Article

Not peer-reviewed version

Accelerating and Compressing Transformer-based PLMs for Enhanced Comprehension of Computer Terminology

Jian Peng and Kai Zhong *

Posted Date: 30 September 2024

doi: 10.20944/preprints202409.2415.v1

Keywords: Pre-training models; parallelism; mixed compression method; computer-specialized terms

Article

Accelerating and Compressing Transformer-Based PLMs for Enhanced Comprehension of Computer Terminology

Jian Peng [1] and Kai Zhong [2,*]

[1] Changsha Normal University, Changsha 410000, Hunan, China
[2] College of Computer Science and Electronic Engineering, Hunan University, Changsha 410000, Hunan, China
[*] Correspondence: startkz@hnu.edu.com

Abstract: Pre-trained language models (PLMs) have significantly advanced natural language processing (NLP), establishing the "pre-training + fine-tuning" paradigm as a cornerstone approach in the field. However, the vast size and computational demands of Transformer-based PLMs present challenges, particularly regarding storage efficiency and processing speed. This paper addresses these limitations by proposing a novel lightweight PLM optimized for accurately understanding domain-specific computer terminology. Our method involves a pipeline parallelism algorithm designed to accelerate training. It is paired with an innovative mixed compression strategy that combines pruning and knowledge distillation to effectively reduce the model size while preserving its performance. The model is further fine-tuned using a dataset that mixes source and target languages to enhance its versatility. Comprehensive experimental evaluations demonstrate that the proposed approach successfully achieves a balance between model efficiency and performance, offering a scalable solution for NLP tasks involving specialized terminology.

Keywords: pre-training models; parallelism; mixed compression method; computer-specialized terms

---

1. Introduction

Artificial Intelligence (AI) stands as an emerging field of technological science, encompassing theories, methodologies, technologies, and application systems aimed at replicating and extending human intelligence. Its evolution marks a pivotal shift in modern society, where AI technologies are progressively transforming various aspects of our lives. Pre-trained Language Models (PLMs), characterized by their vast parameter counts, represent intricate artificial neural network models. The emergence and adoption of such technology signify a significant milestone, ushering artificial intelligence research into the era of generalized artificial intelligence. In the rapid evolution of large models, the amalgamation of extensive datasets, substantial computing power, and sophisticated algorithms has substantially enhanced the pre-training, generation capabilities, and applicability across multi-modal and multi-scenario domains [1,2]. For instance, ChatGPT's remarkable success owes much to the robust computational resources provided by platforms like Microsoft Azure, coupled with extensive datasets such as Wikipedia. The strategy of fine-tuning GPT models and incorporating reinforcement learning with human feedback (RLHF) has further propelled its advancements, all built upon the robust foundation of the Transformer architecture. Indeed, the Transformer architecture serves as the backbone for many large models, capable of significant model variations through parameter adjustments. This sets the stage for comprehensive exploration and experimentation to further harness the potential of large models and their underlying architectures. Let's delve into a detailed examination of experimental procedures and findings.

PLMs are characterized by their scale, emergence, and generality. These models, such as ChatGPT, are significant in the quest for generalized AI due to their vast number of parameters and deep network structures. They possess the capability to learn and comprehend intricate features and patterns, thereby demonstrating remarkable abilities in natural language understanding, intent recognition, inference, context modeling, language generation, and various other natural language processing tasks. Additionally, they exhibit a general-purpose problem-solving ability, making them valuable assets in the pursuit of general artificial intelligence. The current development trajectory of large models is illustrated in Figure 1.

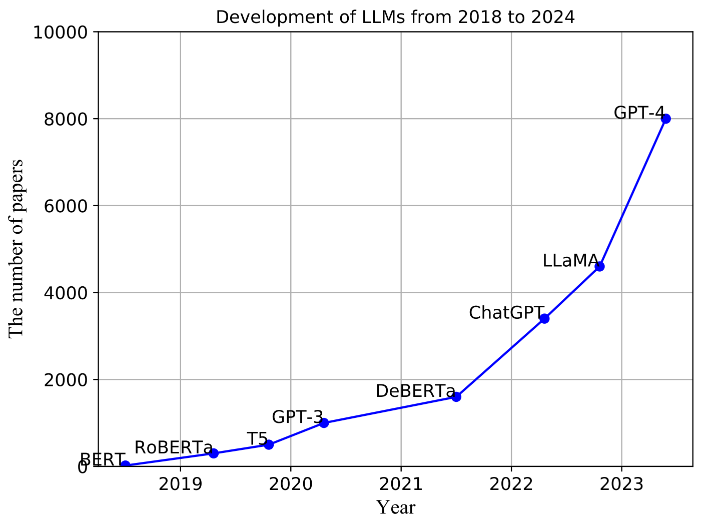

Figure 1. LLMs development process.

The training paradigm of PLMs, characterized by "pre-training + fine-tuning," has revolutionized natural language processing. Initially, a language model based on the Transformer architecture undergoes pre-training on a large-scale corpus, followed by fine-tuning on a downstream task. This approach ensures exceptional model performance, even with limited training data for the downstream task. The success of PLMs in English has prompted their widespread adoption in other languages, leading to the development of language-specific versions such as ChineseBERT [3], CamemBERT [4], and RobBERT [5]. However, pre-training language-specific models typically demand abundant training data [6]. For low-resource or resource-poor languages, leveraging the cross-lingual capabilities of multilingual PLMs to transfer relevant knowledge from source languages for target language tasks has emerged as an active research direction. Nonetheless, similar to monolingual PLMs, the immense parameter count in multilingual PLMs poses challenges for deployment on resource-constrained devices. Thus, reducing storage and computational demands for reasoning in multilingual PLMs through model compression techniques has become a pressing issue in the industry. Moreover, as the performance of pre-trained language models continues to advance, their parameter counts have surged exponentially. For instance, the GPT-3 model by OpenAI boasts a staggering 175 billion parameters [7]. This exponential growth in parameters brings forth computational, storage, and power requirements that pose significant challenges for real-world application scenarios. Addressing these challenges is crucial to harnessing the full potential of PLMs in practical settings.

Large models possess robust generalization capabilities, yet often lack domain-specific expertise. Consider Computer English as a case in point. In contrast to everyday English, Computer English exhibits distinctive features, characterized by fragmented vocabulary specific to the computing domain and widespread usage of abbreviated proprietary terms [8,9]. Research indicates that translating Computer English requires not only addressing vocabulary nuances but also employing specialized translation techniques to achieve desired outcomes. In today's context, computers have permeated various facets of daily life and work, leading individuals to encounter computer-related professional English gradually. Examples include error feedback, prompts, and assistance in computer operations. Additionally, a significant portion of technical details in computer-related data is articulated in English. Consequently, difficulty in accurately interpreting such content may hinder internet access and impede work efficiency. To effectively handle domain-specific tasks like Computer English translation, PLMs must undergo training tailored to specific domains. Customizing models for various scenarios within the domain facilitates the development of comprehensive models tailored to specific fields. This approach ensures that PLMs possess the requisite expertise to address domain-specific challenges effectively.

Extensive research has been conducted on PLMs, exploring various algorithms such as quantization, pruning, distillation, dynamic networks, data parallelism, model parallelism, and fine-tuning [10–14]. However, two prominent challenges persist regarding acceleration, compression, and fine-tuning techniques for large language models:

- Many compression algorithms necessitate fine-tuning or even retraining the model post-compression. Notably, the significant challenge associated with large models lies in the considerable cost incurred by model fine-tuning or training. Consequently, several algorithms for large models are delving into approaches that circumvent the need for tuning, such as quantization and pruning.
- Large models prioritize generality and generalization capabilities over performance in singular tasks. However, targeting specific domains requires a concentrated effort to align with downstream tasks more effectively. Hence, there's a growing emphasis on developing strategies that enhance alignment with specific domains while leveraging the inherent generality of large models.

Our Approaches. To address these challenges, we propose LightChatGLM, based on the ChatGLM-6B architecture, specifically tailored for training PLMs optimized for computerized English tasks. LightChatGLM integrates structured pruning and knowledge distillation with advanced techniques such as pipeline parallelism to create a lightweight model. This approach optimizes efficiency while maintaining excellent cross-linguistic capabilities, making it particularly effective for tasks involving computerized English.

Contributions. The main contributions of this paper are summarized as follows.

