Quantum Knowledge Distillation for Large Language Models

Lingxiao Li[ac], Yihao Wang[c], Jiacheng Fan[ac], Jing Li[ac], Sujuan Qin[abc], Qiaoyan Wen[abc], Fei Gao[abc*]

[a]*State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China.*
[b]*National Engineering Research Center of Disaster Backup and Recovery, Beijing University of Posts and Telecommunications, Beijing, China.*
[c]*School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing, China.*

Abstract

Large Language Models (LLMs) are integral to advancing natural language processing, used extensively from machine translation to content creation. However, as these models scale to billions of parameters, their resource demands increase dramatically. Meanwhile, quantum computing is recognized for efficiently solving complex problems with quantum characteristics like superposition and entanglement, providing a novel approach to these challenges. This paper attempts to combine quantum computing with LLMs and proposes a Quantum knowledge Distillation algorithm for LLMs (QD-LLM), aimed at reducing the computational and memory overhead required for model loading and inference. Specifically, during the distillation stage, data is fed simultaneously into both the LLMs and the designed quantum student model to initially quantify the difference between their outputs; subsequently, with the help of the true label, the optimization of the quantum student model is executed to minimize the difference with the LLM's output. Throughout this process, only the parameters of the quantum student network are updated to make its output closer to that of the LLMs, thereby achieving the purpose of distillation. Finally, the optimized student model obtained by QD-LLM can efficiently solve domain-specific tasks during inference without the usage of the original LLMs. Experimental results show that, compared to mainstream compression methods, QD-LLM significantly reduces the number of training parameters, memory consumption, training time, and inference time while maintaining performance. Moreover, the optimized student model obtained by QD-LLM surpasses specific models designed for these tasks. We believe that QD-LLM can lay the groundwork for exploring the utilization of quantum computing in model compression and its potential extension to other natural language processing challenges.

Keywords: LLMs, Knowledge Distillation, Compression, Quantum Neural Networks, Quantum Computing

1. Introduction

The field of artificial intelligence has seen rapid advancements over the past years, leading to the emergence of numerous Large Language Models (LLMs) [1, 2, 3, 4]. These models, fueled by vast amounts of big data [5, 6, 7], have demonstrated significant advantages in language understanding across a wide range of domains, achieving consistently superior performance.

Utilizing open-source LLMs [8, 9, 10, 11] for various tasks requires loading the models for inference. However, the process of loading and running inference on LLMs incurs substantial computational and memory overhead, posing significant challenges for low-resource environments. To address this issue, extensive research has been conducted on model compression, aiming to significantly reduce the inference cost while minimizing the impact on model performance. These works can be broadly categorized based on their focus and underlying mechanisms into knowledge distillation [12, 13, 14], quantization [15, 16], and pruning [17, 18]. Knowledge distillation effectively compresses

☆This work is supported by National Natural Science Foundation of China (Grant Nos.62371069, 62372048, 62272056), the 111 Project B21049 and supported by BUPT Excellent Ph.D. Students Foundation (Grant CX20241055, CX2023120).

*Corresponding author

Email addresses: lingxiao-li@bupt.edu.cn (Lingxiao Li[ac]), yh-wang@bupt.edu.cn (Yihao Wang[c]), magic_star@bupt.edu.cn (Jiacheng Fan[ac]), leej05@126.com (Jing Li[ac]), qsujuan@bupt.edu.cn (Sujuan Qin[abc]), wqy@bupt.edu.cn (Qiaoyan Wen[abc]), gaof@bupt.edu.cn (Fei Gao[abc])

LLMs for specific tasks by designing student models and strategies [19, 20], or by fine-tuning student models directly using the outputs of teacher models [21], allowing student models to learn from the larger-scale networks. Quantization focuses on reducing the precision of model weights or activations through techniques such as matrix decomposition [15], second-order approximation [22], dynamic weight pruning [23], and adaptive precision allocation [24], effectively lowering the computational resources required and speeding up model inference [16]. Pruning reduces the complexity of large models by identifying and removing unimportant parameters [17] or modules [25], thus making the model more sparse and computationally efficient [26]. These methods have significantly reduced the computational burden of open-source LLMs while maintaining inference capability.

As LLMs continue to grow in size, there remains room for further optimization in model compression, especially in scenarios with extremely low resources or where faster inference is required. Traditional methods may struggle to meet these demands. Fortunately, extensive research has demonstrated that quantum computing can solve certain challenging problems efficiently using fewer quantum parameters [27, 28, 29, 30, 31]. This has led to the rise of Quantum Neural Networks (QNNs) [32, 33, 34], which show great potential in tasks such as classification [35] by leveraging quantum circuits to make efficient predictions with fewer parameters.

In this paper, we innovatively apply quantum computing to the distillation of LLMs and propose a novel Quantum knowledge Distillation algorithm for LLMs (QD-LLM), which utilizes the QNN to compress LLMs. In this method, a QNN designed specifically for large model knowledge distillation is proposed as the student model to distills knowledge from an LLM teacher model. During the training phase, the teacher model guides the training of the quantum circuit, allowing the predictions of the quantum circuit to gradually approach those of the teacher model. During the entire distillation process, the parameters of the quantum student network will be updated to align its output more closer with that of the LLMs, while the parameters of the LLMs remain unchanged, thus facilitating effective knowledge distillation. In the inference phase, the optimized student model obtained by QD-LLM operates independently of the LLMs, utilizing only the trained quantum circuit for efficient inference. Experiments on various tasks, such as multi-class topic analysis and binary sentiment analysis, show that QD-LLM significantly compresses LLMs while maintaining strong performance, resulting in faster inference speed and reduced memory consumption. This could potentially open new avenues for more efficient inference in language models. It should be noted that our experiments were conducted only on multiple classification tasks, as current computational capabilities do not support large-scale qubit simulations. So it is difficult to show similar advantages on complex tasks such as generation like LLMs.

The main contributions are summarized as follows.

- We propose a quantum computing-based model compression algorithm, termed the QD-LLM method. Distinct from traditional distillation methods that primarily rely on neural networks, this method applies quantum computing to the distillation of LLMs. The advantage of quantum computing lies in its ability to retain more information through the superposition and entanglement of quantum states. This enables the QD-LLM method to accelerate the distillation process while effectively enhancing the model's capacity to process and preserve information.

- We designed a loss function suitable for QD-LLM. This loss function integrates the gap between the quantum student network with the LLM teacher network and the true label, providing a direct basis for the quantum student network to better learn effective knowledge.

- We conducted extensive experiments, and the results show that using QD-LLM to distill LLMs is feasible and demonstrates superior performance. Compared with knowledge distillation baselines, QD-LLM requires significantly fewer parameters, which directly leads to significantly reduced memory usage, training, and inference time. At the same time, the optimized quantum student network inherits the advantages of the LLMs and outperforms related-task baselines in terms of performance and inference time.

2. Related Work

2.1. Large language models

LLMs [4, 36] are equipped with a vast number of parameters that enable them to comprehend text from extensive datasets and tackle various Natural Language Processing (NLP) tasks. Recent advancements in LLMs have been made through techniques like instruction fine-tuning [37, 38] and reinforcement learning via human feedback [39, 40], significantly enhancing their performance. At the same time, many researchers are focusing on building open-source LLMs [8, 9, 10, 41], to drive advancements in specific domains and tasks. LLMs are primarily composed of encoder layers and multiple decoders based on the Transformer architecture [42]. Each decoder consists of an attention layer (Attn) and a feed-forward network (FFN). Given an input T, the model first processes it through the encoder to generate the corresponding vector representation h_0, followed by several decoders. The abstraction of this process is summarized as follows:

$$h_l = \text{FFN}(h_{n'}) + h_{n'} = \text{FFN}(\text{Attn}(h_{n-1}) + h_{n-1}) + \text{Attn}(h_{n-1}) + h_{n-1}, \tag{1}$$

where, n represents the $n-th$ layer decoder,$n = 1, 2, \cdots, N$ and N is the number of decoders contained in LLMs.

2.2. *Quantum neural networks*

QNNs, hybrid models that integrates parameterized quantum circuits with classical optimizers, leverages the unique capabilities of quantum mechanics to enhance computational performance and optimization efficiency. These networks have gained momentum in various machine learning tasks, particularly in classification, due to their ability to exploit quantum characteristics for handling complex datasets. Bausch [43] et al. introduced the first quantum recurrent neural network, demonstrating its effectiveness in sequence learning and classification tasks. Du [44] et al. systematically investigated the problem-dependent power of quantum neural classifiers (QCs) on multi-class classification tasks, revealing that QCs exhibit a U-shaped risk curve and can outperform classical classifiers in certain tasks, such as parity datasets. In text classification, where a significant work is BERT Quantum Text Classifier (BERT-QTC) [45]. BERT-QTC demonstrates how quantum classifiers can enhance performance in high-dimensional feature spaces, achieving superior accuracy compared to classical classifiers in NLP tasks. Our work also draws inspiration from this model, especially how it combines quantum circuits with classical neural networks to improve performance. We referenced some of their designs for quantum layers and variational circuits when designing the quantum part of our own hybrid model.

2.3. *Knowledge distillation*

Despite the remarkable performance of LLMs, they often come with substantial computational and storage demands. As a result, model compression is essential for the practical deployment of these models on low-resource machines and for further research. One of the mainstream approaches for model compression is Knowledge Distillation [12], which compresses a large teacher model by training a smaller student model under the teacher's guidance [13]. In computational linguistics, this method either uses the teacher model's outputs at each time step as guidance [19] or fine-tunes the student model directly using the outputs from the teacher model [21, 41]. At present, some quantum distillation algorithms have been proposed. For example, Hasan et al. [46] and Li et al. [47] used the classical convolutional neural network (CNN) and classical machine learning as teachers to guide the training of the quantum student model, thereby significantly improving the performance of the quantum model; Alam et al. [48] proposed a method for enhancing QNNs through knowledge distillation. However, these algorithms are mainly targeted at small-scale models such as CNN, and have not yet explored the compression application of LLMs. In this era of explosive development of large models, the issue of using quantum computing to compress LLMs is worth exploring.

