Don't Stop Pretraining - Continuous Training for Enterprise Large-Scale Model Training

Anonymous ACL submission

Abstract

Artificial Intelligence (AI) has witnessed significant advancements in recent years, particularly in the field of Large Language Models (LLMs). While there has been a surge in the application of AI in consumer-oriented chat dialogue systems, it is crucial to address the challenges associated with implementing large models in enterprise applications, such as data management and cost optimization. Therefore, we present a study across concept and method of large model operation, showing that the principle and technology of revolves around the continuous updating of training data, including both pre-training data and fine-tuning data, and showed that the final goal of domain model training could be achieved through continuous operation of the model. Moreover, the optimization process focuses on calculating and comparing the total cost during the operation of a large model, additionally a time distribution calculation method is employed to reduce the initial input cost of computing power. This optimization approach is particularly suitable for the long-term operation of large models. As a by-product of the validation process, we also obtained preliminary data on how often large models forget (both natural forgetting and forced updating) what they learned, which has implications for knowledge updating and error correction. Overall, our consistent findings indicate that the industry's ability to operate large models continues to improve, leading to a reduction in the cost of their use.

1 Introduction

Large Language Models (LLMs) have emerged as a significant development trend in the field of AI(artificial intelligence)(Yuan et al., 2022). From Llama2(Touvron et al., 2023), ERNIE(Sun et al., 2019) to GPT-4(OpenAI et al., 2023), these models are characterized by their massive size, giant data requirements, and scale computational resources. With the implementation of large model services in enterprise application services, several challenges that need to be addressed. First, the public training data of general large model is inconsistent with enterprise private business data(Weidinger et al., 2021); another challenge is the increasing cost of large model training(Narayanan et al., 2021), which can be exacerbated by the growing demand for computing power; and then large language models have to face the constant knowledge updating(Zhong et al., 2023).

We use the continuous pre-training(Radford et al., 2018) method, it's an effective way to address the security contradiction that arises when training enterprise large models by using sensitive private business data. Existing open large model services or open source large models are trained using data that can be collected on the Internet, while internal enterprise data cannot be obtained through open channels due to security reasons (see Figure 1). Traditional Large Language Model (LLM) service data preparation is carried out through various techniques such as prompt engineering, model fine-tuning(Behnia et al., 2022), and retrieval-augmented generation(RAG)(Lewis et al., 2021). These techniques aim to optimize the performance of an LLM by improving its understanding of specific tasks and data-sets. The performance of a large model is fundamentally determined by the number of parameters it contains. Prompt engineering and fine-tuning can improve its performance on specialized tasks by allowing it to touch private domain knowledge. But, it is important to note these methods alone cannot undertake a lot of training data-set by limited knowledge capacity. RAG is a technique provide models with the ability to access vast external information and generate more accurate and diverse responses. However, there are certain limitations to these models when it comes to generalization and emergence.

In our work, we are aligning, blocking, and en-

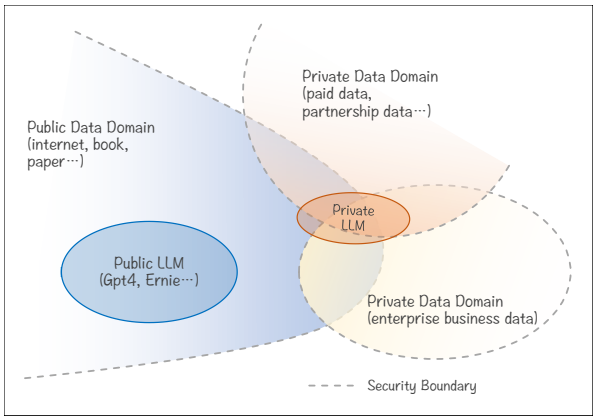


Figure 1: An illustration of private LLM training data distributions. Pre-training data is comprised of public and private domain data, usually cross security boundary (light grey ellipsis) within an even larger target domain, which is not necessarily one of the domains included in the original LM pre-training domain – though overlap is possible. We explore the benefits of continued pre-training on data from the public and private domain distribution.

hancing private training data for continuous pre-training, aim to maximize the knowledge capacity of the LLM while retaining the characteristics of it. In the private data domain, continuous training takes place to enhance security. The private professional data-set is divided based on Ebbinghaus' memory curve(Ebbinghaus, 1885), and merging the generalized learning data set for iterative training. After many iterations of the model, automated and manual tests are conducted to evaluate its performance. The results show that the large model trained in this way performs better than prompt engineering and model fine-tuning, and is on par with the performance of RAG.