- Introduction of a pipeline parallelism-based training method that optimally utilizes computational resources, thereby enhancing training efficiency for Transformer-based LM basic models.
- Proposal of a mixed compression method for pre-trained language models, which amalgamates multiple compression methodologies, each with distinct principles of operation, to capitalize on their respective strengths. Initially, the teacher model is pruned structurally to obtain the student model. Subsequently, the student model undergoes knowledge distillation to recover lost information due to structured pruning, thereby refining its performance.
- Development of a fine-tuning methodology utilizing labeled target language datasets. This fine-tuning method utilizes a small subset of labeled target language datasets, which are randomly mixed with source language labeled data. Subsequently, the model is fine-tuned on the mixed dataset, followed by continuous knowledge distillation on the downstream task. This process facilitates the transfer of cross-linguistic competence from the teacher model to the student model, enhancing the latter's performance in the target language. Finally, experimental comparisons with state-of-the-art methods verified the validity of the proposed methods

The rest of the paper is organized as follows. Section 2 reviews the related work. The process of LightChatGLM is presented Section 3. Section 4 evaluates the performance of LightChatGLM. Section 5 concludes the paper.

2. Related Work

2.1. Training and Inference for PLMs

As transformer-based models continue to evolve, numerous variants have been developed to address various application needs, each introducing additional demands related to latency, throughput, and memory. These requirements pose challenges in deploying models efficiently, making the development of an PLMs inference acceleration framework crucial to address the efficiency needs across various scenarios.

To maximize training throughput, numerous strategies have been employed, including tensor parallelism, pipeline parallelism, expert parallelism, distributed heterogeneity, etc [15–18]. Among various parallel strategies, intermediate result fusion is particularly effective in minimizing redundant intermediate outputs, lowering memory usage, and reducing unnecessary memory I/O and kernel startup overhead. This optimization improves the efficiency of computational resources such as GPUs, CPUs, and registers [19–21]. For instance, Microsoft's DeepSpeed Inference [22] offers an efficient integrated inference system, achieving notable reductions in latency and improvements in throughput. Similarly, FlexGen et.al [23] presents an offloading-based inference system. Additionally, the open-source inference framework Power-Infer [24] introduces an innovative GPU-CPU heterogeneous hybrid inference engine, leveraging the highly localized sparse activation property of PLMs to minimize memory requirements and data transfer overhead between CPU and GPU.

However, many existing acceleration strategies process only a small amount of data at a time, leading to suboptimal performance for small batches and low memory bandwidth utilization, resulting in significant overhead. In this paper, we propose leveraging a pipeline parallelism-based algorithm during the training phase of PLMs to enhance GPU space utilization and improve training efficiency by dividing blocks.

2.2. Compression Techniques

PLMs are known for their emphasis on generality, generalization capabilities, and even emergence across a wide array of tasks and unseen data. Consequently, compressed PLMs must undergo careful verification to ensure their retention of generality and generalization capabilities. In response to these challenges, various compression methods have been proposed specifically tailored for PLMs. Common compression techniques include quantization, pruning, knowledge distillation, and dynamic networks.

Quantization refers to the process of converting a model's parameters and/or computations from high-precision representation to lower-precision formats. This technique is commonly used to reduce the computational resources required for running models and to improve their efficiency, especially in deployment scenarios where hardware constraints are a concern [25–30]. These include methods that do not necessitate retraining, such as post-training quantization (PTQ), and methods that do require retraining, like quantization-aware training (QAT). PTQ methods circumvent the need for an expensive retraining process, making them a more viable direction for most researchers.

Pruning is a technique used to reduce the size and complexity of a neural network by removing certain elements, such as weights, neurons, or entire layers. This helps in making the model more efficient without significantly sacrificing performance. However, its effectiveness can be undermined during the fine-tuning phase, which is often costly, particularly for models with a large number of parameters. Despite these challenges, pruning remains a critical technique for model compression, deserving further investigation to improve its application in PLMs. Notable unstructured pruning methods, such as SparseGPT [31] and Wanda [32], have set benchmarks for subsequent approaches. Wanda, for instance, introduces a unique pruning metric that accounts for both weight magnitude and activation values, while another method, Relative Importance and Activation (RIA), addresses channel corruption by clipping entire rows and columns of the weight matrix. To further enhance pruning effectiveness in PLMs, various auxiliary techniques have been developed, including region-specific sparsity rates [33,34], post-pruning fine-tuning strategies [35–37], and hardware optimizations [38,39]. Despite advancements, significant challenges remain, particularly in integrating pruning with other techniques and addressing the high cost of fine-tuning.

Knowledge distillation (KD) is a technique in machine learning where a smaller, more efficient model (often referred to as the "student") is trained to replicate the performance of a larger, more complex model (the "teacher") [40]. The goal is to transfer the knowledge from the teacher model to the student model, enabling the student to achieve similar performance while being more computationally efficient and faster to deploy [41]. To improve the effectiveness of distillation, various enhancement strategies have been proposed, such as multi-task learning and tailored dataset distillation. For instance, DISCO leveraged a pre-trained language model (PLM) to generate counterfactual data, which was subsequently filtered using a large-scale teacher model for natural language inference tasks [42]. Similarly, PubMedBERT is tailored to handle the nuances and specialized terminology of biomedical text, which makes it particularly useful for tasks such as biomedical information extraction, literature mining, and question answering within the medical domain [43]. PromptMix is a technique designed to enhance the performance of language models by combining multiple prompts in a structured way. It leverages the strengths of different prompts to provide more comprehensive and contextually relevant responses [44].

Combining knowledge distillation with other techniques to improved performance [45–47], motivating researchers to explore the synthesis of multiple compression methods for real-world application scenarios. In this paper, we propose a mixed compression method that combines pruning and knowledge distillation to achieve better performance in obtaining a student model.

Moreover, many compression algorithms often necessitate post-compression fine-tuning or even retraining to regain accuracy [48–52]. However, performing full parametric fine-tuning or training for medium or large models can be prohibitively expensive. Various parameter-efficient fine-tuning (PEFT) algorithms are effective methods to solve the above problems. These algorithms aim to fine-tune as few parameters or cycles as possible to reduce the cost of fine-tuning, such as LORA [53], QLoRA [54], Adapter Tuning [55], Prefix Tuning [56], and Prompt Tuning [57], among others.

In the context of training PLMs for computerized English, this paper introduces a hybrid data fine-tuning method. This method leverages annotated target language data along with source language data, randomly mixing them for fine-tuning purposes. Furthermore, continuous knowledge distillation is performed to transfer cross-linguistic competencies from the teacher model to the student model, thereby enhancing the student model's performance in the target language.

3. Methodology

3.1. Pipeline Parallelism

Referring to the DAPPLE algorithm [58], it accelerates the training of large models by pipelining data in parallel. It divides the training task into multiple phases and utilizes parallel computing resources to execute these phases simultaneously, thereby enhancing training efficiency. Additionally, it adopts a pipelined data transfer method, which enhances the efficiency of data transfer between different stages. LightChatGLM builds upon the DAPPLE algorithm to optimize the training of transformer-based models.

Suppose there is a training dataset consisting of N samples, each denoted as (X_i, Y_i), where X_i is the input sequence and Y_i is the target sequence. The initial transformer-based model, denoted as M_{init}, can be represented as follows:

$$M_{init} = \{\theta_{init}\}, \tag{1}$$

where, θ_{init} denotes the initial parameters of the model. In the pipeline construction phase, we construct a pipeline consisting of multiple parallel stages. Each stage executes sequentially in the pipeline and passes the data in a pipelined manner. Suppose there is a pipeline containing K parallel stages, denoted as

$$S = \{S_1, S_2, ..., S_K\}, \tag{2}$$

where, S_i denotes the i-th stage, and each stage contains computational tasks such as forward propagation, back propagation and parameter update. For each stage S_i, if we assume that there are P_{max} parallel processing units that can perform computational tasks simultaneously, then it can be expressed as P_{max} processing units working simultaneously, i.e. $parallel(S_i) = P_{max}$. This means that the tasks in stage S_i are assigned to P_{max} processing units to maximize parallelism and processing efficiency.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 30 September 2024 doi:10.20944/preprints202409.2415.v1

In the data transfer and update phase, we transfer the intermediate results computed in each phase to the next phase and leverage parallel computing resources to update the model parameters simultaneously in the parameter update phase. Denoting the communication time in the data transfer phase as T_{comm}, the intermediate results computed in each phase S_i will be transferred to the next phase S_{i+1}. T_{comm} is represented by

$$T_{comm} = \sum_{i=1}^{K-1} transfer(S_i, S_{i+1}), \tag{3}$$

where, $transfer(S_i, S_{i+1})$ indicates the data transfer time from stage S_i pass to stage S_{i+1}. T_{update} denotes the time of the parameter update phase, which uses parallel computing resources to update the parameters of the model simultaneously as follows:

$$T_{update} = parallel(S_K). \tag{4}$$

P_{opt} to denote the number of optimized parallel computing resources. By dynamically adjusting the degree of parallelism as follows, the algorithm can be made to achieve optimal performance in different hardware environments.

$$P_{opt} = optimize(P_{max}), \tag{5}$$

where, $optimize(P_{max})$ denotes a function that dynamically adjusts the degree of parallelism according to the hardware environment and task requirements. Based on the above description, a DAPPLE-based improved algorithm can be obtained for training the transformer-based model, as shown in Algorithm 1.