3. Preliminary

3.1. *Quantum computing: fundamental concepts*

Quantum computing is based on the principles of quantum mechanics and offers significant advantages over classical computing in terms of parallel processing and solving complex problems. Classical computers use bits as the smallest unit of information, where each bit can only exist in one of two states: 0 or 1. In contrast, quantum computers use quantum bits (qubits), which can exist in a superposition of states, allowing them to process multiple solutions simultaneously. The state of a quantum bit can be expressed as a quantum superposition:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \tag{2}$$

where, α and β are complex coefficients that satisfy the normalization condition:

$$|\alpha|^2 + |\beta|^2 = 1. \tag{3}$$

This equation implies that the quantum bit exists in a superposition of both $|0\rangle$ and $|1\rangle$ states, with $|\alpha|^2$ and $|\beta|^2$ representing the probability amplitudes of measuring the quantum bit in either of the states.

The power of quantum computing lies in its ability to utilize quantum entanglement and quantum interference. Quantum entanglement refers to the phenomenon where multiple qubits are correlated such that the state of one qubit cannot be described independently of the state of the other qubits, even when they are physically separated. Quantum algorithms such as Shor algorithm and Grover algorithm exploit the parallelism provided by quantum superposition and entanglement, enabling them to outperform classical algorithms in specific problem domains. Despite the enormous potential, quantum computing still faces many challenges, particularly related to qubit scalability, noise management, and error correction. Nevertheless, it is widely regarded as having the potential to revolutionize fields such as optimization, material science, drug discovery, and beyond. This capability enables quantum computers to evaluate many possible solutions in parallel, offering exponential speedup over classical counterparts in certain computational tasks. As a result, quantum computing holds the potential to solve complex problems that are infeasible for classical computers, particularly in fields such as optimization, cryptography, and simulation.

3.2. Quantum gates

Quantum computation relies on quantum circuits, and the core operations within these circuits are quantum gates. Quantum gates not only modify the states of quantum bits but also influence the computation process via quantum interference effects. Similar to classical logic gates in conventional computing, quantum gates are the fundamental building blocks of quantum computing. By combining different quantum gates, it is possible to perform complex computational tasks. Quantum circuits typically consist of multiple quantum gates. These gates, through entanglement and superposition of qubits, perform computations and optimize model parameters.

3.2.1. Common quantum gates

Pauli gates are a set of fundamental quantum gates, including Pauli-X, Pauli-Y, and Pauli-Z gates. They perform rotation operations on quantum bits, altering their states. These gates are widely used in QNNs, especially in the control of quantum states and the generation of quantum entanglement.

Pauli-X Gate (X Gate). The X gate is equivalent to the classical NOT gate and swaps the states $|0\rangle$ and $|1\rangle$. The matrix representation of the X gate is:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}. \tag{4}$$

In QNNs, the X gate is used to perform basic bit-flipping operations, often employed in initializing and transforming quantum states.

Pauli-Z Gate (Z Gate). The Z gate applies a phase flip to the quantum bit. It leaves $|0\rangle$ unchanged, but it introduces a negative phase when applied to $|1\rangle$. The matrix representation is:

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{5}$$

The Z gate is used in quantum computing for phase rotations and the control of quantum entanglement, where it selectively flips the phase of a quantum state.

Pauli-Y Gate (Y Gate). The Y gate performs a complex rotation, transforming $|0\rangle$ to $-i|1\rangle$ and $|1\rangle$ to $i|0\rangle$. The matrix representation is:

$$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}. \tag{6}$$

3.2.2. Hadamard Gate (H Gate)

The Hadamard gate is a crucial quantum gate, especially in QNNs. It transforms quantum bits from definite states (e.g., $|0\rangle$ or $|1\rangle$) into uniform superposition states. Specifically, the Hadamard gate maps both $|0\rangle$ and $|1\rangle$ to superposition states, allowing quantum bits to be in multiple states simultaneously. This superposition effect enables parallel computation across multiple potential states, providing significant computational power for quantum circuits. The effect of the Hadamard gate can be described as follows:

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \quad H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle). \tag{7}$$

Its matrix representation is:

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{8}$$

The Hadamard gate is used to place quantum bits into superposition, allowing quantum computation to process multiple potential states in parallel. In QNNs, the Hadamard gate is commonly used to generate complex quantum states, providing richer representations for model optimization and training, especially when dealing with high-dimensional data and solving complex optimization problems.

3.2.3. CNOT Gate (Controlled-NOT Gate)

The Controlled-NOT (CNOT) gate is a two-qubit gate, widely used in quantum computing and QNNs, particularly in the creation of quantum entanglement. The CNOT gate works by flipping the target qubit when the control qubit is in the state $|1\rangle$; otherwise, the target qubit remains unchanged. The matrix representation of the CNOT gate is:

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{9}$$

In QNNs, the CNOT gate is primarily used to create entanglement between quantum bits, which is a unique feature of quantum computing. Quantum entanglement allows QNNs to represent more complex relationships between data and enables more efficient computations during model training.

3.2.4. CZ Gate (Controlled-Z Gate)

The Controlled-Z (CZ) gate is also a two-qubit gate. It applies a Z transformation to the target qubit when the control qubit is in the state $|1\rangle$, introducing a negative phase. Its matrix representation is:

$$\text{CZ} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{10}$$

The CZ gate is commonly used in QNNs for phase control between quantum bits. It plays a crucial role in adjusting the quantum states' phases and helps quantum networks capture data features across higher dimensions. In quantum machine learning and quantum optimization, the CZ gate is used to adjust quantum states, improving performance in high-dimensional spaces.

3.3. Simulation

Currently, quantum computers are in the Noisy intermediate-scale quantum (NISQ) era because they cannot effectively simulate or perform large-scale, high-fidelity quantum computing tasks due to their limited number of qubits, insufficient error correction capabilities, and high noise levels. In this era, the most existing quantum algorithms still rely on classical computers to simulate quantum circuits.

To simulate quantum algorithms in classical computers, researchers have developed Python packages including *mindquantum*, *Qiskit*, *Cirq*, and *Torchquantum*, which support frameworks that allow researchers to simulate of quantum gates and circuits in quantum networks, making them key players in model optimization and training.

4. Methodology

4.1. *QD-LLM Overall*

Text, as one of the main forms of communication, is widely used on social networks. Influenced by factors like users' intentions and personalities, texts on social media exhibit diverse styles and themes. Sentiment and topic analysis of such content can support various applications and decisions, making it a prominent subject in computational linguistics. In this work, we use text as the research object to introduce the design of the QD-LLM method. In the proposed method, LLMs act as teacher models to guide the training of QNN, which is designated as student models. During the inference phase, QD-LLM only uses the trained quantum circuits, without requiring the LLMs, allowing for efficient inference. The overall architecture of QD-LLM is shown in Figure 1.

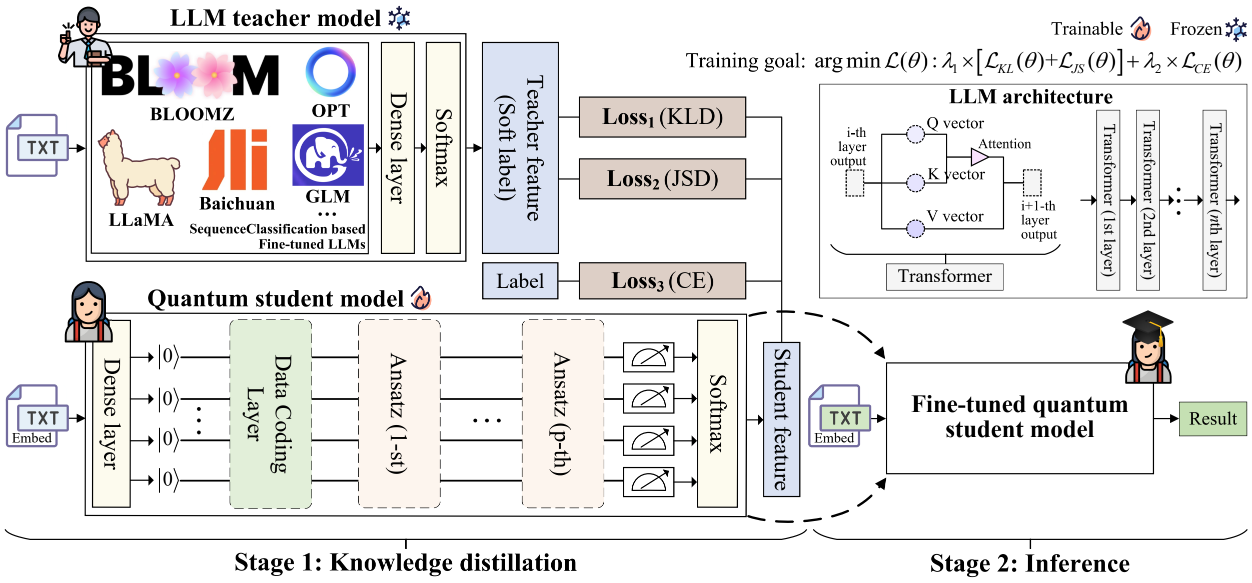

Figure 1: QD-LLM Architecture. The QD-LLM method consists of two main components: "Knowledge Distillation" and "Inference". First, we convert generative LLMs into a classification mode and fine-tune them with domain-specific data to ensure optimal performance in the target domain. The fine-tuned LLMs serve as the teacher models for distillation. During the "Knowledge Distillation" phase, domain-specific data is input into both the LLMs and the quantum student model. The output of the quantum student model is compared against the LLM output and the ground truth, measuring the combined loss. While the LLM's parameters are frozen during distillation, the quantum student model's parameters are trainable, with the objective of minimizing the combined loss. In the "Inference" phase, only the trained quantum student model is used for efficient domain-specific inference.