Our method, a sustainable approach to pre-training, also reduce the pressure on computing costs., businesses can save on expenses and improve overall efficiency. Today large models training require massive amounts of training data and computational power, often reaches millions or even tens of millions dollar. For example, GPT-3 175B model required 3.14E23 FLOPs of computing for training, even at theoretical 28T FLOPs for V100, this will take 355GPU years and cost $4.6M for a single training run(Brown et al., 2020). This has led to the search for more practical and cost-effective infrastructure solutions that can efficiently optimize model training and reasoning. Continuous model pre-training, which involves scheduling idle inference resources to exchange time for space. By reducing the scale of cluster computing power traditionally used for pre-training large models, the small cluster and multi-iteration method can achieve the same effect. This method is advantageous because it allows businesses to utilize their existing resources more efficiently, without the need for extensive investments in high-end computing infrastructure. Additionally, it promotes a more sustainable approach to AI development by reducing the environmental impact of excessive energy consumption and waste associated with large-scale computing resources. We while cutting costs by more than 50%.

In the operation of model services, it is crucial to continuously update the knowledge of the private domain. This ensures that the models remain accurate, up-to-date, and capable of delivering reliable results. In order to investigate effective methods of knowledge renewal, we conducted experiments comparing two approaches: natural knowledge forgetting and forced knowledge renewal. Through the analysis of experimental data, we have concluded that forced updating of general knowledge is faster than precise numerical knowledge. Unfortunately, we have not been able to observe obvious natural forgetting in limited iterations. More research is needed for the root cause.

2 Continuous Training Domain LLM

Continuous pre-training refers to the process of training a base version of a model using existing data and then further refining the model using additional industry knowledge. The goal is to enhance the model's ability to understand and predict outcomes in a specific domain.In this section, we provide detail of the theoretical basis of continuous pre-training, data preparation methods, training methods and test results.

2.1 Theoretical Basis

The field of natural language processing (NLP) has witnessed significant advancements, particularly in the area of pre-training language models (LLMs). Pre-training involves training a model on a large corpus of unlabeled text to learn general language representations. This general pre-training is typically followed by fine-tuning on specific tasks using labeled data. However, this two-step process may not fully exploit the potential of domain-specific or task-specific data. To address

this limitation, we have proposed a technique called continual pre-training. Continual pre-training aims to seamlessly connect the stages of general pre-training and supervised fine-tuning, thereby enhancing the model's performance in specific domains or tasks(see Figure 2).

It achieves this by extend the training process to the operational stage with the same objective as general pre-training while focusing on unlabeled corpora tailored to specific domains or tasks.

By our method, we focus on utilizing decoder-only transformer models, which represent the current mainstream LLM architecture. The model is trained to predict the next token in a given sequence of tokens, given the preceding context. This objective encourages the model to learn meaningful representations that capture the syntactic and semantic structures of natural language:

$$\max_{\theta} \sum_{i=1}^{N} \log P(u_i | u_{<i}; \theta), \tag{1}$$

Given an unsupervised corpus of tokens u, where N is the size of the context window, and the conditional probability P is modeled using a neural network with parameters θ. These parameters are trained using stochastic gradient descent.

Pure forgetting curve is exponential in nature. Due to the fact that Ebbinghaus used the saving on relearning rather than recall, the curve can be best described by a fast-decay power function. Today, we know that forgetting is exponential in nature. It is very similar to radioactive decay. It is governed by similar principles.(Wozniak, 2017) The forgetting curve is expressed using an exponential decay model: $R = e^{-\frac{t}{S}}$, where R is the memory retention, or what fraction of the information can be retained. t is the time elapsed since learning the information. e is the base of the natural logarithm, approximately equal to 2.71828. S is the memory strength, which changes based on factors such as the depth of learning and the amount of repetition.