Algorithm 1: Pipeline Parallel Transformer-based Model Training Algorithm

```
1: **Initialize** base model $M$, training data $T_d$, iterations $num\_epochs$, sample batches $iterate\_batches$
2: **Initialize** the number of pipeline stages $num\_stages$
3: **Initialize** the parallelism $parallelism = determine\_parallelism()$
4: Construct the pipeline $pipeline = construct\_pipeline(num\_stages)$
5: **for** $epoch$ in range($num\_epochs$) **do**
6:     **for** $batch$ in range($iterate\_batches$) **do**
7:         **for** $stage$ in range($num\_stages$) **do**
8:             forward_pass(pipeline[stage], batch)
9:             synchronize()
10:        **end for**
11:        **for** $stage$ in reversed(range($num\_stages$) **do**
12:            backward_pass_and_update(pipeline[stage], batch)
13:            synchronize()
14:        **end for**
15:    **end for**
16:    Output algorithm performance evaluation results
17: **end for**
18: **return** Trained model
```

In Algorithm 1, the initialization phase includes setting up the model, training dataset, number of pipeline stages, and parallelism, as depicted in lines 1-3 of Algorithm 1. Then, during the model training phase, samples are initially divided based on sample blocks, followed by parameter calculation and update via forward and backward propagation, as indicated in lines 4-14 of the algorithm. Eventually, the trained transformer model is obtained. The time complexity of the algorithm primarily revolves around three key factors: data transmission time T_{comm}, task training time T_c, and parameter updating time T_{update}. Considering these factors, the overall time complexity of the algorithm can be expressed as $max(K \cdot T_{comm}, T_c, T_{update})$.

3.2. Mixed Compression

The computations in the transformer structure are first reorganized by attention head and the proposed structured pruning method is given based on this, and then the knowledge distillation process for PLM is introduced. A typical transformer layer comprises a multi-head attention (MHA) sublayer and a feed-forward network (FFN) sublayer. By segregating the computation of MHA and FFN into independent parallel tasks, computational resources can be utilized more efficiently. This restructuring of computations within the transformer layer facilitates accelerated computation, thereby enhancing the efficiency of both model training and inference processes.

In the standard transformer layer, assuming that each transformer layer has N_a attention heads, the MHA obtains the parameters associated with the attention heads as query Q, key K, and value V matrices by transforming the input sequence X through a linear transformation as follows:

$$Q_i = XW_i^Q, K = XW_i^K, V = XW_i^V, \tag{6}$$

where, W_i^Q, W_i^K, W_i^V are the weight matrices of the linear transformation. Then the attention score is calculated, the Softmax function is applied to get the attention weights, and finally the output for each position is obtained by weighted summation as follows:

$$A_i(Q, K, V) = Softmax(\frac{Q_i K_i^T}{\sqrt{d_k}}), \tag{7}$$

where, d_k represents the dimension of the query or key. Then, the output of each attention head is concatenated and linearly transformed to obtain the final attention head output, i.e., the attention matrix A is weighted and summed against the value matrix V by

$$Output_i = A_i V_i. \tag{8}$$

Finally, the output matrix $Output$ of each attention header is spliced and linearly transformed to obtain the final attention header output by

$$Output = Concat(Ouput_1, Ouput_2, ..., Ouput_{N_a})W^O, \tag{9}$$

where, W^O is the weight matrix of the output linear transformation and $Concat$ denotes the splicing operation. The concept of reorganization involves treating the computation of each attention head as an individual task to be executed in parallel. Consequently, the computation of each head in the multi-head attention mechanism can be decomposed into independent computing units that operate without interference, thereby achieving parallel computation.

The FFN performs a linear transformation of the inputs at each position and then generates the final output by means of an activation function (e.g. ReLU) and another linear transformation. This process is applied independently to each position in the input sequence, meaning that each vector at each position undergoes the same FFN transformation. This independence allows the model to capture position-specific features, which are crucial for tasks like language modeling and machine translation.

As depicted in Figure 2, leveraging the aforementioned formulas, if we maintain the number of attention heads N_a unchanged while altering the dimension of the attention heads from d_a to $d_a^{'}$, the model dimension will consequently become $d^{'} = d_a^{'} \times N_a$. Similarly, if the dimension of the FFN intermediate layer d_{ffn} is adjusted to $d_{ffn}^{'} = \frac{d_a^{'}}{d_a} \times d_{ffn}$, LightChatGLM can preserve the original transformer structure parameter scale. Additionally, a subset of the transformer layer is selected from the ChatGLM model to undergo structural pruning in the depth direction. By employing pruning in both the depth and dimension directions, LightChatGLM can obtain a student model with reduced layers and smaller dimensions, initialized by the teacher model parameters.

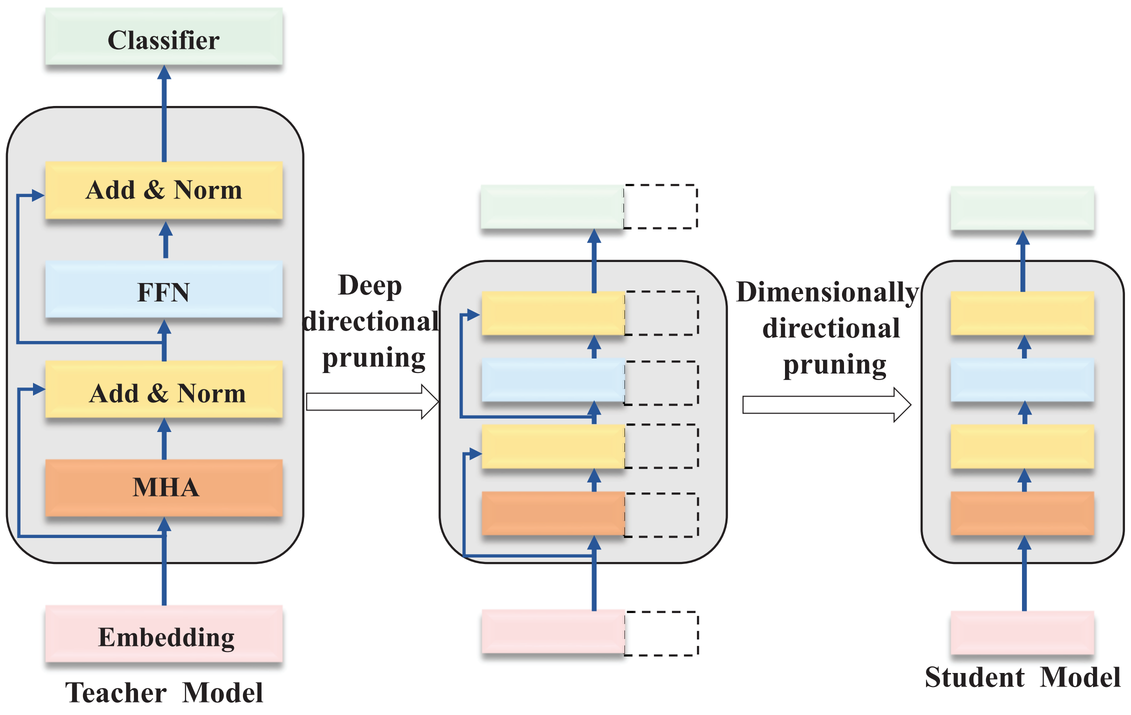

Figure 2. Structured Pruning Strategy.

After deriving the student model through structured pruning, LightChatGLM proceeds with knowledge distillation to facilitate the migration of bilingual knowledge from the ChatGLM model to the student model. This step aims to help the student model recover the information lost due to structured pruning. Drawing from the knowledge distillation approach employed in the task-independent phase of the TinyBERT framework [59], LightChatGLM adopts a distinct layer mapping strategy. Reference to existing literature reveals [60,61] that the last layer strategy proves more beneficial for the student model compared to the average strategy utilized in the original TinyBERT framework during the task-independent knowledge distillation phase of the pre-training process. Consequently, LightChatGLM opts to employ the last layer for distillation instead of distilling all hidden layer knowledge from the teacher model.