4.2. *Quantum distillation for LLMs*

4.2.1. *LLM teacher model*

In LLMs, given a sequence of text (token sequence) $T : w_1, w_2, \cdots, w_{t-1}$, the model can generate the probability of each token in the vocabulary $V = \{v_1, v_2, \cdots, v_n\}$ as the next token, represented by a conditional probability distribution $P(w_t|w_1, \cdots, w_{t-1})$, where n is the size of the vocabulary. This is computed as follows:

$$P(w_t|w_1, w_2, \cdots, w_{t-1}) = \text{Softmax}(LM(w_1, w_2, \cdots, w_{t-1})),\ w_t \text{ in } V. \tag{11}$$

The function $LM(\cdot)$ abstracts formula (1) and represents the complete language model calculation process, where Softmax$(\cdot)$ represents the probability conversion process. Constructing the vocabulary V is central to the operation of LLMs, and the order and size of vocabularies differ between models. To distill a generative LLM effectively, the student model must align with the teacher model's vocabulary. However, current vocabularies have reached the scale of tens of thousands, which cannot be effectively learned by the limited number of qubits in quantum models. Therefore, in this paper, we mainly study the distillation of LLMs on multiple classification tasks (e.g., sentiment analysis, subject analysis, etc.) In these tasks, we convert generative LLMs into a classification mode, i.e. given input data x, the LLMs output the probability $P(y|x)$ that x belongs to a certain class, represented as a feature vector. This transformation removes the dependency on large vocabularies, allowing effective distillation into quantum student models.

For LLMs to perform optimally on specific domains and tasks, fine-tuning is necessary using domain-specific data. However, full parameter fine-tuning of LLMs would lead to prohibitive computational costs. Therefore, this work employs the Low-Rank Adaptation (LoRA) [49] technique for efficient fine-tuning, which means that LoRA here is just to get the output of LLM teachers. LoRA modifies only a subset of the LLM's weights by adding a low-rank $\Delta\mathbf{w}$ matrix to the original weight matrix $\mathbf{W}_0 \in \mathbb{R}^{d\times k}$, $\Delta\mathbf{w}$ is determined by two low-rank (i.e., low-dimensional) matrices a and b, with the following relationship:

$$\Delta\mathbf{w} = \mathbf{a} \circ \mathbf{b}, \mathbf{a} \in \mathbb{R}^{d\times r}, \mathbf{b} \in \mathbb{R}^{r\times k}, \tag{12}$$

where, $\circ$ represents the matrix outer product, r is the rank of a and b , d and k are the linear layer dimensions and number of attention heads of the original LLMs, respectively. When resources are limited or the pre-trained knowledge is required to retain, LoRA provides a method to efficiently fine-tune large models. Finally, the original LLMs parameters and $\Delta\mathbf{w}$ are merged to obtain the final fine-tuned LLMs parameters $\mathbf{W}_{ft-LLM}$:

$$\mathbf{W}_{ft-LLM} = \mathbf{W}_0 + \Delta\mathbf{w}. \tag{13}$$

In this way, the complete LLMs to be distilled are obtained. That is, LoRA only affects the output of LLM, and the fine-tuned LLM is what really needs distillation, as shown in Formula 13.

4.2.2. Quantum student model

The student network is composed of a classical-quantum data encoding layer and an ansatz with p layers, which harness the advantages of quantum circuits in processing complex data. The classical-quantum encoding layer retains the semantic information from the classical vector, ensuring an accurate representation of the classical semantics. Meanwhile, the ansatz introduces trainable parameters, enhancing the expressiveness and flexibility of the quantum circuit. This design enables the quantum student network to capture complex semantic information from text, while leveraging quantum state evolution in high-dimensional space to improve the learning and classification capabilities of the student model.

Classical-Quantum Data Coding Layer. For the quantum circuit to process textual data, the text needs to be encoded into quantum states. Previous research [50] showed that one-hot encoding can map each token to a distinct initial quantum state, suitable for small vocabulary tasks. However, since each qubit can only represent a limited number of states, one-hot encoding becomes impractical for tasks involving large vocabularies.

In QD-LLM, an embedding layer [51] is used to map token sequences into high-dimensional semantic vectors $\mathbf{E}$. Please note: the embedding layer is not trainable and serves solely as a mapping tool. The semantic vectors are then passed through a fully connected layer for dimensionality reduction, ensuring that the final vector dimension matches the number of qubits. The dimensionality reduction is expressed by the following formula:

$$\mathbf{z} = \mathbf{w}\mathbf{E} + \mathbf{b}, \mathbf{w} \in \mathbb{R}^{n\times m}, \mathbf{E} \in \mathbb{R}^{m}, \mathbf{b} \in \mathbb{R}^{n}, \tag{14}$$

where, $\mathbf{w}$ and $\mathbf{b}$ are the weights and biases of the fully connected layer, m and n are the dimensions of $\mathbf{E}$ and the dimension after dimensionality reduction (i.e., n qubits), respectively. The reduced-dimensional vector $\mathbf{z}$ can not only express the semantics of the token, but also be embedded in the quantum circuit as a parameter to control the rotation of the qubit, thereby encoding the classical information into a quantum state for subsequent quantum circuit processing, as shown in Figure 2 a).

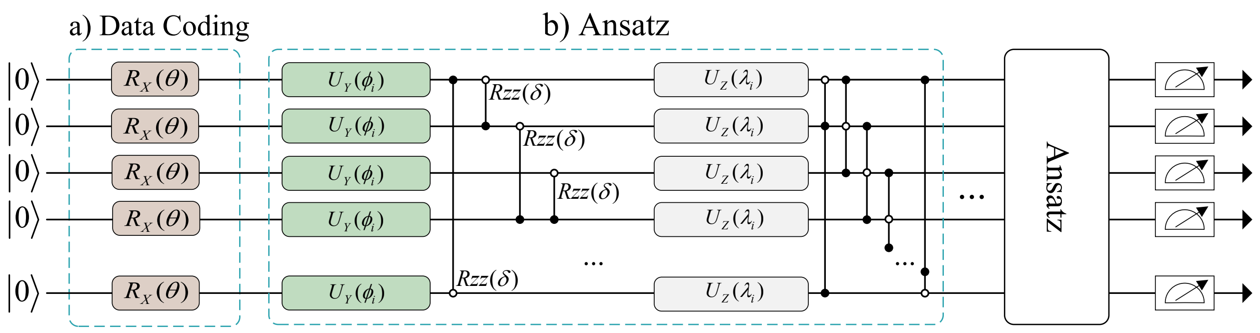

Figure 2: The structure of the parameterized quantum circuit. It presents the architecture of a complete variational quantum circuit, including both the encoding layer for classical-to-quantum data transformation and the Ansatz component. Figure 2 a) represents the encoding layer, where classical input data is encoded into quantum states through parameterized single-qubit rotation gates R_X. Figure 2 b) illustrates the Ansatz, which consists of multiple layers of parameterized rotation gates and entangling gates, used to further optimize the quantum states based on the training parameters. The overall circuit improves the expressiveness of the quantum model by stacking rotation and entanglement operations, enabling it to better capture the features of the input data.

Finally, the output of the quantum measurement is passed through a softmax function to produce the output of the student model.

Specifically, the elements of the vector serve as the rotation angles for single-qubit rotation gates R_X applied to each qubit, allowing for the classical numerical values to be encoded into quantum states. The operation of the rotation gate is described by the following equation:

$$R_X(\theta_i) = \exp\left(-i\frac{\theta_i}{2}\sigma_X\right), \tag{15}$$

where, σ_X is the Pauli-X matrix. θ_i is the *i-th* element in the semantic vector after dimension reduction, which is the rotation angle of the *i-th* qubit. The quantum state of the coding layer can be expressed as:

$$|\psi_{\text{enc}}\rangle = \prod_{i=1}^{n} R_X(\theta_i)|0\rangle^n, \tag{16}$$

The initial quantum state $|0\rangle^n$ represents all qubits in the $|0\rangle$. At this point, the classical vector has been successfully mapped into the quantum state $|\psi_{\text{enc}}\rangle$ preparing it for further quantum operations.

Ansatz Design. In the parameterized quantum circuit Ansatz designed by QD-LLM, we enhance the expressiveness of quantum circuits through a series of quantum gate operations, enabling the model to better capture and process complex input information. The parameters of these gates are optimized during the training process, thereby improving the circuit's ability to represent the features of the input data. The structure of the Ansatz is illustrated in Figure 2 b).

For a single layer of the Ansatz, each qubit q_i first undergoes a unified rotation operation $U_Y(\theta_i)$, consolidating the effects of three consecutive R_Y gates into a single more potent operation. This enhances the qubit states' ability to flexibly express the input data features. The corresponding equation for U_Y is given as follows:

$$U_Y(\phi_i) = R_Y(\phi_{i1}) + R_Y(\phi_{i2}) + R_Y(\phi_{i3}) \tag{17}$$

$$R_Y(\phi) = \exp\left(-i\frac{\phi}{2}\sigma_Y\right), \tag{18}$$

where $\phi_i = \phi_{i1} + \phi_{i2} + \phi_{i3}$ is the cumulative rotation angle for qubit q_i. Here, ϕ_{i1}, ϕ_{i2}, and ϕ_{i3} are the rotation angles corresponding to each of the three applications of the R_Y gate, and σ_Y is the Pauli-Y matrix.

Following this, we apply the R_{ZZ} gate to introduce entanglement between adjacent qubits q_i and q_j, thereby enhancing the nonlinear expression capability of the overall quantum state:

$$R_{ZZ}(\delta_{i,j}) = \exp\left(-i\frac{\delta_{i,j}}{2}\sigma_Z \otimes \sigma_Z\right), \tag{19}$$

where $\delta_{i,j}$ is the entanglement parameter between qubits q_i and q_j, and σ_Z is the Pauli-Z matrix.

Similarly, a unified rotation operation $U_Z(\lambda_i)$ is applied, which consolidates the effects of three R_Z gates:

$$U_Z(\lambda_i) = R_Z(\lambda_{i1}) + R_Z(\lambda_{i2}) + R_Z(\lambda_{i3}) \tag{20}$$

$$R_Z(\lambda) = \exp\left(-i\frac{\lambda}{2}\sigma_Z\right), \tag{21}$$

where $\lambda_i = \lambda_{i1} + \lambda_{i2} + \lambda_{i3}$ is the cumulative rotation angle for qubit q_i. Here, $\lambda_{i1}, \lambda_{i2}$, and λ_{i3} are the rotation angles corresponding to each of the three applications of the R_Z gate.

