.

As an example of addition-based fine-tuning and selection-based fine-tuning [32], the addition-based fine-tuning method involves adding extra parameters by introducing adapter modules to each layer of the pre-trained model. By only fine-tuning the newly added adapter structures and the layer norm layers, it achieves results comparable to full-parameter fine-tuning.

In the adapter module, the input feature's feedforward sub-layer consists of a down-projection layer that maps the d-dimensional feature h to an r-dimensional feature (where $r < d$). This results in the down-projection matrix $W_{\text{down}} \in \mathbb{R}^{d\times r}$. After passing through a nonlinear layer $f(\cdot)$, a feedforward sub-layer is used to map the low-dimensional feature back to the original high-dimensional feature using the matrix $W_{\text{up}} \in \mathbb{R}^{r\times d}$. This ultimately enables parameter-efficient modifications. The process can be expressed as:

$$h \leftarrow h + f(h_{\text{down}})W_{\text{up}} \tag{3}$$

Inspired by the concept of prompts, Li et al [33]. proposed another parameter-efficient fine-tuning method called prefix-tuning. In contrast to the dimensionality reduction and restoration approach of adapter-based fine-tuning, prefix-tuning introduces and modifies prefix parameters.

Specifically, it constructs a continuous and task-related sequence of virtual tokens (prefix) before the input tokens, and during training, only the prefix for the specific task is modified. Specifically, two sets of prefix vectors, $P_k, P_v \in \mathbb{R}^{l\times d}$, are connected to the original keys K and values V in each layer of the Transformer architecture.

Then, the constructed prefixes and values are used to compute multi-head attention as follows:

$$\begin{aligned} head_i = Attn(\,& xW_q^{(i)}, \\ & concat(\,P_k^{(i)}, CW_k^{(i)}), \\ & concat(\,P_v^{(i)}, CW_v^{(i)})\,) \end{aligned} \tag{4}$$

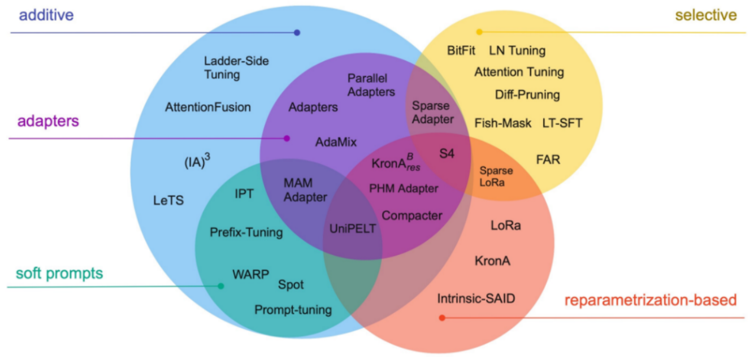

Fig. 11. Parameter Efficient Fine-Tuning

C. Prompt Tuning

Since the introduction of GPT, ELMo, and BERT, the Pre-training + Fine-tuning paradigm has been widely adopted across various natural language processing (NLP) tasks. Although this approach has outperformed traditional supervised learning methods in many cases, it also presents some challenges. For instance, when fine-tuning on most downstream tasks, there is often a significant gap between the objectives of the downstream tasks and the pre-training goals, which can result in limited performance improvements. Additionally, the fine-tuning process typically requires a large amount of labeled data.

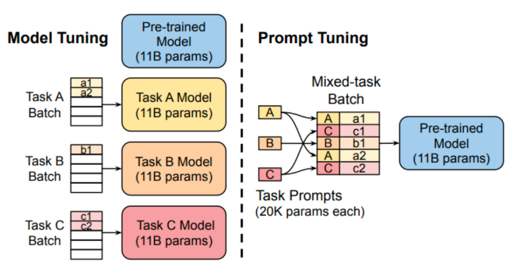

Fig. 12. The parameter efficiency of prompt tuning

To address these issues, a new fine-tuning paradigm known as Prompt Tuning, pioneered by models such as GPT-3 and PET, has emerged. This approach aims to avoid introducing additional parameters by adding prompt templates, allowing language models to achieve optimal performance in few-shot or zero-shot scenarios.