Assuming that the student model and the teacher model consist of M_s and N_t layers of transformer, respectively, and the last hidden state of the teacher model is denoted as $H_t^{N_t}$, We utilize $H_t^{N_t}$ as a soft label to guide the training of the student model. For each training sample, the input data are forwarded through the first M_s layers of the student model to obtain the last hidden state $H_s^{M_s}$ of the student model. Then, the squared loss function is employed to measure the loss of the distillation process as follows:

$$L_{KD} = \frac{1}{2}\sum_{i,j}(H_t^{N_t}(i,j) - H_s^{M_s}(i,j))^2, \tag{10}$$

where, $H_t^{N_t}(i,j)$ and $H_s^{M_s}(i,j)$ denote the j-th feature of the i-th sample of the hidden state in the last layer of the teacher model and the student model. The parameters of the student model are optimized by minimizing the loss function L_{KD} so that the student model can closely approximate the output of the teacher model. Through the last layer knowledge distillation strategy, LightChatGLM successfully transfers bilingual knowledge from the teacher model to the student model, enabling the student model to recover the information lost due to structured pruning. This strategy is particularly effective in task-independent knowledge distillation scenarios during the pre-training phase, where the final layer strategy proves more beneficial to the student model compared to distilling all hidden layers of the teacher model. The whole process is shown in Algorithm 2. The trained teacher model undergoes structural pruning in both the depth and gradient directions, as illustrated in lines 1-6 of Algorithm 2. Subsequently, knowledge distillation is carried out to further transfer bilingual knowledge from the teacher model to the student model, enabling the student model to recover information lost during structural pruning, as outlined in lines 7-14 of Algorithm 2. Ultimately, the compressed student model is obtained.

Algorithm 2: PLMs Mixed Compression Algorithm

```
1:  Initialize teacher_model by Algorithm 1, source_dataset $D_s$, target_dataset $D_d$, $num\_epochs$
2:  for $epoch$ in range($num\_epochs$) do
3:    Get the teacher_model
4:    Calculate MHA output by Eq.9
5:    Calculate FFN
6:    Update teacher_model after pruning
7:    for $batch$ in range($D_s$ and $D_d$) do
8:      Generate soft_labels by teacher_model($D_s(batch)$)
9:      Smooth logits
10:     Calculate loss and gradients by Eq.10
11:     Generate hard_labels by teacher_model($D_d(batch)$)
12:     Smooth logits
13:     Calculate loss and gradients by Eq.10
14:     Update student_model
15:   end for
16: end for
17: return student_model
```

3.3. Hybrid Data Fine-Tuning

By leveraging structured pruning to initialize the student model and subsequently transferring cross-lingual knowledge from the teacher model through knowledge distillation, LightChatGLM achieves a streamlined model with cross-lingual capabilities. To enhance the student model's understanding of computer English, LightChatGLM continues with cross-language knowledge distillation on downstream tasks to further refine the model's performance in computer languages. Specifically, LightChatGLM aims to refine the teacher model through fine-tuning to enhance its cross-linguistic competence. Then, this competence is transferred to the student model through cross-language knowledge distillation on downstream tasks. This section introduces a hybrid data-based fine-tuning method wherein annotated data from both the source and target languages are randomly mixed for fine-tuning the teacher model. This approach enables the teacher model to autonomously learn the connections and similarities between languages during training, thereby improving its performance in the target language with the aid of rich source language knowledge.

Prepare labeled datasets for the source and target languages as D_s and D_d respectively. These datasets contain training samples for the target task along with their corresponding labels. Randomly mix the labeled data from D_s and D_d to create a hybrid dataset. In the mixed data fine-tuning process, a balancing parameter α is required to appropriately weigh the contributions of the source and target languages. The value of α can be determined based on the relative sizes of the source and target language datasets, as well as their significance to the target task. This relationship can be expressed by the following formula:

$$\alpha = \frac{w_s \cdot n_s}{w_s \cdot n_s + w_d \cdot n_d}, \tag{11}$$

where, w_s and w_d represent the weighting coefficients for the source and target languages, which are usually based on task-relevant performance metrics, such as translation accuracy or cross-language comprehension. The terms n_s and n_d denote the number of samples in the source and target language datasets.

Furthermore, the teacher model is adapted to the characteristics of the target task and its performance is improved by performing backpropagation and parameter updating on the hybrid data. The performance of the teacher model on the mixed data can be measured using the cross-entropy loss function. Specifically, for each sample $(D_s(i),D_d(j))$, the final objective function can be expressed as:

$$L_{hybrid} = -\frac{1}{N}\sum_{i=1,j=1}^{N}(\alpha \cdot log(f_{teacher}(D_s(i))) + (1-\alpha) \cdot log(f_{teacher}(D_d(j)))), \tag{12}$$

where, N is the total number of samples in the hybrid dataset, α is a balancing parameter between the source and target language samples, $f_{teacher}$ represents the prediction function of the teacher model.

During the fine-tuning process with random mixing, LightChatGLM continues to perform knowledge distillation to transfer the cross-linguistic competence of the teacher model to the student model, thereby enhancing the performance of the student model on the target language. The loss function for knowledge distillation on the downstream task is then formulated as follows:

$$L_{distill} = -\frac{1}{M}\sum_{j=1}^{M}(\lambda \cdot log(f_{teacher}(D_d(j))) + (1-\lambda) \cdot log(f_{student}(D_d(j)))), \tag{13}$$

where, M is the total number of samples in the target language dataset, $f_{student}$ represents the prediction function of the student model. The balance parameter λ is used to control the weighting between the teacher model predictions and the student model predictions, and typically, the value of λ can be adjusted between 0 and 1. Larger values of λ tend to rely more on the direct predictions of the student model for the target language, while smaller values of λ rely more on the guidance of the teacher model. The optimal λ value can be determined by cross-validation or performance on the validation set.

The detailed process is shown in Algorithm 3. In Algorithm 3, the first step involves randomly mixing the annotated data from both source and target languages. Subsequently, the teacher model is fine-tuned based on this mixed dataset, allowing it to adapt to the target task through back-propagation and parameter updating on the mixed data, as depicted in lines 1-8 of Algorithm 3. Throughout the fine-tuning process using randomly mixed data, the algorithm continually performs knowledge distillation to transfer the cross-linguistic competencies of the teacher model to the student model, thereby enhancing the performance of the student model on the target language, as shown in lines 9-14 of Algorithm 3. Ultimately, a streamlined student model with exceptional comprehension abilities in computer English can be achieved.

Algorithm 3: Hybrid Data Fine-Tuning Algorithm

```
1:  Initialize teacher_model and student_model by Algorithm 2, $mixed\_dataset$ by $D_s$ and $D_d$,
    $num\_epochs$
2:  for $epoch$ in range($num\_epochs$) do
3:      for $batch$ in range($mixed\_dataset$) do
4:          $D_s(i)$,$D_d(j) \leftarrow batch$
5:          Calculate $L_{hybrid}$ by Eq.12
6:          backward_propagation($L_{hybrid}$)
7:          Update teacher_model
8:      end for
9:      for $batch$ in range($mixed\_dataset$) do
10:         $D_d(j) \leftarrow batch$
11:         Get output teacher_output $\leftarrow$ teacher_model
12:         Calculate $L_{distill}$ by Eq.13
13:         backward_propagation($L_{distill}$)
14:         Update student_model
15:     end for
16: end for
17: return student_model
```

4. Experimental Evaluation

4.1. Experiment Setup

Hardware Environment: The server we used is equipped with an Intel(R) Xeon(R) Silver 4210R CPU at 2.40GHz and NVIDIA Tesla T4 GPUs with 16GB of RAM each, interconnected via PCIe-III. The server runs on a 64-bit Ubuntu 20.04 system with CUDA toolkit version 10.2 and PyTorch 1.10.2.

Hyper-Parameter Setting: LightChatGLM utilized Wikipedia pages containing computer-specific English and Chinese nouns as the training corpus. These corpora were randomly mixed together. The hyperparameters were set as follows: a batch size of 256, a maximum sequence length of 128, a dropout rate of 0.1, parameter decay of 0.05, and 400,000 steps for model parameter updates. The learning rate was initially set to 0.9 and decayed after the first 10% of the update steps. Based on these initial parameters, we perform a hyperparameter search to further optimize the model performance. Hyper-parameter search is usually performed using grid search or random search, combined with cross-validation, to select the optimal combination of parameters to enhance the model performance on a specific task [48,62,63].

For the downstream multilingual task, the datasets we used is shown in Table 1. The WMT 2020 Chinese-English comprises both Chinese and English segments from the Chinese-English translation tasks of the WMT competition, serving as a benchmark dataset for bilingual machine translation tasks. The UM-Corpus, developed by the University of Macau, is primarily used for high-quality research on Chinese-English parallel corpora, making it suitable for machine translation and semantic understanding tasks. The Ai Challenger encompasses Chinese-English parallel corpora for various tasks and is an essential resource for research in natural language processing and machine translation.