Finally, a controlled-X ($CNOT$) gate is applied to further enhance the entanglement between the qubits. For the control qubit q_i and the target qubit q_j, the $CNOT$ gate is defined as:

$$CNOT(q_i, q_j) = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes \sigma_X. \tag{22}$$

This operation conditions the state of q_j on q_i, applying an X gate (flip operation) to q_j if q_i is in the $|1\rangle$ state, and no operation if q_i is in the $|0\rangle$ state.

After these rotation and entanglement operations, the output quantum state $\left|\psi^1_{\text{ansatz}}\right\rangle$ for the first quantum layer is obtained as:

$$\left|\psi^1_{\text{ansatz}}\right\rangle = \prod_{i=1}^{n} U_Y(\phi_i) \prod_{i,j=i+1}^{n} RZZ(\delta_{i,j}) \prod_{i=1}^{n} U_Z(\lambda_i) \prod_{i,j=i+1}^{n} CNOT(q_i, q_j) \left|\psi_{\text{enc}}\right\rangle, \tag{23}$$

These operations constitute a single layer of the quantum circuit. Following this, there are $p - 1$ additional identical quantum layers, and the output quantum state $\left|\psi^k_{\text{ansatz}}\right\rangle$ is measured to obtain the result $m = \text{Measure}(\left|\psi^k_{\text{ansatz}}\right\rangle)$. The measurement results form a real-valued vector, and by applying a softmax function, the final probabilities representing the features of the quantum student model are obtained.

4.2.3. Distillation process

After obtaining the outputs from both the LLM teacher model and the quantum student model, QD-LLM need to optimize the quantum student model's output based on the LLM teacher model and the true labels. To measure the discrepancy between the quantum student model's output and the LLM teacher model's output, we employ both KL divergence and JS divergence. KL divergence quantifies the difference between two probability distributions and is commonly used in knowledge distillation tasks [52]. The symmetry of JS divergence ensures a balanced evaluation of the differences between the two models, avoiding bias toward either one, making it a fairer measure of divergence [53].

JS divergence is less sensitive to noise compared to KL divergence, which helps prevent the student model from overly relying on the potential overfitting of the teacher model. The combination of these two divergences effectively promotes the training of the student model, enabling it to not only absorb the knowledge of the teacher model but also enhance its stability and performance when faced with different tasks. Additionally, cross-entropy (CE) is used to measure the difference between the student model's output and the true labels. The addition of CE ensures that the distillation process is not just about mimicking the teacher's behavior but also guarantees the accuracy of the student model's output with respect to the true labels. This "dual optimization" (learning both the teacher's output and minimizing the discrepancy with the true labels) helps the student model improve its generalization ability in real-world applications.

The overall loss function for QD-LLM is formulated as follows:

$$\mathbb{L}(\theta) = \lambda_1 \underset{x\sim p_x}{\mathbb{E}} \left[\underbrace{\sum_i f_i(x) \log \frac{f_i(x)}{\mathbf{q}_{\theta,i}(x)}}_{\mathbb{L}_{KL}\text{ part}} + \underbrace{\frac{1}{2}\left(\sum_i f_i(x) \log \frac{f_i(x)}{m_i(x)} + \sum_i \mathbf{q}_{\theta,i}(x) \log \frac{\mathbf{q}_{\theta,i}(x)}{m_i(x)} \right)}_{\mathbb{L}_{JS}\text{ part}} \right] + \lambda_2 \underset{\substack{x\sim p_x \\ y\sim \tilde{p}(\cdot|x)}}{\mathbb{E}} \left[\underbrace{-\sum_i y_i \log \mathbf{q}_{\theta,i}(x)}_{\mathbb{L}_{CE}\text{ part}} \right], \tag{24}$$

where, λ is the assigned weight, which is: $\lambda_1 = 1 - \lambda_2$, θ is the quantum student model training parameter, $\mathbb{E}$ represents the average loss of the training batch, x is the text set of the current batch, y is the true label, and $m_i(x) =$

$\frac{1}{2}(f_i(x) + \mathbf{q}_{\theta,i}(x))$, where $\mathbf{q}_{\theta,i}(x)$ means the quantum student model's output and $f_i(x)$ means the LLM teacher model's output. Only the parameters of the quantum student model are trained during the distillation process, while the parameters of the LLM teacher model remain frozen.

4.2.4. Training

During the entire distillation process, the goal of the QD-LLM approach is to minimize the loss function, i.e., argmin $\mathbb{L}$. By minimizing this total loss function, we can adjust the parameters of the quantum student model so that its output becomes increasingly closer to both the LLM teacher model and the true labels. As the value of $\mathbb{L}$ decreases, it indicates that the performance of the quantum student model is approaching that of the LLM teacher model in the specific domain, eventually allowing the quantum student model to complete tasks independently.

4.3. Inference

During the inference phase, the quantum student model obtained by QD-LLM can independently perform tasks by processing specific domain data input into the model. The model then executes its quantum computational procedures to obtain the final prediction results. This entire inference process does not rely on LLMs. Due to the reduced number of parameters in the quantum student model, there is a significant decrease in both computational resource requirements and time consumption, making inference more efficient. Compared to traditional LLMs, the performance and efficiency advantages of the quantum student model in inference become particularly pronounced.

5. Experiments

In this section, we present a comparative analysis of QD-LLM and existing methods. To minimize the randomness inherent in the optimization process, each solving procedure is repeated five times, and the average of these five values is reported as the result. All experiments were conducted on NVIDIA GeForce RTX 4090 GPUs.

5.1. Settings

Datasets. To comprehensively evaluate the method, we selected three classic or recent research hotspot tasks in computational linguistics: **1**. Emotion analysis [54], **2**. Hiding detection [55], and **3**. Thematic analysis [56]. Emotion analysis involves understanding and extracting emotional information from text, aiding in the analysis of public opinion and intentions [57]. Hiding detection refers to concealing sensitive information within normal media, making it difficult for unauthorized individuals to detect and extract [58]. Detection of such hiding is critical for preventing misuse of this technology and safeguarding national security [59]. Thematic analysis aims to uncover the implicit themes in text, enabling theme classification and improving the effectiveness of applications such as question-answering systems [56]. The datasets contain an equal number of samples per category and are randomly divided into training, validation, and testing sets in a 6:2:2 ratio, with specific details presented in Table 1.

Table 1: Detailed information of the dataset

Datasets	Num of texts	Num of classes	Average length	Num of tokens
Emotion analysis	24,000	2	33.1704	335,101
Hiding detection	10,000	2	10.7992	53,996
Thematic analysis	20,000	4	13.9625	331,704

Baselines. The baselines in this paper are divided into two parts: compression-task baselines and related-task baselines related to the three datasets mentioned above. **Compression-task baselines**: **1**. TinyBERT [60], **2**. MiniLLM [19], **3**. DistilBERT [14], **4**. PKD [61]. **Related-task baselines**: **5**. TextRCNN [62], **6**. HiTIN [63], **7**. LSFLS [64], and **8**. BERT-QTC [45].

- TextRCNN [62] proposes a hierarchical text classification global model that better captures label dependencies and interactions between text and label spaces, achieving significant results.

- HiTIN [63] employs tree isomorphism techniques to enhance hierarchical text classification by accurately mapping label hierarchies, substantially improving classification performance.
- LSFLS [64] presents a low-shot learning framework for text steganalysis, facilitating detection of hidden information with few labeled samples. It enhances adaptability to various steganographic methods, improving performance in low-data scenarios.
- BERT-QTC [45] treats quantum circuits as feature extractors for text classification in heterogeneous computing environments, providing a new perspective on text classification tasks.

Simulation. As we all know, most existing QNNs algorithms in the NISQ era still rely on classical computers to simulate quantum circuits, as current quantum hardware lacks sufficient qubits and fault-tolerant capabilities. Thus, in our experiment, we use classical computer to simulate QD-LLM. Detailed information about the quantum part of the simulation experiment can be found in Table 2.

Table 2: Detailed information of the quantum simulation

Num of qubits	Python version	Software development kit	Measurement
11	3.8.0	mindquantum=0.9.0	Z Gate

Hyperparameters. For the teacher models, we selected some open-source LLMs, including BLOOMZ-1.1B, BLOOMZ-3B, OPT-6.7B, LLaMA2-7B, and LLaMA3-8B, with the rank r of LoRA set to 200. In the student model, the learning algorithm used is Adam [65], with an initial learning rate of 0.06, guidance weight w at 0.9, and the number of epochs set to 10, with a batch_size of 8.

5.2. Comparison with compression-task baselines on parameters

We use the number of model parameters to measure the memory consumption of the method. The specific results are shown in Table 3.

Table 3: Comparison with original LLMs and knowledge distillation baselines on parameters. The units are Billion (B) and Million (M), and there are: 1B=1,000M. "LoRA $_{\text{(LLMs)}}$" represents the number of parameters of the LoRA matrix of LLMs. "$\downarrow$" means the lower the corresponding value, the better. "Proportion" represents the proportion of the corresponding distillation method with the largest number of parameters (PKD) among all distillation baselines. "**Bold**" represents the best result. "_*_" represents the suboptimal result.

Model	Param $\downarrow$	Model	Param $\downarrow$
BLOOMZ-1.1B	1.1B	LoRA $_{\text{(BLOOMZ-1.1B)}}$	29.50M
BLOOMZ-3B	3B	LoRA $_{\text{(BLOOMZ-3B)}}$	61.46M
OPT-6.7B	6.7B	LoRA $_{\text{(OPT-6.7B)}}$	104.89M
LLaMA2-7B	7B	LoRA $_{\text{(LLaMA2-7B)}}$	157.32M
LLaMA3-8B	8B	LoRA $_{\text{(LLaMA3-8B)}}$	118.00M

Methods	Param $\downarrow$	Proportion $\downarrow$
TinyBERT [60]	<u>14.35M*</u>	27.49%
MiniLLM [19]	33.36M	63.91%
DistilBERT [14]	52.20M	100.00%
PKD [61]	52.20M	100.00%
Ours	**9,275**	0.02%

From the results in Table 3, we can see that QD-LLM has excellent performance in terms of the number of model parameters. Its parameter volume is only 0.2%-0.6% of the existing compression methods, which greatly reduces the model size. Such low memory usage makes our model particularly suitable for deployment on memory-constrained devices.