In our research, we have extended the DSR (Wozniak, 2018) algorithm based on the Ebbinghaus curve by replacing the time-based parameter with one based on the number of repetitions. This modification is rooted in the understanding that unlike human memory, which decays over time, the parameters of electronic storage do not necessarily decay over time. Instead, these parameters are subject to adjustments due to continuous training iterations, potentially resulting in overwriting or loss of previously formed memories. Consequently, it is essential to strengthen these memories in subsequent training. The new memory curve algorithm is described below. Exponential memory curve model:

$$R = e^{\ln 0.9 \cdot \frac{N}{S}}, \tag{2}$$

- Initial stability: $S_0 = 0.5$
- Number of training iterations: $N >= 0$
- Memory stability parameter calculation: $S_L = S_0^{-f \cdot L}$
  - Count of memory: $L >= 0$
  - Attenuation coefficient: $f = 0.3$

Using the above algorithm, we draw a diagram of interval repetition, and use the interval repetition data to slice the continuous training data set to achieve the balance between training requirements and training costs.

Using the above algorithm, we draw a diagram of interval repetition(see Figure 3).

See Appendix A for detail.

2.2 Data Preparation

In this paper, we focus on enhancing the continuous knowledge acquisition performance of LLMs through continual pre-training. Training of general LLMs has already used a large amount of internet data, so to ensure the effectiveness of knowledge training, we collected a large amount of private data (IDC information). These data are not exposed on the internet, so training and testing based on this dataset can verify the results of continuous pre-training. Furthermore, in order to preserve the world knowledge and generalization ability of LLMs, we also incorporate general text data into the training data-set.

Data Pre-processing To maintain the quality of pre-training private datasets, we have implemented a comprehensive pipeline that consists of data reformulation, filtering, and deduplication processes. This pipeline is designed to ensure that the data used for training is of the highest quality and reliability. In order to establish a standard for our private data center's data, we have used the GB50174-2017 Data Center Design Specification(MOHURD, 2017) as data reference. This specification provides