Table 1. DATASETS

Name	Chinese words/million	English words/million	Training set size	Development set size	Test set size
WMT 2020 Chinese-English [64]	130	110	90%	5%	5%
UM-Corpus [65]	50	48	80%	10%	10%
Ai challenger [66]	250	200	80%	10%	10%

During fine-tuning or distillation of the student model for this task, the hyperparameters were adjusted accordingly. Specifically, the maximum sequence length was set to 128, the batch size to 32, and the balance factors α and λ were both set to 0.65. Additionally, the smoothed logit temperature value was set to 1.

Comparison Models: We evaluate the performance of LightChatGLM by comparing it with the following typical schemes:

- **mBERT_drop** [67] : mBERT is a multilingual BERT model pre-trained for 104 languages, featuring the same structure as the original BERT model. Meanwhile, mBERT_drop represents a compression technique specifically designed for mBERT, involving the direct pruning of the top Transformer layer of the mBERT model.
- **DistilmBERT** [68]: The multilingual version of DistilBERT is a pre-training model that employs knowledge distillation techniques to decrease the size and enhance the speed of the BERT model. The concept behind its design is straightforward: construct a smaller model, referred to as DistilBERT, as the Student model, and utilize the original BERT model as the Teacher model. The goal is for the Student model to learn from the Teacher model as much as possible, thereby retaining the reasoning capabilities of the Teacher model to the fullest extent possible.
- **ChatGLM-6B** [69,70] : ChatGLM-6B is an open-source, bilingual conversation language model built on the GLM architecture. Leveraging model quantization technology, it demands as little as 6GB of video memory when operating at the INT4 quantization level.

4.2. Experiment Analysis

Figure 3 illustrates the variation in training time per epoch for the Transformer base model across different sample block sizes and a comparison of various parallelization strategies. As the sample block size increases, the training speed also increases, with the optimal speed achieved when the sample block size is set to 256. Additionally, the use of different parallelization strategies can significantly reduce the training time per epoch. LightChatGLM leverages pipelined parallelism to facilitate efficient distributed training across multiple GPUs, resulting in a minimum training time of 180 seconds per epoch. However, the higher communication overhead associated with data and model parallelism slightly extends the training duration. It is also important to consider that memory usage escalates substantially as the sample block size increases, necessitating careful design of the sample block size to achieve optimal training acceleration. This training speedup effect is observed on several training sets, including WMT 2020, UM-Corpus, and Ai challenger. experiments on these datasets show that the training speedup is indeed the result of the combined effect of pipeline parallelization and reasonable sample block division.

Figure 4 illustrates the validation loss of the mixed compression method compared to random pruning over an equal number of training rounds. We trained the base LightChatGLM model on the UM-Corpus dataset using the full dataset and parameter set to establish the initial validation loss. Subsequently, the model was subjected to mixed compression and random pruning to derive the corresponding student models, with the validation loss computed on the validation set. As shown in Figure 4, the mixed compression approach, which integrates structured pruning and knowledge distillation techniques, achieves higher compression efficiency while preserving model performance. The validation loss curves for the mixed compression method typically exhibit faster convergence and stabilize at lower loss values. In contrast, random pruning methods, which achieve compression by randomly removing certain weights from the model, may cause greater fluctuations in model performance. As a result, the validation loss curves for random pruning tend to converge more slowly and stabilize at relatively higher values. The experimental data demonstrate that the mixed compression method outperforms random pruning in both convergence speed and final validation loss, effectively maintaining model performance.

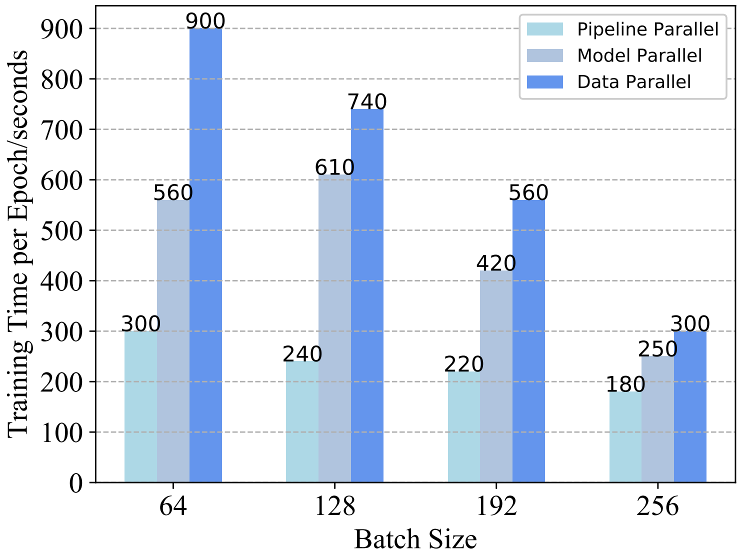

Figure 3. Effect of different parallel strategies on training time with different batch size.

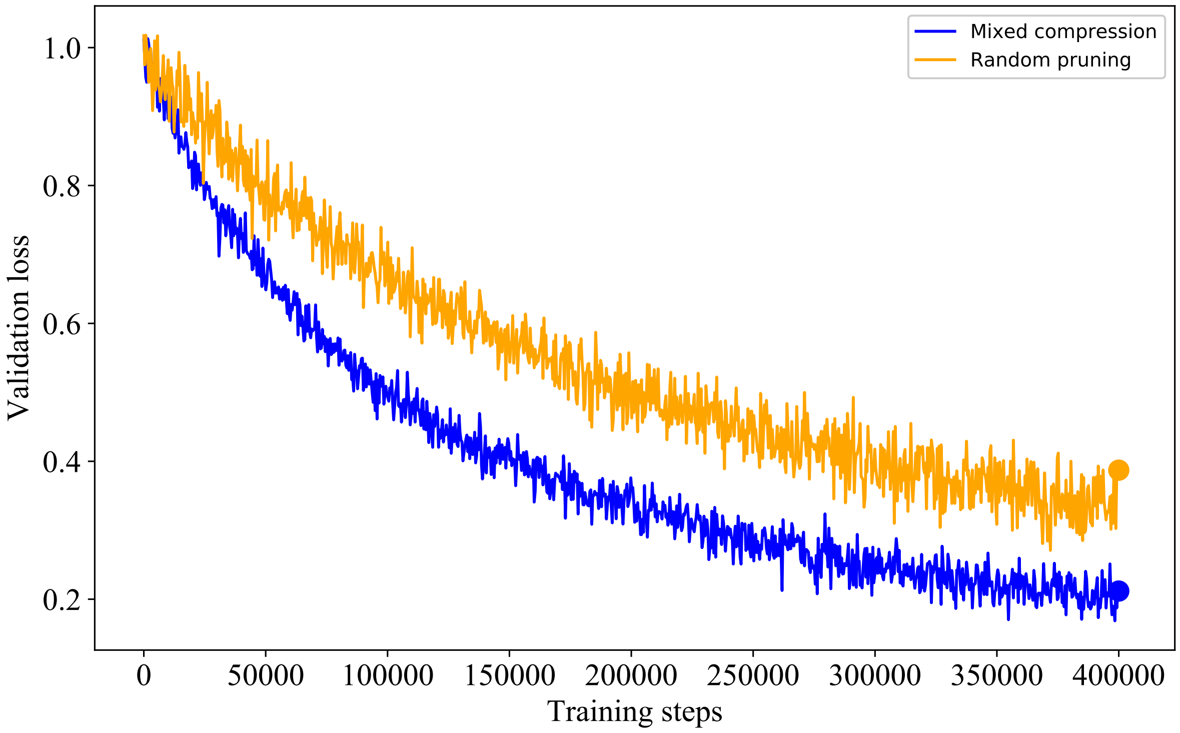

Figure 4. Validation loss over training steps for mixed compression and random pruning.

Table 2 offers a comprehensive comparison of the accuracy achieved in computerized English-Chinese translation without the application of target data for fine-tuning and knowledge distillation. A noteworthy observation from the table is that the absence of utilizing rich source-language annotated data for cross-language knowledge migration results in diminished generalization ability of the compressed model. Among the various methods evaluated, DistilmBERT's approach yields the most favorable outcomes with 46.3% in Chinese-English bilingual translation. However, it is important to note that despite this, the performance of the student model obtained through LightChatGLM after mixed compression demonstrates remarkable proximity to that of the teacher model within 9.4% gap. LightChatGLM effectively facilitates the migration of bilingual knowledge from the teacher model to the student model, which proves to be particularly effective in task-independent knowledge distillation scenarios during the pre-training stage. This successful knowledge transfer highlights the efficacy of LightChatGLM in preserving the essential knowledge and capabilities of the teacher model while achieving significant compression, thereby demonstrating its potential for practical deployment in real-world applications.