5.3. *Comparison with compression-task baselines on time cost*

For resource consumption, we measure knowledge distillation time (training per epoch) and inference time of QD-LLM and baselines. The specific results are shown in Figure 3.

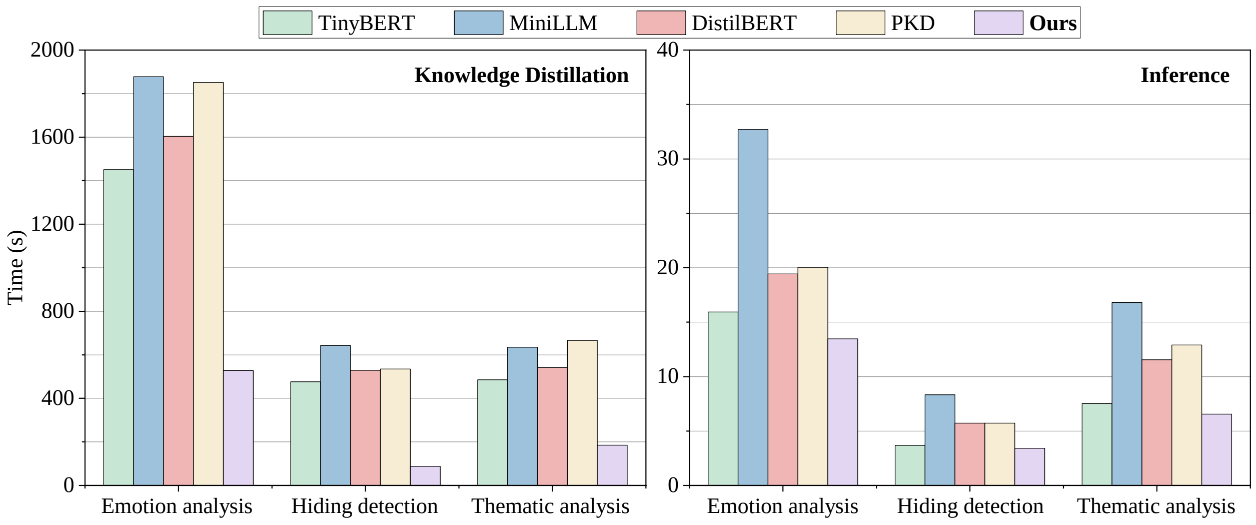

Figure 3: Comparison with original LLMs and knowledge distillation baselines on the knowledge distillation time and inference time. The horizontal axis represents the datasets, the vertical axis represents the time cost, and its unit is second.

According to the results in Figure 3, we can see that in the three datasets, the proposed QD-LLM has shorter training and inference time than the existing knowledge distillation baseline method. This shows that the QD-LLM simulated by classical computers has faster processing capabilities than the baseline method, which also proves that quantum computing has immeasurable potential in knowledge distillation. Furthermore, the difference in inference time of tasks comes from the length of the text. Since the length of the text determines the sequence length tensor in the input, teacher network, and quantum student network, it increases exponentially with the increase of length. Therefore, datasets with long texts are more resource-intensive.

5.4. *Comparison with compression-task baselines on performance*

For performance evaluation, we use **1**. Accuracy (Acc), **2**. Precision (P), **3**. Recall (R), and **4**. F1 score (F1) as metrics. These formulas are as follows:

$$\mathrm{Acc} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}, \tag{25}$$

$$\mathrm{P} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}, \tag{26}$$

$$\mathrm{R} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \tag{27}$$

$$\mathrm{F1} = 2 \times (\mathrm{P} \times \mathrm{R})/(\mathrm{P} + \mathrm{R}), \tag{28}$$

where, TP (True Positive) indicates that it correctly predicts the positive class when the sample is actually positive, FP (False Positive) indicates that it incorrectly predicts the positive class when the sample is actually negative, TN (True Negative) indicates that it correctly predicts the negative class when the sample is actually negative, and FN (False Negative) indicates that it incorrectly predicts the negative class when the sample is actually positive are standard terms in classification metrics.

In addition, to balance the performance and the cost, that is, the required model parameters or knowledge distillation time, we also give two comprehensive indicators: **5**. Acc/Parameters and **6**. Acc/Tkd (knowledge distillation time). In other words, the advantages of the method cannot be judged only by the speed of knowledge distillation or performance, but also by comprehensive consideration of these aspects. These specific performances are shown in Table 4, Table 5, and Table 6.

Table 4: Comparison on performance in **Emotion analysis** dataset. "↑" means the higher the corresponding value, the better. "**Bold**" represents the best result. "_*" represents the suboptimal result. "Acc/Param" represents the benefits that each parameter can bring to the Acc. "Acc/Tkd" represents the benefits that one second distillation time can bring to the Acc. "e-η" represents $1 \times 10^{\eta}$ in scientific notation.

Emotion analysis		Acc ↑	P ↑	R ↑	F1 ↑	Acc/Param ↑	Acc/Tkd ↑
BLOOMZ-1.1B	+TinyBERT	0.7781	0.7615	0.7940	0.7774	5.42e-8*	1.17e-3*
	+MiniLLM	0.8110	0.7945	0.8267	0.8103	2.43e-8	7.12e-4
	+DistilBERT	0.7542	0.7351	0.7678	0.7511	1.45e-8	3.78e-4
	+PKD	0.7661	0.7492	0.7821	0.7653	1.47e-8	3.54e-4
	+Ours	0.7802	0.7654	0.7956	0.7809	**8.42e-5**	**1.68e-3**
BLOOMZ-3B	+TinyBERT	0.7843	0.7675	0.8000	0.7834	5.47e-8*	5.26e-4*
	+MiniLLM	0.8070	0.7920	0.8249	0.8081	2.42e-8	5.20e-4
	+DistilBERT	0.7535	0.7400	0.7740	0.7566	1.44e-8	1.90e-4
	+PKD	0.7701	0.7529	0.7856	0.7689	1.48e-8	2.21e-4
	+Ours	0.7923	0.7782	0.8100	0.7938	**8.54e-5**	**1.67e-3**
OPT-6.7B	+TinyBERT	0.7943	0.7805	0.8094	0.7947	5.54e-8*	5.12e-4*
	+MiniLLM	0.8130	0.7985	0.8347	0.8162	2.44e-8	3.56e-4
	+DistilBERT	0.7591	0.7420	0.7755	0.7584	1.45e-8	2.07e-4
	+PKD	0.7739	0.7542	0.7869	0.7702	1.48e-8	1.99e-4
	+Ours	0.8157	0.7913	0.8225	0.8066	**8.79e-5**	**1.39e-3**
LLaMA2-7B	+TinyBERT	0.7977	0.7831	0.8131	0.7978	5.56e-8*	4.99e-4*
	+MiniLLM	0.8136	0.8020	0.8388	0.8200	2.44e-8	3.71e-4
	+DistilBERT	0.7622	0.7440	0.7792	0.7612	1.46e-8	2.00e-4
	+PKD	0.7805	0.7620	0.7985	0.7798	1.50e-8	1.61e-4
	+Ours	0.8105	0.7965	0.8308	0.8133	**8.74e-5**	**1.50e-3**
LLaMA3-8B	+TinyBERT	0.7946	0.7855	0.8130	0.7990	5.54e-8*	4.07e-4*
	+MiniLLM	0.8177	0.7995	0.8361	0.8174	2.45e-8	3.68e-4
	+DistilBERT	0.7648	0.7435	0.7785	0.7606	1.47e-8	1.67e-4
	+PKD	0.7894	0.7657	0.8009	0.7829	1.51e-8	1.62e-4
	+Ours	0.8072	0.7940	0.8281	0.8107	**8.70e-5**	**1.42e-3**
Overall	Param proportion ↓	Acc ↑	P ↑	R ↑	F1 ↑	Acc/Param ↑	Acc/Tkd ↑
TinyBERT	27.49%*	0.7898	0.7756	0.8059	0.7905	5.51e-8*	6.24e-4*
MiniLLM	63.91%	**0.8125**	**0.7973**	**0.8322**	**0.8144**	2.44e-8	4.65e-4
DistilBERT	100.00%	0.7588	0.7409	0.7750	0.7576	1.45e-8	2.28e-4
PKD	100.00%	0.7760	0.7568	0.7908	0.7734	1.49e-8	2.19e-4
Ours	**0.02%**	0.8012*	0.7851*	0.8174*	0.8011*	**8.64e-5**	**1.53e-3**

Table 5: Comparison on performance in **Hiding detection** dataset. "↑" means the higher the corresponding value, the better. "**Bold**" represents the best result. "_*" represents the suboptimal result. "Acc/Param" represents the benefits that each parameter can bring to the Acc. "Acc/Tkd" represents the benefits that one second distillation time can bring to the Acc. "e-η" represents $1 \times 10^{\eta}$ in scientific notation.