Prompt Tuning mainly addresses two issues:

- Bridging the Gap between Pre-training and Fine-tuning: Pre-training tasks primarily focus on Masked Language Modeling (MLM), while downstream tasks introduce new training parameters, resulting in significant differences between the objectives of the two stages. Therefore, it is necessary to reduce the gap between the goals of Pre-training and Fine-tuning.
- Avoiding Overfitting of the Head: During the Fine-tuning stage, additional parameters are introduced to adapt to specific task requirements, which can easily lead to overfitting when the number of training samples is limited, thereby reducing the model's generalization ability. Thus, the challenge is to address the overfitting problem associated with pre-trained language models.

Fig. 13. The three core steps of the RLHF training process

After obtaining the pre-trained model, let θ represent the model parameters, Y be a sequence representing class labels, and X be a sequence of tokens. The model's output is given by $P(Y|[X,\theta])$. Prompt tuning involves adding a pre-defined prompt P_{prompt} to the input X during the generation of Y, maximizing the probability $Pr_\theta(y_i|[P_{\text{prompt}};X,\theta]), y_i \in Y$ of obtaining the correct Y, while keeping (part of) the model parameters θ unchanged.

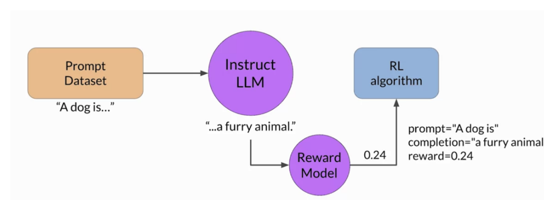

Fig. 14. Use the reward model to fine-tuning LLM with RL

Implementing Prompt Tuning only requires designing templates or prompts, while the model and training objectives are reused from the pre-training phase. Throughout the training process, no additional parameters need to be added, or only a minimal number of parameters related to the prompt are introduced, while all other parameters remain pre-trained.

D. Reinforcement Learning Fine-Tuning

In August 2017, the OpenAI team proposed the Proximal Policy Optimization (PPO) method [99], which fine-tunes

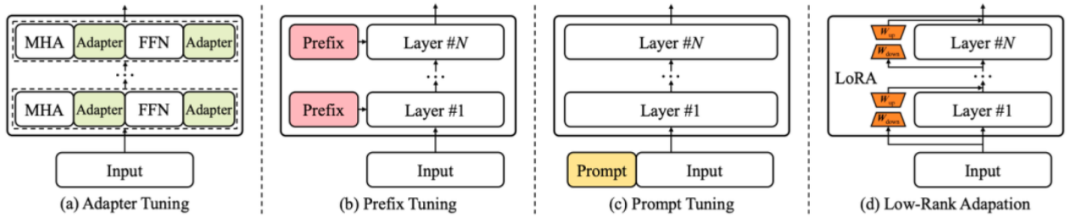

Fig. 15. Comparison of Four Fine-Tuning Techniques

models by collecting human evaluations. In each iteration, each of the N parallel participants collects data over T time steps. Then, a surrogate loss is constructed based on the $N \times T$ time steps of data, which is optimized using Stochastic Gradient Descent (SGD) over K epochs, or alternatively, the Adaptive Moment Estimation (ADAM) method can be used to achieve better performance [119]. In each iteration, the following objective is optimized using stochastic gradient ascent:

$$L_t^{\mathrm{CLIP}+VF+S}(\theta) = \hat{E}_t\left[L_t^{\mathrm{CLIP}}(\theta) - c_1 L_t^{VF}(\theta) + c_2 S\left[\pi_\theta\right](s_t)\right] \quad (5)$$

Reinforcement Learning Fine-tuning (RL Fine-tuning) is a technique used to optimize large language models by refining their behavior based on feedback. The key idea is to align the model's responses more closely with desired outputs by using a reward signal, which guides the model to produce better results over time.