Table 2. COMPARISON OF EXPERIMENTAL RESULTS OF DIFFERENT COMPRESSION METHODS UNDER FINE-TUNING WITHOUT TARGET DATA (TRANSLATION ACCURACY)

Student/Teacher Model	English	Chinese	AVG
mBERT_drop/mBERT	58.4/60.4	30.4/37.6	44.4/49.0
DistilmBERT/mBERT	**59.2**/60.4	33.3/37.6	**46.3**/49.0
ChatGLM-6B/ChatGLM	57.6/60.2	**32.8**/39.5	45.2/49.9
LightChatGLM(Ours)/ChatGLM	57.2/60.2	30.1/39.5	43.7/49.9

Table 3 presents a comprehensive comparison of the accuracy achieved in computerized English to Chinese translations using a combination of fine-tuning and knowledge distillation with both target annotated language data and source language data. Upon examination of the table, it becomes evident that the accuracy of Chinese and English translations for all models significantly improves after employing hybrid data fine-tuning. This enhancement can be attributed to the utilization of labeled target data in conjunction with further knowledge distillation, which collectively contribute to the model's enhanced performance in the domain of Chinese and English bilingual translation. Furthermore, it is noteworthy that LightChatGLM exhibits slightly lower performance with 57.4% in English comprehension compared to the other models. This disparity arises from the relatively smaller English corpus available in the dataset utilized, as compared to the teacher model. However, by augmenting the training data with a more extensive Chinese corpus, LightChatGLM has the potential to achieve superior results with 33.3% in English to Chinese comprehension. This result underscores the importance of dataset composition and the need for mixed data to effectively train models for bilingual translation tasks. The dataset used in Tables 2 and 3 is Ai challenger.

Table 3. COMPARISON OF EXPERIMENTAL RESULTS OF DIFFERENT COMPRESSION METHODS UNDER FINE-TUNING WITH MIXED DATA (TRANSLATION ACCURACY)

Student/Teacher Model	English	Chinese	AVG
mBERT_drop/mBERT	58.7/61.2	32.4/41.1	45.6/51.2
DistilmBERT/mBERT	**59.8**/61.2	32.8/41.1	**46.3**/51.2
ChatGLM-6B/ChatGLM	58.2/60.5	33.1/40.5	45.7/50.5
LightChatGLM(Ours)/ChatGLM	57.4/60.5	**33.3**/40.5	45.4/50.5

In order to enhance the cross-linguistic capabilities of teacher models, LightChatGLM employs a hybrid data-based fine-tuning approach. This method involves randomly mixing annotated data from both the source and target languages, followed by fine-tuning the teacher model on the hybrid dataset while continuing knowledge distillation on downstream tasks. This process aims to yield student models with superior performance. Figure 5 illustrates the average results of Chinese-English bilingual migration experiments conducted using three different fine-tuning methods on three different dataset. Here, the English training set serves as the source-language labeled corpus, while the Chinese validation set acts as the target-language labeled corpus. From Figure 5, it is evident that the performance of fine-tuning solely using the source language yields the poorest results. This can be attributed to the absence of target annotation language, leading to diminished generalization ability of the fine-tuned model. Conversely, the fine-tuning method incorporating the target annotation language and hybrid annotation language demonstrates improved performance results.

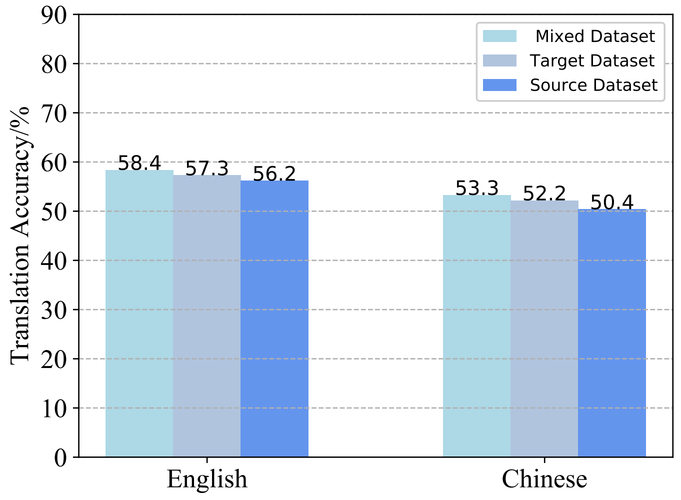

Figure 5. Experimental average results of cross-language migration with different fine-tuning methods on three different datasets.

We conducted multiple experiments across three distinct datasets to evaluate the performance of LightChatGLM in maintaining the integrity of the original ChatGLM-6B model under various compression rates. As shown in Figure 6, LightChatGLM effectively reduces storage costs and accelerates training, leading to more efficient resource utilization. The model retains 89.3% of the original performance after compression, while saving 35% in memory usage, and achieves 90.1% retention under 25% sparsity. These results indicate that the mixed compression approach—incorporating structured pruning and knowledge distillation, along with hybrid fine-tuning using both target and source labeled data—successfully recovers the performance of the compressed model. The proposed method demonstrates strong capabilities in bilingual Chinese-English translation.

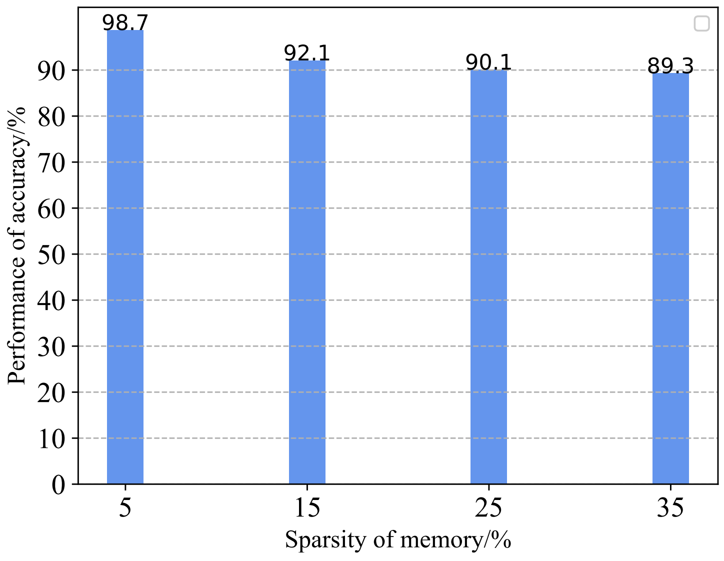

Figure 6. Performance of compression models with different sparsities of memory.

LightChatGLM employs a two-fold approach to accelerate the training of Transformer base models and enhance their cross-linguistic capabilities. Firstly, it utilizes a pipeline parallel approach to expedite the training process. Secondly, it employs a multilingual pre-trained model compression technique that combines pruning and knowledge distillation methods. This allows the student model to refine its cross-linguistic abilities effectively. The efficacy of the proposed method is validated through comparative experiments with other methods, focusing on teacher model acceleration and structured pruning of the Transformer base model. LightChatGLM demonstrates superior performance on a real Chinese-English bilingual dataset, achieving enhanced comprehension in both languages. Moreover, by leveraging both abundant source language data and a limited amount of target language labeled data, LightChatGLM implements mixed data fine-tuning across source and target languages. This approach enables the acquisition of a more robust teacher model. Furthermore, continuous distillation on downstream tasks facilitates the migration of cross-linguistic competencies from the teacher model to the student model, thereby enhancing the student model's performance in the target language.

5. Conclusions

The powerful cross-linguistic capabilities of multilingual pre-trained models enable them to handle tasks in resource-poor languages by transferring relevant knowledge from resource-rich languages. However, they also face challenges such as large model sizes and slow inference speeds, which hinder their deployment in real-world applications. To address these challenges and reduce training and storage overheads, this paper proposes LightChatGLM, an effective method based on pipeline parallelism and mixed compression for training models suitable for computer English translation applications. Firstly, LightChatGLM utilizes a pipeline parallelism-based approach to enhance the training of Transformer-based LM base models. This method optimizes the training process by leveraging pipeline parallelism, thereby improving efficiency. Secondly, a mixed compression method is proposed for pre-trained language models, which combines pruning and knowledge distillation techniques. This approach involves structurally pruning the teacher model and prompting the student model to recover lost information through knowledge distillation. By doing so, the model's cross-linguistic capabilities are further refined, ensuring better performance. Finally, to obtain a specialized model tailored for computerized English comprehension, the annotated target language dataset is used for fine-tuning. Furthermore, knowledge distillation on downstream tasks is continued to enhance the model's performance. Numerous experiments have been conducted to validate the effectiveness of LightChatGLM. Moving forward, we aim to address the time overhead associated with repeatedly running large models during compression. Additionally, we plan to explore collaborative hardware and software approaches to train pre-trained mini-models with superior performance.