Hiding detection		Acc ↑	P ↑	R ↑	F1 ↑	Acc/Param ↑	Acc/Tkd ↑
BLOOMZ-1.1B	+TinyBERT	0.7901	0.7754	0.8079	0.7913	5.51e-8*	2.35e-3*
	+MiniLLM	0.8140	0.8010	0.8397	0.8199	2.44e-8	1.81e-3
	+DistilBERT	0.7687	0.7593	0.7918	0.7752	1.47e-8	7.88e-4
	+PKD	0.7785	0.7621	0.7705	0.7791	1.49e-8	7.69e-4
	+Ours	0.7933	0.7833	0.8143	0.7985	**8.55e-5**	**9.79e-3**
BLOOMZ-3B	+TinyBERT	0.7933	0.7742	0.8067	0.7901	5.53e-8*	2.36e-3*
	+MiniLLM	0.8188	0.7945	0.8309	0.8123	2.45e-8	1.55e-3
	+DistilBERT	0.7769	0.7635	0.7954	0.7791	1.49e-8	7.45e-4
	+PKD	0.7833	0.7635	0.7720	0.7807	1.50e-8	7.74e-4
	+Ours	0.8093	0.7956	0.8306	0.8127	**8.73e-5**	**9.67e-3**
OPT-6.7B	+TinyBERT	0.7953	0.7837	0.8155	0.7993	5.54e-8*	1.36e-3*
	+MiniLLM	0.8177	0.8030	0.8407	0.8214	2.45e-8	1.10e-3
	+DistilBERT	0.7876	0.7655	0.7978	0.7813	1.51e-8	4.45e-4
	+PKD	0.7894	0.7656	0.7743	0.7831	1.51e-8	4.47e-4
	+Ours	0.8231	0.8050	0.8420	0.8231	**8.87e-5**	**8.94e-3**
LLaMA2-7B	+TinyBERT	0.7995	0.7735	0.8273	0.7886	5.50e-8*	1.52e-3*
	+MiniLLM	0.8217	0.8090	0.8506	0.8293	2.46e-8	1.11e-3
	+DistilBERT	0.7759	0.7640	0.7956	0.7795	1.49e-8	4.94e-4
	+PKD	0.7851	0.7651	0.7738	0.7826	1.50e-8	4.96e-4
	+Ours	0.8201	0.8051	0.8419	0.8231	**8.84e-5**	**8.97e-3**
LLaMA3-8B	+TinyBERT	0.8007	0.7862	0.8157	0.8021	5.59e-8*	1.34e-3*
	+MiniLLM	0.8208	0.7990	0.8375	0.8178	2.46e-8	1.09e-3
	+DistilBERT	0.7852	0.7660	0.7993	0.7823	1.50e-8	4.41e-4
	+PKD	0.7896	0.7730	0.7819	0.7911	1.51e-8	4.39e-4
	+Ours	0.8159	0.8013	0.8354	0.8180	**8.80e-5**	**8.84e-3**

Overall	Param proportion ↓	Acc ↑	P ↑	R ↑	F1 ↑	Acc/Param ↑	Acc/Tkd ↑
TinyBERT	27.49%*	0.7958	0.7786	0.8146	0.7943	5.53e-8*	1.78e-3*
MiniLLM	63.91%	**0.8186**	**0.8013**	**0.8399**	**0.8201**	2.45e-8	1.33e-3
DistilBERT	100.00%	0.7789	0.7637	0.7960	0.7795	1.49e-8	5.82e-4
PKD	100.00%	0.7852	0.7659	0.7745	0.7833	1.50e-8	5.85e-4
Ours	**0.02%**	0.8123*	0.7981*	0.8328*	0.8151*	**8.76e-5**	**9.24e-3**

Table 6: Comparison on performance in **Thematic analysis** dataset. “↑” means the higher the corresponding value, the better. “**Bold**” represents the best result. “*” represents the suboptimal result. “Acc/Param” represents the benefits that each parameter can bring to the Acc. “Acc/Tkd” represents the benefits that one second distillation time can bring to the Acc. “e-η” represents $1 \times 10^{\eta}$ in scientific notation.

Thematic analysis		Acc ↑	P ↑	R ↑	F1 ↑	Acc/Param ↑	Acc/Tkd ↑
BLOOMZ-1.1B	+TinyBERT	0.7883	0.7729	0.8135	0.7927	5.49e-8*	2.65e-3*
	+MiniLLM	0.8409	0.8235	0.8605	0.8416	2.52e-8	1.87e-3
	+DistilBERT	0.7727	0.7490	0.7832	0.7657	1.48e-8	8.59e-4
	+PKD	0.7827	0.7616	0.7699	0.7783	1.50e-8	8.96e-4
	+Ours	0.8497	0.8202	0.8730	0.8458	**9.16e-5**	**6.48e-3**
BLOOMZ-3B	+TinyBERT	0.7888	0.7731	0.8133	0.7927	5.50e-8*	1.84e-3*
	+MiniLLM	0.8485	0.8375	0.8867	0.8614	2.54e-8	1.65e-3
	+DistilBERT	0.7627	0.7520	0.7864	0.7688	1.46e-8	5.87e-4
	+PKD	0.7769	0.7628	0.7711	0.7795	1.49e-8	5.97e-4
	+Ours	0.8541	0.8310	0.8845	0.8569	**9.21e-5**	**6.13e-3**
OPT-6.7B	+TinyBERT	0.7957	0.7776	0.8119	0.7944	5.54e-8*	1.43e-3*
	+MiniLLM	0.8167	0.7880	0.8236	0.8054	2.45e-8	1.12e-3
	+DistilBERT	0.7757	0.7550	0.7906	0.7724	1.49e-8	4.91e-4
	+PKD	0.7803	0.7655	0.7741	0.7828	1.49e-8	4.55e-4
	+Ours	0.8541	0.8432	0.8947	0.8682	**9.21e-5**	**6.09e-3**
LLaMA2-7B	+TinyBERT	0.7960	0.7784	0.8125	0.7951	5.55e-8*	1.43e-3*
	+MiniLLM	0.8621	0.8380	0.8870	0.8618	2.58e-8	1.09e-3
	+DistilBERT	0.7860	0.7613	0.7980	0.7792	1.51e-8	4.68e-4
	+PKD	0.7896	0.7664	0.7751	0.7839	1.51e-8	4.68e-4
	+Ours	0.8670	0.8433	0.8946	0.8682	**9.35e-5**	**3.74e-3**
LLaMA3-8B	+TinyBERT	0.7902	0.7665	0.8038	0.7847	5.51e-8*	1.35e-3*
	+MiniLLM	0.8630	0.8245	0.8842	0.8533	2.59e-8	1.25e-3
	+DistilBERT	0.7782	0.7650	0.8054	0.7847	1.49e-8	4.43e-4
	+PKD	0.7915	0.7680	0.7767	0.7855	1.52e-8	4.52e-4
	+Ours	0.8654	0.8357	0.8885	0.8613	**9.33e-5**	**3.05e-3**
Overall	Param proportion ↓	Acc ↑	P ↑	R ↑	F1 ↑	Acc/Param ↑	Acc/Tkd ↑
TinyBERT	27.49%*	0.7918	0.7737	0.8110	0.7919	5.52e-8*	1.74e-3*
MiniLLM	63.91%	0.8462*	0.8223*	0.8684*	0.8447*	2.54e-8	1.40e-3
DistilBERT	100.00%	0.7751	0.7565	0.7927	0.7742	1.49e-8	5.70e-4
PKD	100.00%	0.7842	0.7649	0.7734	0.7820	1.50e-8	5.73e-4
Ours	**0.02%**	**0.8581**	**0.8347**	**0.8871**	**0.8601**	**9.25e-5**	**5.10e-3**

According to the results in Table 4 to Table 6, we can see that in the three datasets, the proposed QD-LLM method has generally excellent performance. In addition, based on the comparison of the two constructed comprehensive evaluation indicators, it can be found that the QD-LLM method can bring excellent performance at a lower cost, which also shows the potential advantages of quantum computing in the field of knowledge distillation. Furthermore, the QD-LLM method performs best on the thematic analysis dataset and second on the hiding detection dataset. Compared with the binary classification, thematic analysis is a more challenging four-class classification problem, which shows that QD-LLM has greater potential in more complex classification tasks.

In conclusion, QD-LLM not only surpasses baselines in memory usage and inference speed but also exhibits outstanding adaptability and efficiency in complex multitasking scenarios. This demonstrates the advantages of our proposed student model in knowledge distillation.

5.5. *Comparison with related-task baselines*

When compared with leading specific designed baselines across various fields, student model obtained by QD-LLM demonstrates exceptional performance, particularly in the core tasks of sentiment analysis, steganalysis, and topic analysis. Despite the strong performance of baseline methods within their respective domains, the student model obtained by QD-LLM shows the ability to surpass these algorithms, achieving significant improvements in accuracy and F1 scores. This indicates that the QD-LLM-optimized student model not only possesses the capability to independently complete complex tasks but also exhibits excellent cross-task generalization, consistently outperforming existing leading algorithms.

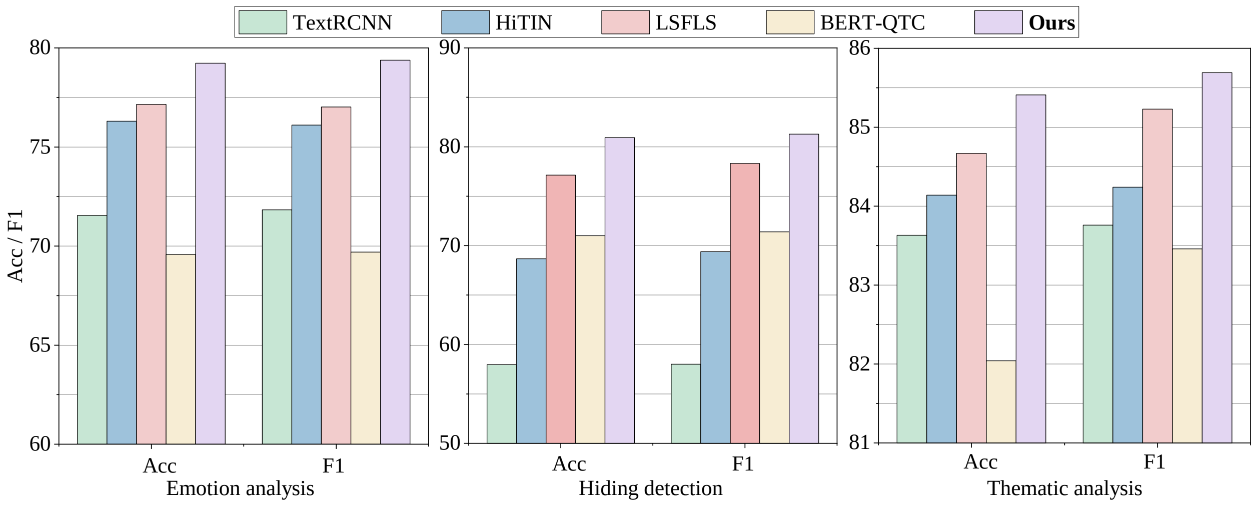

Figure 4: Comparison with related-task baselines. The horizontal axis represents the Acc and F1, and the vertical axis represents the performance (%).