The principles of reinforcement learning fine-tuning include the following four stages:

- Reinforcement Learning Basics: In the RL framework, an agent interacts with an environment. The agent takes action, and based on the feedback or reward it receives, it adjusts its behavior to maximize the cumulative reward. The goal is to learn a policy that maps states to action that yield high rewards.
- Reward Signal and Feedback: In RL Fine-tuning for language models, a reward function is designed to measure the quality of the generated outputs. This function can be based on human feedback, quality metrics, or specific criteria such as grammaticality, relevance, or factual accuracy. The reward serves as a signal that encourages the model to generate responses that align with human preferences or task-specific objectives.
- Policy Optimization: The language model's parameters are adjusted through policy optimization algorithms, such as Proximal Policy Optimization. These algorithms update the model's policy to increase the likelihood of generating high-reward outputs, improving its ability to produce desirable text responses.
- Human-in-the-loop Approach: RL Fine-tuning often involves a human-in-the-loop component, where human evaluators provide feedback on the quality of the model's responses. This feedback is used to shape the reward function, making it more reflective of human preferences and improving the model's alignment with those preferences.

RL Fine-tuning refines the behavior of pre-trained language models by iteratively improving their responses based on feedback, allowing them to perform better on specific tasks or in specialized applications.

V. Conclusion and Future Work

Large language models (LLMs) have transformed the field of natural language processing (NLP) by demonstrating remarkable capabilities in understanding and generating human-like text. Fine-tuning pre-trained language models to adapt to specific tasks has emerged as a popular research topic, playing a critical role in enhancing their adaptability and performance across various domains, including healthcare, education, and software development. Initially, fine-tuning was seen as a process of adjusting all parameters of a pre-trained model to cater to a specific task. However, applying this method to large language models often incurs substantial computational costs.

Building upon traditional full parameter tuning, researchers have developed various optimization methods for fine-tuning, focusing on reducing resource usage while enhancing performance. Parameter-efficient fine-tuning significantly reduces the computational load by modifying only a small subset of model parameters, allowing for sustained or even improved performance efficiency. Prompt tuning, as a versatile fine-tuning approach, cleverly employs prompts or indicative tags to guide the model's responses, enabling it to adapt to specific tasks with minimal parameter changes. This method has proven effective in both task-specific learning and transfer learning scenarios.

Instruction tuning further refines the model's understanding of semantics and tasks, enhancing its contextual awareness and logical reasoning capabilities. This approach helps LLMs better handle diverse tasks with explicit instructions, enabling them to respond effectively to complex directives. Additionally, integrating reinforcement learning with human feedback during the fine-tuning process aligns the model's outputs more closely with human preferences, offering promising prospects for refining model behavior in accordance with human judgment and preferences. This multifaceted approach to fine-tuning paves the way for more robust and adaptable language models capable of meeting the demands of real-world applications.

This paper primarily provides an overview of fine-tuning techniques for large language models and analyzes the principles behind several specific fine-tuning methods. The fine-tuning of large language models is a rapidly evolving field that requires ongoing research to advance methodologies, enhance computational efficiency, improve task-specific performance, increase interpretability, and strengthen the complex relationships between multimodal integration across different domains. The goal is to develop more refined large language models. Future work needs to integrate and analyze all existing fine-tuning strategies, focusing on the research of more advanced fine-tuning techniques to address critical issues in the key research areas of large language models.

References

[1] R. Tinn, H. Cheng, Y. Gu, N. Usuyama, X. Liu, T. Naumann, J. Gao, and H. Poon, "Fine-tuning large neural language models for biomedical natural language processing," Patterns, vol. 4, no. 4, Jun. 2023.

[2] D. Gao, Y. Ma, S. Liu, M. Song, L. Jin, W. Jiang, X. Wang, W. Ning, S. Yu, Q. Xuan, X. Cai, and L. Yang, "FashionGPT: LLM instruction fine-tuning with multiple LoRA-adapter fusion," Knowledge-Based Systems, vol. 299, p. 112043, 2024.

[3] L. Feng, Y. Yang, M. Tan, T. Zeng, H. Tang, Z. Li, Z. Niu, and F. Feng, "Adaptive Multi-Source Domain Collaborative Fine-Tuning for Transfer Learning," PeerJ Computer Science, vol. 10, no. e2107, 2024.