Author Contributions: Conceptualization, Jian Peng and Kai Zhong; methodology, Jian Peng; software, Jian Peng; formal analysis, Jian Peng; writing—original draft preparation, Jian Peng; writing—review and editing, Kai Zhong; funding acquisition, Jian Peng. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Foreign Language Research Joint Project of the Social Science Foundation of Hunan Province (Grant No. 19WLH34).

Data Availability Statement: The data presented in this study are openly available in WMT 2020 https://www.statmt.org/wmt20, Um Corpus http://nlp2ct.cis.umac.mo/um-corpus/, and Ai challenger https://github.com/AIChallenger/AI_Challenger_2018?tab=readme-ov-file.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Chang, Y.; Wang, X.; Wang, J.; Wu, Y.; Yang, L.; Zhu, K.; Chen, H.; Yi, X.; Wang, C.; Wang, Y.; et al. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology* **2023**.
2. Hu, L.; Liu, Z.; Zhao, Z.; Hou, L.; Nie, L.; Li, J. A survey of knowledge enhanced pre-trained language models. *IEEE Transactions on Knowledge and Data Engineering* **2023**.

3. Sun, Z.; Li, X.; Sun, X.; Meng, Y.; Ao, X.; He, Q.; Wu, F.; Li, J. Chinesebert: Chinese pretraining enhanced by glyph and pinyin information. *arXiv preprint arXiv:2106.16038* **2021**.
4. Martin, L.; Muller, B.; Suárez, P.J.O.; Dupont, Y.; Romary, L.; de La Clergerie, É.V.; Seddah, D.; Sagot, B. CamemBERT: a tasty French language model. *arXiv preprint arXiv:1911.03894* **2019**.
5. Delobelle, P.; Winters, T.; Berendt, B. Robbert: a dutch roberta-based language model. *arXiv preprint arXiv:2001.06286* **2020**.
6. Wu, S.; Dredze, M. Are all languages created equal in multilingual BERT? *arXiv preprint arXiv:2005.09093* **2020**.
7. Floridi, L.; Chiriatti, M. GPT-3: Its nature, scope, limits, and consequences. *Minds and Machines* **2020**, *30*, 681–694.
8. Ping, Z.; Chunrong, W. Research on the Characteristics of English Text Based on Computer and Its Translation. *Journal of Physics: Conference Series* **2021**, *1992*.
9. Regina, D.; V, A.D. Computer-Based Vocabulary Learning in the English Language: A Systematic Review. *Theory and Practice in Language Studies* **2022**, *12*, 2365–2373.
10. Xu, C.; McAuley, J. A survey on model compression and acceleration for pretrained language models. In Proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, 2023, Vol. 37, pp. 10566–10575.
11. Wang, W.; Chen, W.; Luo, Y.; Long, Y.; Lin, Z.; Zhang, L.; Lin, B.; Cai, D.; He, X. Model Compression and Efficient Inference for Large Language Models: A Survey. *arXiv preprint arXiv:2402.09748* **2024**.
12. Choudhary, T.; Mishra, V.; Goswami, A.; Sarangapani, J. A comprehensive survey on model compression and acceleration. *Artificial Intelligence Review* **2020**, *53*, 5113–5155.
13. Zhu, X.; Li, J.; Liu, Y.; Ma, C.; Wang, W. A survey on model compression for large language models. *arXiv preprint arXiv:2308.07633* **2023**.
14. Zhao, H.; Chen, H.; Yang, F.; Liu, N.; Deng, H.; Cai, H.; Wang, S.; Yin, D.; Du, M. Explainability for large language models: A survey. *ACM Transactions on Intelligent Systems and Technology* **2024**, *15*, 1–38.
15. Zheng, L.; Li, Z.; Zhang, H.; Zhuang, Y.; Chen, Z.; Huang, Y.; Wang, Y.; Xu, Y.; Zhuo, D.; Xing, E.P.; et al. Alpa: Automating inter-and {Intra-Operator} parallelism for distributed deep learning. In Proceedings of the 16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22), 2022, pp. 559–578.
16. Shoeybi, M.; Patwary, M.; Puri, R.; LeGresley, P.; Casper, J.; Catanzaro, B. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053* **2019**.
17. Li, S.; Liu, H.; Bian, Z.; Fang, J.; Huang, H.; Liu, Y.; Wang, B.; You, Y. Colossal-ai: A unified deep learning system for large-scale parallel training. In Proceedings of the Proceedings of the 52nd International Conference on Parallel Processing, 2023, pp. 766–775.
18. Rasley, J.; Rajbhandari, S.; Ruwase, O.; He, Y. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In Proceedings of the Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 3505–3506.
19. Chen, T.; Moreau, T.; Jiang, Z.; Zheng, L.; Yan, E.; Shen, H.; Cowan, M.; Wang, L.; Hu, Y.; Ceze, L.; et al. {TVM}: An automated {End-to-End} optimizing compiler for deep learning. In Proceedings of the 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18), 2018, pp. 578–594.
20. Jiang, X.; Wang, H.; Chen, Y.; Wu, Z.; Wang, L.; Zou, B.; Yang, Y.; Cui, Z.; Cai, Y.; Yu, T.; et al. MNN: A universal and efficient inference engine. *Proceedings of Machine Learning and Systems* **2020**, *2*, 1–13.
21. Lopes, N.P. Torchy: A tracing jit compiler for pytorch. In Proceedings of the Proceedings of the 32nd ACM SIGPLAN International Conference on Compiler Construction, 2023, pp. 98–109.
22. Aminabadi, R.Y.; Rajbhandari, S.; Awan, A.A.; Li, C.; Li, D.; Zheng, E.; Ruwase, O.; Smith, S.; Zhang, M.; Rasley, J.; et al. Deepspeed-inference: enabling efficient inference of transformer models at unprecedented scale. In Proceedings of the SC22: International Conference for High Performance Computing, Networking, Storage and Analysis. IEEE, 2022, pp. 1–15.
23. Sheng, Y.; Zheng, L.; Yuan, B.; Li, Z.; Ryabinin, M.; Chen, B.; Liang, P.; Ré, C.; Stoica, I.; Zhang, C. Flexgen: High-throughput generative inference of large language models with a single gpu. In Proceedings of the International Conference on Machine Learning. PMLR, 2023, pp. 31094–31116.
24. Song, Y.; Mi, Z.; Xie, H.; Chen, H. Powerinfer: Fast large language model serving with a consumer-grade gpu. *arXiv preprint arXiv:2312.12456* **2023**.