The results show that QD-LLM is an effective knowledge distillation method by two aspects:

First, as a knowledge distillation method, QD-LLM showcases outstanding performance. Specifically, it not only achieves significant results in model compression but also excels in memory usage, inference speed, and training efficiency. Compared to other knowledge distillation methods, QD-LLM maintains performance comparable to or even exceeding that of larger models while significantly reducing resource consumption, which is its core advantage.

Second, the student model obtained by QD-LLM is capable of completing tasks independently and exhibits strong generalization across different tasks. Whether in sentiment analysis, steganalysis, or topic analysis, the student model outperforms existing baseline methods, showing notable improvements in accuracy and F1 scores.

5.6. *Ablation experiments*

5.6.1. *Ablation of the different loss functions*

We also explored the impact of different loss functions on the performance. We conducted ablation experiments on the loss function of the proposed QD-LLM method in the five LLMs and three tasks in Tables 4 to 6.

In order to clearly and intuitively show the performance benefits brought by different loss functions, we integrated the data and obtained the final results, as shown in Table 7.

Table 7: Ablation of the different loss functions. "CE" is the sum of labels for interactive optimization model, "KL" and "JS" are the sum of teacher models for interactive optimization model, "KL+JS+CE" is the loss function of the complete QD-LLM method. The specific content is shown in "Section 4.2.3". Here we present the average results in different datasets and LLMs. "**Bold**" represents the best result.

	Acc	F1
CE	0.7388	0.7401
KL	<u>0.8195*</u>	<u>0.8217*</u>
JS	0.8156	0.8209
KL+JS+CE (QD-LLM)	**0.8242**	**0.8254**

The results in Table 7 show that QD-LLM requires different loss functions to comprehensively optimize the quantum student model, so that the quantum student model can be closer to the performance of the LLM teacher model.

5.6.2. *Ablation of the different LLMs*

In addition, since the distilled LLMs are also a variable in the proposed method, we also conducted ablation experiments under different LLMs. The specific results are shown in Table 8.

Table 8: Ablation experiments under different LLMs. "**Bold**" represents the best result.

QD-LLM		BLOOMZ-1.1B	BLOOMZ-3B	OPT-6.7B	LLaMA2-7B	LLaMA3-8B
Emotion analysis	Acc	0.7802	0.7923	0.8157	0.8105	0.8072
	P	0.7654	0.7782	0.7913	0.7965	0.7940
	R	0.7956	0.8100	0.8225	0.8308	0.8281
	F1	0.7809	0.7938	0.8066	0.8133	0.8107
Hiding detection	Acc	0.7933	0.8093	0.8231	0.8201	0.8159
	P	0.7833	0.7956	0.8050	0.8051	0.8013
	R	0.8143	0.8306	0.8420	0.8419	0.8354
	F1	0.7985	0.8127	0.8231	0.8231	0.8180
Thematic analysis	Acc	0.8497	0.8541	0.8541	0.8670	0.8654
	P	0.8202	0.8310	0.8432	0.8433	0.8357
	R	0.8730	0.8845	0.8947	0.8946	0.8885
	F1	0.8458	0.8569	0.8682	0.8682	0.8613
Overall	Acc	0.8077	0.8186	0.8310	**0.8325**	0.8295
	P	0.7896	0.8016	0.8132	**0.8150**	0.8103
	R	0.8276	0.8417	0.8531	**0.8558**	0.8507
	F1	0.8084	0.8211	0.8326	**0.8349**	0.8300

From the results in Table 8, it can be found that the overall performance of the proposed QD-LLM method shows an increasing trend as the scale of the LLM teacher model increases. This is because larger-scale LLMs can provide better knowledge for the quantum student model, so that after fine-tuning, it can more effectively perform reasoning tasks in this field alone. Furthermore, this increasing trend does not always exist, because the scale of the quantum student model is fixed and it has a performance upper limit. When the quantum student model reaches the performance upper limit, it is difficult to improve the performance of the quantum student model only by increasing the scale of LLMs. Therefore, it is necessary to increase the scale of quantum bits and quantum student models in order to further improve their performance.

6. Conclusion

As the scale of LLMs continues to expand, there remains room for optimization in model compression. This paper presents a novel approach to distilling LLMs using quantum algorithms, called QD-LLM. QD-LLM consists of an LLM teacher model and a quantum student model, with the quantum student simulating the distribution of outputs from LLMs and the true results. During the inference phase, the trained quantum student model can efficiently handle various tasks. Experimental results confirm the feasibility of this method, with the quantum student network demonstrating reduced parameter count, lower memory usage, and improved inference efficiency compared to baselines, while also achieving superior model performance. Compared to mainstream knowledge distillation baselines and the original LLMs, QD-LLM significantly reduces the number of training parameters, memory consumption, and inference time while maintaining performance, thereby alleviating the challenges of applying LLMs in resource-constrained environments.

It is noteworthy that QD-LLM has currently only been tested on multiple classification tasks. Due to the limitations of traditional computers in simulating large-scale qubits, it is difficult to demonstrate similar advantages in generative tasks and others. Furthermore, the performance of QD-LLM is not the upper limit of quantum algorithms for compressing LLMs. By increasing the number of qubits, optimizing quantum circuit design, and improving the distillation process, further enhancements in distillation effectiveness may be achieved. In the future, improving the compression effects of LLMs and enhancing the performance of the quantum student network are worthwhile research directions. Furthermore, we hope that QD-LLM can provide valuable insights into exploring the construction of quantum distillation methods for other tasks.

7. Discussion

Although the proposed QD-LLM method can distill large-scale language models well and perform well on multiple datasets, they are still essentially classification tasks. We tried more complex task types, including language understanding and generation tasks, but found that the QD-LLM method performed poorly on these tasks. We believe that this may be due to the characteristics of the generation task requiring a mechanism similar to the LLMs, that is, building a quantum vocabulary to generate quantum states by encoding tokens. However, since existing classical computers have difficulty simulating large-scale quantum bits, it is impossible to effectively encode such a quantum vocabulary.

8. Future Work

The performance of QD-LLM is not unlimited. In other words, increasing the size of the teacher model does not necessarily distill a better QD-LLM. This is because the performance of QD-LLM depends on the nonlinear fitting ability of the quantum student network, which is limited by the number of quantum bits. If the number of quantum bits is not increased and the compression level is only improved by increasing the size of the teacher model, the performance improvement of the quantum student model will become extremely limited. In the next research, we will try to improve the performance ceiling of QD-LLM by changing the structure of quantum neural networks or exploring how to run large-scale quantum bits. In addition, how to achieve efficient encoding under the condition of a limited number of quantum bits so that the quantum network can be competent for generation tasks is the focus of future research.

Acknowledgment

Thank the editor and reviewers of "*Information Fusion*" for their review and constructive suggestions.

Declaration of competing interest

The authors declare no conflict of interest.

Funding

This work is supported by National Natural Science Foundation of China (Grant Nos.62371069, 62372048, 62272056), the 111 Project B21049 and supported by BUPT Excellent Ph.D. Students Foundation (Grant CX20241055, CX2023120).

Author contributions

Lingxiao Li: Conceptualization; Data curation; Investigation; Methodology; Software; original draft. Yihao Wang: Conceptualization; Investigation; Resources; Software; original draft. The first two authors contributed equally to this work. Jiacheng Fan: Formal analysis; Investigation; Resources; Software. Jing Li: Original draft; review & editing; Validation. Sujuan Qin: Funding acquisition; Supervision. Qiaoyan Wen: Formal analysis; review & editing. Fei Gao: Funding acquisition; Investigation; Project administration; Supervision; review & editing.

References

[1] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al., Llama: Open and efficient foundation language models, arXiv preprint arXiv:2302.13971 (2023).

[2] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al., Llama 2: Open foundation and fine-tuned chat models, arXiv preprint arXiv:2307.09288 (2023).

[3] T. GLM, A. Zeng, B. Xu, B. Wang, C. Zhang, D. Yin, D. Rojas, G. Feng, H. Zhao, H. Lai, et al., Chatglm: A family of large language models from glm-130b to glm-4 all tools, arXiv preprint arXiv:2406.12793 (2024).

[4] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al., Gpt-4 technical report, arXiv preprint arXiv:2303.08774 (2023).

[5] M. S. Mahmud, J. Z. Huang, S. García, Clustering approximation via a fusion of multiple random samples, Information Fusion 101 (2024) 101986.

[6] M. S. Mahmud, J. Z. Huang, R. Ruby, A. Ngueilbaye, K. Wu, Approximate clustering ensemble method for big data, IEEE Transactions on Big Data 9 (4) (2023) 1142–1155.

[7] Y. Cai, J. Z. Huang, A. Ngueilbaye, X. Sun, Adaptive neighbors graph learning for large-scale data clustering using vector quantization and self-regularization, Applied Soft Computing 167 (2024) 112256.

[8] A. Meta, Introducing meta llama 3: The most capable openly available llm to date, Meta AI (2024).

[9] A. Yang, B. Xiao, B. Wang, B. Zhang, C. Bian, C. Yin, C. Lv, D. Pan, D. Wang, D. Yan, et al., Baichuan 2: Open large-scale language models, arXiv preprint arXiv:2309.10305 (2023).

[10] N. Muennighoff, T. Wang, L. Sutawika, A. Roberts, S. Biderman, T. L. Scao, M. S. Bari, S. Shen, Z.-X. Yong, H. Schoelkopf, et al., Crosslingual generalization through multitask finetuning, arXiv preprint arXiv:2211.01786 (2022).

[11] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin, et al., Opt: Open pre-trained transformer language models, arXiv preprint arXiv:2205.01068 (2022).

[12] G. Hinton, Distilling the knowledge in a neural network, Proceedings of the Deep Learning Workshop of NIPS 2014 (2015).

[13] J. Gou, B. Yu, S. J. Maybank, D. Tao, Knowledge distillation: A survey, International Journal of Computer Vision 129 (6) (2021) 1789–1819.