[4] J. Li, A. Sangalay, C. Cheng, Y. Tian, J. Yang, "Fine Tuning Large Language Model for Secure Code Generation," in Proceedings of the 2024 IEEE/ACM First International Conference on AI Foundation Models and Software Engineering (FORGE 2024), Lisbon, Portugal, 14 April 2024, pp. 86-90.

[5] Z. Liu, N. Zhou, M. Liu, Z. Liu, and C. Xu, "Efficient Neural Network Fine-Tuning via Layer Contribution Analysis," in Advanced Intelligent Computing Technology and Applications - 20th International Conference, ICIC 2024, Tianjin, China, Aug. 5-8, 2024, pp. 350-361. Springer, 2024.

[6] J. Lee, W. Jung, and S. Baek, "In-House Knowledge Management Using a Large Language Model: Focusing on Technical Specification Documents Review," Applied Sciences, vol. 14, no. 5, p. 2096, 2024.

[7] M. Mosbach, A. Khokhlova, M. A. Hedderich, and D. Klakow, "On the Interplay Between Fine-tuning and Sentence-level Probing for Linguistic Knowledge in Pre-trained Transformers," in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP 2020, Online Event, 16-20 November 2020, pp. 2502-2516.

[8] R. Schmirler, M. Heinzinger, and B. Rost, "Fine-tuning protein language models boosts predictions across diverse tasks," Nature Communications, vol. 15, no. 1, pp. 7407, Dec. 2024.

[9] J. Zhang, H. Yan, J. Lin, and H. Zhang, "Domain-specific large language models for fault diagnosis of heating, ventilation, and air conditioning systems by labeled-data-supervised fine-tuning," Applied Energy, vol. 377, pp. 124378, 2025.

[10] W. Zhang, Q. Wang, X. Kong, J. Xiong, S. Ni, D. Cao, B. Niu, M. Chen, Y. Li, R. Zhang, Y. Wang, L. Zhang, X. Li, Z. Xiong, Q. Shi, Z. Huang, Z. Fu, and M. Zheng, "Fine-tuning large language models for chemical text mining," Chem. Sci., vol. 15, pp. 10600-10611, 2024.

[11] A. Wang, C. Liu, J. Yang, and C. Weng, "Fine-tuning large language models for rare disease concept normalization," Journal of the American Medical Informatics Association, vol. 31, no. 9, pp. 1801–1808, 2024.

[12] H.-C. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, and D. Mollura, "Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning," IEEE Transactions on Medical Imaging, vol. 35, no. 5, pp. 1285–1298, May 2016.

[13] T. Brown, B. Mann, N. Ryder, et al., "Language Models are Few-Shot Learners," Advances in Neural Information Processing Systems, vol. 33, pp. 1877–1901, 2020.

[14] T. ValizadehAslani, Y. Shi, H. Wang, Y. Zhang, M. Hu, L. Zhao, and H. Liang, "Two-stage fine-tuning with ChatGPT data augmentation for learning class-imbalanced data," Neurocomputing, vol. 592, pp. 127801, 2024.

[15] R. K. Mahabadi, J. Henderson, and S. Ruder, "COMPACTER: Efficient Low-Rank Hypercomplex Adapter Layers," Advances in Neural Information Processing Systems, vol. 34, 2021, pp. 1022-1035.

[16] J. Liu, C. Sha, and X. Peng, "An Empirical Study of Parameter-Efficient Fine-Tuning Methods for Pre-Trained Code Models," Proceedings of the 38th IEEE/ACM International Conference on Automated Software Engineering (ASE), Luxembourg, Sep. 2023, pp. 397-408.

[17] G. W. Anderson and D. Castano, "Measure of Fine Tuning," Physics Letters B, vol. 347, no. 3-4, pp. 300-306, 1995.

[18] M. Lajkó, D. Horváth, V. Csuvik, and L. Vidács, "Fine-Tuning GPT-2 to Patch Programs, Is It Worth It?," Computational Science and Its Applications – ICCSA 2022 Workshops, Malaga, Spain, Jul. 2022, vol. 13380, Lecture Notes in Computer Science, Springer, Cham, pp. 79-91.