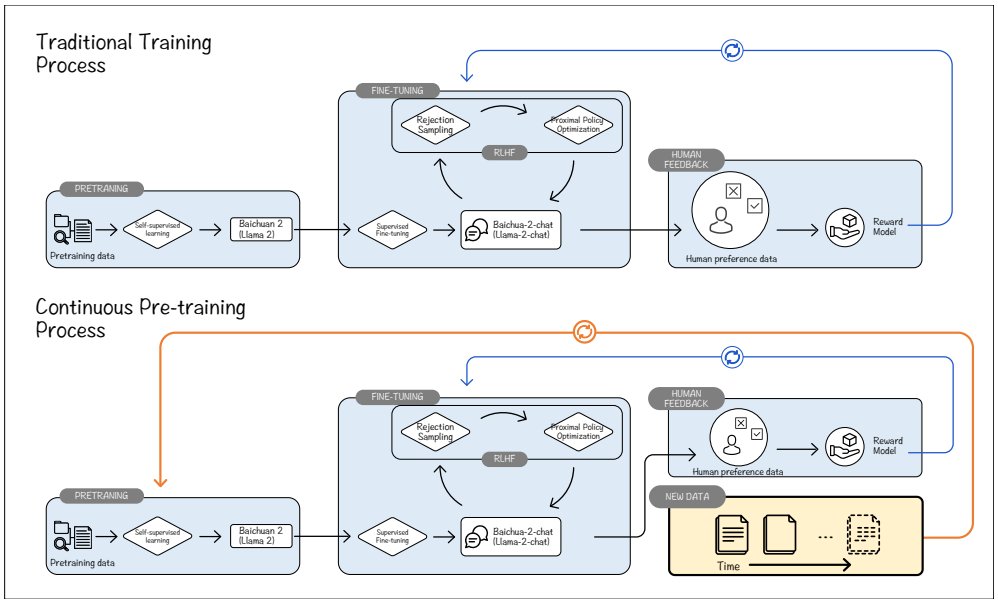


Figure 2: Traditional large model training deployment process involves several stages, including one-time Pre-training and multiple iterations Fine-tuning. The limitations of this process are primarily rooted in the fine-tuning stage. One of the main issues is the limited knowledge capacity of the fine-tuning stage. Additionally, continuous fine-tuning the fine-tuned model further may lead to losing the previous ability. This issue can be addressed by optimized fine-tuning data sets, but it still faces the problem of an upper limit and dilution of the original knowledge. The process of improving large model training deployment involves multiple iterations of pre-training, which allows for the incorporation of new and expanded knowledge(continuously acquire new knowledge over time) into the large model. This gradual transformation of the general large model into a professional large model is achieved through a series of pre-training rounds, with the pre-trained base model acting as the foundation for each fine-tuning session.

guidelines for the proper handling and management of data in a data center environment, ensuring that the data is consistent and reliable.

Dataset Mixing Strategy During the continuous pre-training process, sliding window sampling data is used for publicly available internet data. Each time, 50M new data are taken from the original data-set as the next iteration's incremental data. Each time the training on public data is maintained around 500M. In the private domain, data selection is carried out using a repeated interval method based on the Ebbinghaus forgetting curve(see Figure 4).

An example is illustrated in Table 4.

See Appendix B for detail.

2.3 Continuous Pre-training

We select Baicuan 2 (Yang et al., 2023) as the backbone models of continuous pre-training. In the process of model continual pre-training, it is essential to monitor development through various stages of training. This involves a detailed assessment of multiple model checkpoints, which are saved states of the model at different intervals during the

	Public Data	Private Domain Data				
Round	500M	Each IDC has ∼50M data				Total
Rd-1	SW_Data_1	IDC1_Data				∼550M
Rd-2	SW_Data_2	IDC2_Data	IDC1_Data			∼600M
Rd-3	SW_Data_3	IDC3_Data	IDC2_Data			∼600M
Rd-4	SW_Data_4	IDC4_Data	IDC3_Data	IDC1_Data		∼650M
Rd-5	SW_Data_5	IDC5_Data	IDC4_Data	IDC2_Data		∼650M
Rd-6	SW_Data_6	IDC6_Data	IDC5_Data	IDC3_Data		∼650M
Rd-7	SW_Data_7	IDC7_Data	IDC6_Data	IDC4_Data	IDC1_Data	∼700M
Rd-8	SW_Data_8	IDC8_Data	IDC7_Data	IDC5_Data	IDC2_Data	∼700M
Rd-9	SW_Data_9	IDC9_Data	IDC8_Data	IDC6_Data	IDC3_Data	∼700M
Rd-10	SW_Data_10	IDC10_Data	IDC9_Data	IDC7_Data	IDC4_Data	∼700M
Rd-11	SW_Data_11	IDC11_Data	IDC10_Data	IDC8_Data	IDC5_Data	∼700M
Rd-12	SW_Data_12	IDC12_Data	IDC11_Data	IDC9_Data	IDC6_Data	∼700M

Table 1: The pre-training data size. The actual amount of data per IDC is approximately 50M(±10%), and this is only an approximation value.

continual pre-training phase. When examining Figure 5(a), certain key steps might stand out where significant changes in the loss curve occur. This approach contrasts with standard pre-training, the loss function curve obtained through continuous pre-training converges faster compared to the standard pre-training loss curve. Additional, during each round of iterative training, we train for 500 extra steps (without adding new data), and compare the loss curve obtained with the loss curve of the next iteration when new data is added. See Figure5(b) to Figure5(i). As the number of itera-

tions increases, this impact gradually diminishes, proving that the model has learned new knowledge. More accurate training results are shown through test data.

See Appendix C for detail.