[14] S. Victor, D. Lysandre, C. Julien, T. Wolf, Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, Proceedings of 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS 2019 (2019).

[15] Y. Xu, X. Han, Z. Yang, S. Wang, Q. Zhu, Z. Liu, W. Liu, W. Che, Onebit: Towards extremely low-bit large language models, Proceedings of the Conference on Neural Information Processing Systems (2024).

[16] A. Tseng, J. Chee, Q. Sun, V. Kuleshov, C. De Sa, Quip#: Even better llm quantization with hadamard incoherence and lattice codebooks, Proceedings of the International Conference on Machine Learning (2024).

[17] Y. Zhang, Y. Li, X. Wang, Q. Shen, B. Plank, B. Bischl, M. Rezaei, K. Kawaguchi, Finercut: Finer-grained interpretable layer pruning for large language models, Proceedings of the Compression Worshop at NeurIPS 2024 (2024).

[18] M. Xia, T. Gao, Z. Zeng, D. Chen, Sheared llama: Accelerating language model pre-training via structured pruning, Proceedings of the Twelfth International Conference on Learning Representations (2024).

[19] Y. Gu, L. Dong, F. Wei, M. Huang, Minillm: Knowledge distillation of large language models, in: Proceedings of the Twelfth International Conference on Learning Representations, 2024.

[20] G. Cui, L. Yuan, N. Ding, G. Yao, B. He, W. Zhu, Y. Ni, G. Xie, R. Xie, Y. Lin, et al., Ultrafeedback: Boosting language models with scaled ai feedback, in: Forty-first International Conference on Machine Learning, 2024.

[21] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, T. B. Hashimoto, Alpaca: A strong, replicable instruction-following model, Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html 3 (6) (2023) 7.

[22] E. Frantar, S. Ashkboos, T. Hoefler, D. Alistarh, Gptq: Accurate post-training quantization for generative pre-trained transformers, Proceedings of the Eleven International Conference on Learning Representations (2022).

[23] W. Shao, M. Chen, Z. Zhang, P. Xu, L. Zhao, Z. Li, K. Zhang, P. Gao, Y. Qiao, P. Luo, Omniquant: Omnidirectionally calibrated quantization for large language models, Proceedings of the Twelfth International Conference on Learning Representations (2023).

[24] H. Wang, B. Liu, H. Shao, B. Xiao, K. Zeng, G. Wan, Y. Qian, Claq: Pushing the limits of low-bit post-training quantization for llms, arXiv preprint arXiv:2405.17233 (2024).

[25] X. Ma, G. Fang, X. Wang, Llm-pruner: On the structural pruning of large language models, Advances in neural information processing systems 36 (2023) 21702–21720.

[26] F. Lagunas, E. Charlaix, V. Sanh, A. M. Rush, Block pruning for faster transformers, The 2021 Conference on Empirical Methods in Natural Language Processing (2021).

[27] W. J. Yun, J. Park, J. Kim, Quantum multi-agent meta reinforcement learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 37, 2023, pp. 11087–11095.

[28] L. Li, J. Li, Y. Song, S. Qin, Q. Wen, F. Gao, An efficient quantum proactive incremental learning algorithm, SCIENCE CHINA Physics, Mechanics & Astronomy 68 (1) (2025) 1–9.

[29] Z. He, M. Deng, S. Zheng, L. Li, H. Situ, Training-free quantum architecture search, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 38, 2024, pp. 12430–12438.

[30] J. Li, F. Gao, S. Lin, M. Guo, Y. Li, H. Liu, S. Qin, Q. Wen, Quantum k-fold cross-validation for nearest neighbor classification algorithm, Physica A: Statistical Mechanics and its Applications 611 (2023) 128435.

[31] A. G. Barreto, F. F. Fanchini, J. P. Papa, V. H. C. de Albuquerque, Why consider quantum instead classical pattern recognition techniques?, Applied Soft Computing 165 (2024) 112096.

[32] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, S. Woerner, The power of quantum neural networks, Nature Computational Science 1 (6) (2021) 403–409.

[33] K. Beer, D. Bondarenko, T. Farrelly, T. J. Osborne, R. Salzmann, D. Scheiermann, R. Wolf, Training deep quantum neural networks, Nature communications 11 (1) (2020) 808.

[34] Y. Song, Y. Wu, S. Wu, D. Li, Q. Wen, S. Qin, F. Gao, A quantum federated learning framework for classical clients, Science China Physics, Mechanics & Astronomy 67 (5) (2024) 250311.

[35] F. Fan, Y. Shi, T. Guggemos, X. X. Zhu, Hybrid quantum-classical convolutional neural network model for image classification, IEEE transactions on neural networks and learning systems (2023).

[36] G. Team, R. Anil, S. Borgeaud, Y. Wu, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, et al., Gemini: a family of highly capable multimodal models, arXiv preprint arXiv:2312.11805 (2023).

[37] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, Y. Li, X. Wang, M. Dehghani, S. Brahma, et al., Scaling instruction-finetuned language models, Journal of Machine Learning Research 25 (70) (2024) 1–53.

[38] J. Wei, M. Bosma, V. Y. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, Q. V. Le, Finetuned language models are zero-shot learners, Proceedings of the International Conference on Learning Representations (2021).

[39] G. Swamy, C. Dann, R. Kidambi, Z. S. Wu, A. Agarwal, A minimaximalist approach to reinforcement learning from human feedback, arXiv preprint arXiv:2401.04056 (2024).

[40] T. Kaufmann, P. Weng, V. Bengs, E. Hüllermeier, A survey of reinforcement learning from human feedback, arXiv preprint arXiv:2312.14925 (2023).

[41] W. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez, et al., Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, See https://vicuna. lmsys. org (accessed 14 April 2023) 2 (3) (2023) 6.

[42] A. Vaswani, Attention is all you need, Advances in Neural Information Processing Systems (2017).

[43] J. Bausch, Recurrent quantum neural networks, Advances in neural information processing systems 33 (2020) 1368–1379.

[44] Y. Du, Y. Yang, D. Tao, M.-H. Hsieh, Problem-dependent power of quantum neural networks on multiclass classification, Physical Review Letters 131 (14) (2023) 140601.

[45] C.-H. H. Yang, J. Qi, S. Y.-C. Chen, Y. Tsao, P.-Y. Chen, When bert meets quantum temporal convolution learning for text classification in heterogeneous computing, in: ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2022, pp. 8602–8606.

[46] M. J. Hasan, M. Mahdy, Bridging classical and quantum machine learning: Knowledge transfer from classical to quantum neural networks using knowledge distillation, arXiv preprint arXiv:2311.13810 (2023).

[47] L. Mingze, F. Lei, C. Aaron, Z. Xinyue, P. Miao, H. Zhu, Hybrid quantum classical machine learning with knowledge distillation, in: ICC 2024-IEEE International Conference on Communications, 2024, pp. 1139–1144.

[48] M. Alam, S. Kundu, S. Ghosh, Knowledge distillation in quantum neural network using approximate synthesis, in: Proceedings of the 28th Asia and South Pacific Design Automation Conference, 2023, pp. 639–644.

[49] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, Lora: Low-rank adaptation of large language models, Proceedings of the International Conference on Learning Representations (2021).

[50] J. I. Díaz-Ortiz, A. Villanueva, F. Delgado, Strongly entangling neural network: Quantum-classical hybrid model for quantum natural language processing, in: International Conference on Mathematical Modeling in Physical Sciences, Springer, 2023, pp. 503–514.

[51] J. Devlin, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805 (2018).

[52] H. Lee, Y. Park, H. Seo, M. Kang, Self-knowledge distillation via dropout, Computer Vision and Image Understanding 233 (2023) 103720.

[53] X. Wu, Z. Zhu, G. Chen, W. Pedrycz, L. Liu, M. Aggarwal, Generalized todim method based on symmetric intuitionistic fuzzy jensen–shannon divergence, Expert Systems with Applications 237 (2024) 121554.

[54] A. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, C. Potts, Learning word vectors for sentiment analysis, in: Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies, 2011, pp. 142–150.

[55] Y. Wang, R. Song, R. Zhang, J. Liu, Up4ls: User profile constructed by multiple attributes for enhancing linguistic steganalysis, arXiv preprint arXiv:2311.01775 (2023).

[56] A. H. Khan, H. Kegalle, R. D'Silva, N. Watt, D. Whelan-Shamy, L. Ghahremanlou, L. Magee, Automating thematic analysis: How llms analyse controversial topics, Microsoft Journal for Applied Research (2024).

[57] C.-Y. Chen, T.-M. Hung, Y.-L. Hsu, L.-W. Ku, Label-aware hyperbolic embeddings for fine-grained emotion classification, Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (2023).

[58] Y. Wang, R. Song, R. Zhang, J. Liu, L. Li, Llsm: Generative linguistic steganography with large language model, arXiv preprint arXiv:2401.15656 (2024).

[59] Y. Wang, R. Zhang, J. Liu, V-a3ts: A rapid text steganalysis method based on position information and variable parameter multi-head self-attention controlled by length, Journal of Information Security and Applications 75 (2023) 103512.

[60] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, Q. Liu, Tinybert: Distilling bert for natural language understanding, Findings of the Association for Computational Linguistics: EMNLP 2020 (2020).

[61] S. Sun, Y. Cheng, Z. Gan, J. Liu, Patient knowledge distillation for bert model compression, Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP).

[62] S. Lai, L. Xu, K. Liu, J. Zhao, Recurrent convolutional neural networks for text classification, in: Proceedings of the AAAI conference on artificial intelligence, Vol. 29, 2015.

[63] H. Zhu, C. Zhang, J. Huang, J. Wu, K. Xu, Hitin: Hierarchy-aware tree isomorphism network for hierarchical text classification, Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (2023).

[64] H. Wang, Z. Yang, J. Yang, C. Chen, Y. Huang, Linguistic steganalysis in few-shot scenario, IEEE Transactions on Information Forensics and Security (2023).

[65] D. P. Kingma, Adam: A method for stochastic optimization, Proceedings of the 3rd International Conference for Learning Representations (2014).