[19] Y. Zhong, L. Niu, M. Long, and J. Wang, "Bi-tuning: Efficient Transfer from Pre-trained Models," Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, 2023, pp. 1-15.

[20] Ansell, A., Ponti, E., Korhonen, A., and Vulić, I., "Composable Sparse Fine-Tuning for Cross-Lingual Transfer," Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Dublin, Ireland: Association for Computational Linguistics, 2022, pp. 1778-1796.

[21] B. Hartmann, P. Tamla, F. Freund, and M. L. Hemmje, "Fine-Tune it Like I'm Five: Supporting Medical Domain Experts in Training NER Models Using Cloud, LLM, and Auto Fine-Tuning," in Proc. of the 2023 Artificial Intelligence and Cloud Computing for Smart Applications Conference (AICS), 2023, pp. 1-8.

[22] S. Sledzieski, M. Kshirsagar, M. Baek, B. Berger, R. Dodhia, and J. M. Lavista Ferres, "Democratizing Protein Language Models with Parameter-Efficient Fine-Tuning," Proc. Natl. Acad. Sci., vol. 121, no. 23, Jun. 2024.

[23] A. Rattani and R. Derakhshani, "On Fine-tuning Convolutional Neural Networks for Smartphone based Ocular Recognition," in 2017 IEEE International Joint Conference on Biometrics (IJCB), Denver, CO, USA, 2017, pp. 762-767.

[24] D. Vucetic, M. Tayaranian, M. Ziaeefard, J. J. Clark, B. H. Meyer, and W. J. Gross, "Efficient Fine-Tuning of BERT Models on the Edge," in Proceedings of the 2022 IEEE International Symposium on Circuits and Systems (ISCAS), 2022, pp. 1-5.

[25] D. M. Ghilencea, R. G. Newcomb, and J. D. Wells, "The fine-tuning cost of the likelihood in SUSY models," JHEP, vol. 03, no. 021, 2017.

[26] M. Bisht and R. Gupta, "Fine-Tuned Pre-Trained Model for Script Recognition," International Journal of Mathematical, Engineering and Management Sciences, vol. 6, no. 5, pp. 1297-1314, 2021.

[27] Kasimir Forth, André Borrmann. "Semantic enrichment for BIM-based building energy performance simulations using semantic textual similarity and fine-tuning multilingual LLM." Journal of Building Engineering, 2024, Vol. 95: 110312.

[28] J. Alberto Casas, Jesús M. Moreno, Sandra Robles, and Krzysztof Rolbiecki, "Reducing the fine-tuning of gauge-mediated SUSY breaking," Eur. Phys. J. C, vol. 76, no. 450, 2016.

[29] N. Alnaasan, H. Huang, A. Shafi, H. Subramoni, and D. Panda, "Characterizing Communication in Distributed Parameter-Efficient Fine-Tuning for Large Language Models," Proceedings of the 31st Annual ACM Symposium on Architectures for Networking and Communications Systems (ANCS), 2024.

[30] N. Becherer, J. M. Pecarina, S. Nykl, and K. M. Hopkinson, "Improving optimization of convolutional neural networks through parameter fine-tuning," Neural Computing and Applications, vol. 31, no. 8, pp. 3469-3479, 2019.

[31] K. Inoue, S. Hara, and M. Abe, "Module Comparison of Transformer-TTS for Speaker Adaptation Based on Fine-Tuning," in Proceedings of the APSIPA Annual Summit and Conference, 2020, pp. 826-830.

[32] C. Pornprasit and C. Tantithamthavorn, "Fine-tuning and prompt engineering for large language models-based code review automation," Information and Software Technology, vol. 2024, p. 107523, 2024.

[33] A. Duval, T. Lamson, G. de Leseleuc de Kerouara, and M. Gallé, "Breaking Writer's Block: Low-cost Fine-tuning of Natural Language Generation Models," in Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations, Online, Apr. 19-23, 2021, pp. 278-287.