25. Piao, T.; Cho, I.; Kang, U. SensiMix: Sensitivity-Aware 8-bit index & 1-bit value mixed precision quantization for BERT compression. *PloS one* **2022**, *17*, e0265621.
26. Zhang, W.; Hou, L.; Yin, Y.; Shang, L.; Chen, X.; Jiang, X.; Liu, Q. Ternarybert: Distillation-aware ultra-low bit bert. *arXiv preprint arXiv:2009.12812* **2020**.
27. Qin, H.; Ding, Y.; Zhang, M.; Yan, Q.; Liu, A.; Dang, Q.; Liu, Z.; Liu, X. Bibert: Accurate fully binarized bert. *arXiv preprint arXiv:2203.06390* **2022**.
28. Kim, Y.J.; Henry, R.; Fahim, R.; Awadalla, H.H. Finequant: Unlocking efficiency with fine-grained weight-only quantization for llms. *arXiv preprint arXiv:2308.09723* **2023**.
29. Xiao, G.; Lin, J.; Seznec, M.; Wu, H.; Demouth, J.; Han, S. Smoothquant: Accurate and efficient post-training quantization for large language models. In Proceedings of the International Conference on Machine Learning. PMLR, 2023, pp. 38087–38099.
30. Rastegari, M.; Ordonez, V.; Redmon, J.; Farhadi, A. Xnor-net: Imagenet classification using binary convolutional neural networks. In Proceedings of the European conference on computer vision. Springer, 2016, pp. 525–542.
31. Frantar, E.; Alistarh, D. Sparsegpt: Massive language models can be accurately pruned in one-shot. In Proceedings of the International Conference on Machine Learning. PMLR, 2023, pp. 10323–10337.
32. Sun, M.; Liu, Z.; Bair, A.; Kolter, J.Z. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695* **2023**.
33. Yin, L.; Wu, Y.; Zhang, Z.; Hsieh, C.Y.; Wang, Y.; Jia, Y.; Pechenizkiy, M.; Liang, Y.; Wang, Z.; Liu, S. Outlier weighed layerwise sparsity (owl): A missing secret sauce for pruning llms to high sparsity. *arXiv preprint arXiv:2310.05175* **2023**.
34. Xu, P.; Shao, W.; Chen, M.; Tang, S.; Zhang, K.; Gao, P.; An, F.; Qiao, Y.; Luo, P. BESA: Pruning Large Language Models with Blockwise Parameter-Efficient Sparsity Allocation. *arXiv preprint arXiv:2402.16880* **2024**.
35. Syed, A.; Guo, P.H.; Sundarapandiyan, V. Prune and tune: Improving efficient pruning techniques for massive language models **2023**.
36. Zhang, Y.; Zhao, L.; Lin, M.; Sun, Y.; Yao, Y.; Han, X.; Tanner, J.; Liu, S.; Ji, R. Dynamic sparse no training: Training-free fine-tuning for sparse llms. *arXiv preprint arXiv:2310.08915* **2023**.
37. Boža, V. Fast and optimal weight update for pruned large language models. *arXiv preprint arXiv:2401.02938* **2024**.
38. Xia, H.; Zheng, Z.; Li, Y.; Zhuang, D.; Zhou, Z.; Qiu, X.; Li, Y.; Lin, W.; Song, S.L. Flash-llm: Enabling cost-effective and highly-efficient large generative model inference with unstructured sparsity. *arXiv preprint arXiv:2309.10285* **2023**.
39. Srinivasan, V.; Gandhi, D.; Thakker, U.; Prabhakar, R. Training large language models efficiently with sparsity and dataflow. *arXiv preprint arXiv:2304.05511* **2023**.
40. Hinton, G.; Vinyals, O.; Dean, J. Distilling the Knowledge in a Neural Network, 2015, [arXiv:stat.ML/1503.02531].
41. Gou, J.; Yu, B.; Maybank, S.J.; Tao, D. Knowledge distillation: A survey. *International Journal of Computer Vision* **2021**, *129*, 1789–1819.
42. Chen, Z.; Gao, Q.; Bosselut, A.; Sabharwal, A.; Richardson, K. Disco: distilling counterfactuals with large language models. *arXiv preprint arXiv:2212.10534* **2022**.
43. Gu, Y.; Zhang, S.; Usuyama, N.; Woldesenbet, Y.; Wong, C.; Sanapathi, P.; Wei, M.; Valluri, N.; Strandberg, E.; Naumann, T.; et al. Distilling large language models for biomedical knowledge extraction: A case study on adverse drug events. *arXiv preprint arXiv:2307.06439* **2023**.
44. Sahu, G.; Vechtomova, O.; Bahdanau, D.; Laradji, I.H. Promptmix: A class boundary augmentation method for large language model distillation. *arXiv preprint arXiv:2310.14192* **2023**.
45. Kurtic, E.; Kuznedelev, D.; Frantar, E.; Goin, M.; Alistarh, D. Sparse finetuning for inference acceleration of large language models. *arXiv preprint arXiv:2310.06927* **2023**.
46. Ahmad, Z.; Illanko, K.; Khan, N.; Androutsos, D. Human action recognition using convolutional neural network and depth sensor data. In Proceedings of the Proceedings of the 2019 International Conference on Information Technology and Computer Communications, 2019, pp. 1–5.

47. Mishra, A.; Latorre, J.A.; Pool, J.; Stosic, D.; Stosic, D.; Venkatesh, G.; Yu, C.; Micikevicius, P. Accelerating sparse deep neural networks. *arXiv preprint arXiv:2104.08378* **2021**.
48. Ding, N.; Qin, Y.; Yang, G.; Wei, F.; Yang, Z.; Su, Y.; Hu, S.; Chen, Y.; Chan, C.M.; Chen, W.; et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence* **2023**, *5*, 220–235.
49. Sun, K.; Luo, X.; Luo, M.Y. A survey of pretrained language models. In Proceedings of the International Conference on Knowledge Science, Engineering and Management. Springer, 2022, pp. 442–456.
50. Min, B.; Ross, H.; Sulem, E.; Veyseh, A.P.B.; Nguyen, T.H.; Sainz, O.; Agirre, E.; Heintz, I.; Roth, D. Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Computing Surveys* **2023**, *56*, 1–40.
51. Xu, L.; Xie, H.; Qin, S.Z.J.; Tao, X.; Wang, F.L. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment. *arXiv preprint arXiv:2312.12148* **2023**.
52. Fu, Z.; Yang, H.; So, A.M.C.; Lam, W.; Bing, L.; Collier, N. On the effectiveness of parameter-efficient fine-tuning. In Proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, 2023, Vol. 37, pp. 12799–12807.
53. Hu, E.J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685* **2021**.
54. Xu, Y.; Xie, L.; Gu, X.; Chen, X.; Chang, H.; Zhang, H.; Chen, Z.; Zhang, X.; Tian, Q. Qa-lora: Quantization-aware low-rank adaptation of large language models. *arXiv preprint arXiv:2309.14717* **2023**.
55. He, R.; Liu, L.; Ye, H.; Tan, Q.; Ding, B.; Cheng, L.; Low, J.W.; Bing, L.; Si, L. On the effectiveness of adapter-based tuning for pretrained language model adaptation. *arXiv preprint arXiv:2106.03164* **2021**.
56. Li, X.L.; Liang, P. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190* **2021**.
57. Lester, B.; Al-Rfou, R.; Constant, N. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691* **2021**.
58. Fan, S.; Rong, Y.; Meng, C.; Cao, Z.; Wang, S.; Zheng, Z.; Wu, C.; Long, G.; Yang, J.; Xia, L.; et al. DAPPLE: A pipelined data parallel approach for training large models. In Proceedings of the Proceedings of the 26th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, 2021, pp. 431–445.
59. Jiao, X.; Yin, Y.; Shang, L.; Jiang, X.; Chen, X.; Li, L.; Wang, F.; Liu, Q. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351* **2019**.
60. Wang, W.; Wei, F.; Dong, L.; Bao, H.; Yang, N.; Zhou, M. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems* **2020**, *33*, 5776–5788.
61. Jiao, X.; Chang, H.; Yin, Y.; Shang, L.; Jiang, X.; Chen, X.; Li, L.; Wang, F.; Liu, Q. Improving task-agnostic BERT distillation with layer mapping search. *Neurocomputing* **2021**, *461*, 194–203.
62. Loshchilov, I.; Hutter, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101* **2017**.
63. Yin, Y.; Chen, C.; Shang, L.; Jiang, X.; Chen, X.; Liu, Q. Autotinybert: Automatic hyper-parameter optimization for efficient pre-trained language models. *arXiv preprint arXiv:2107.13686* **2021**.
64. Koehn, P.; Chaudhary, V.; El-Kishky, A.; Goyal, N.; Chen, P.J.; Guzmán, F. Findings of the WMT 2020 shared task on parallel corpus filtering and alignment. In Proceedings of the Proceedings of the Fifth Conference on Machine Translation, 2020, pp. 726–742.
65. Tian, L.; Wong, D.F.; Chao, L.S.; Quaresma, P.; Oliveira, F.; Yi, L. UM-Corpus: A Large English-Chinese Parallel Corpus for Statistical Machine Translation. In Proceedings of the LREC, 2014, pp. 1837–1842.
66. Wu, J.; Zheng, H.; Zhao, B.; Li, Y.; Yan, B.; Liang, R.; Wang, W.; Zhou, S.; Lin, G.; Fu, Y.; et al. Ai challenger: A large-scale dataset for going deeper in image understanding. *arXiv preprint arXiv:1711.06475* **2017**.
67. Sajjad, H.; Dalvi, F.; Durrani, N.; Nakov, P. On the effect of dropping layers of pre-trained transformer models. *Computer Speech & Language* **2023**, *77*, 101429.
68. Sanh, V.; Debut, L.; Chaumond, J.; Wolf, T. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108* **2019**.

69. Zeng, A.; Liu, X.; Du, Z.; Wang, Z.; Lai, H.; Ding, M.; Yang, Z.; Xu, Y.; Zheng, W.; Xia, X.; et al. Glm-130b: An open bilingual pre-trained model. *arXiv preprint arXiv:2210.02414* **2022**.
70. Du, Z.; Qian, Y.; Liu, X.; Ding, M.; Qiu, J.; Yang, Z.; Tang, J. GLM: General Language Model Pretraining with Autoregressive Blank Infilling. In Proceedings of the Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2022, pp. 320–335.