2.4 Evaluation

For autoregressive LLM, each task is considered as a text generation task. Therefore, we can use automatic evaluation metrics for text generation to assess the model's performance on different tasks. In order to more accurately evaluate the performance of private domain model after continuous training, we conducted tests based on evaluation research related to large models(Chang et al., 2023). We create a benchmark test data-set based on private domain training data and tested the model from two aspects: automated evaluation and manual evaluation. For the interference of random seed (Random seed) in the model output process during the training, we adopted a 3-round zero-shot method for testing and recorded the highest score.

The Table 2 shows the automated and manual test results of the knowledge output by the large model from different data centers at different iterative stages of continuous training. From the test results, the accuracy of the trained private domain knowledge is significantly higher than that of the untrained model. With the increase of iterative rounds, the accuracy gradually improves. The knowledge accuracy of the data center and the control group are very close after 3 interval repetitions. The knowledge accuracy of individual data centers is higher than that of the control group, which may be caused by the amount and quality of training data, and further research is needed. The accuracy of the control group fluctuates less, indicating that the continuous pre-training of the model has little effect on the original model.

See Appendix D for detail.

2.5 Result

By constructing private domain validation data-set after multiple iterations of testing, evaluation provides a direct reflection of the impact of continuous pre-training model capabilities. Experiment results demonstrate that:

(1) Continual pre-training with a combination of domain-specific and general data is crucial for enhancing model capabilities across different subsets of private domain knowledge, as traditional pre-training and fine-tuning alone are insufficient for sustainable training of new knowledge into large models.

(2) The interval repetition algorithm, based on the Ebbinghaus memory curve, is an efficient approach to training large models by addressing challenges in traditional pre-training methods through data mixing and iterative small batch retraining, ultimately achieving high accuracy rates with limited iterations.

3 Continuous Training Cost

Training large language models (LLMs) is a significant topic that cannot be overlooked, as the computational power required for model training, especially pre-training, directly translates to cost. In order to reduce costs, we propose a continuous pre-training method that effectively reduces the number of tokens involved in the training, compared to traditional methods. This approach leads to successful reductions in computational resources, computational cluster scale, and GPU utilization.

3.1 Theoretical Basis

In this essay, we will discuss the two articles(Kaplan et al., 2020)(Hoffmann et al., 2022) by OpenAI and DeepMind that introduced factors affecting model performance and how to prepare models and training data within a given budget. It's tricky that the concepts first proposed by OpenAI were later challenged by DeepMind, which may be related to the number of model parameters and learning rate. Nevertheless, several consistent conclusions can be reached:

- Model performance depends most strongly on scale, which consists of three factors: the number of model parameters N (excluding embeddings), the size of the data-set D, and the amount of compute C used for training.

- **Smooth power laws:** Performance has a power-law relationship with each of the three scale factors N, D, C when not bottle-necked by the other two.

- When working within a fixed compute budget C but without any other restrictions on the model size N or available data D, N and D should be increased in proportion.

Based on the above analysis and the architecture of transformer model, it can be deduced that given

Round		IDC1	IDC2	IDC3	IDC4	IDC5	IDC6	IDC7	IDC8		Control Group
Rd-1	Auto	29.2	1.4	4.9	2.4	5.6	2.0	3.8	1.0	|	33.3
	Human	57.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	|	83.1
Rd-2	Auto	26.6	24.1	9.4	4.3	3.1	1.0	2.8	1.5	|	35.5
	Human	79.7	70.6	0.0	0.0	0.0	0.0	0.0	0.0	|	82.6
Rd-3	Auto	32.7	30.6	24.2	7.5	3.4	2.0	4.6	1.9	|	35.1
	Human	84.2	72.2	53.9	0.0	0.0	0.0	0.0	0.0	|	79.4
Rd-4	Auto	38.2	41.2	38.2	19.5	4.5	1.0	4.9	1.7	|	33.9
	Human	66.5	68.3	70.5	45.6	0.0	0.0	0.0	0.0	|	78.9
Rd-5	Auto	38.8	30.9	30.3	25.9	23.4	2.0	4.5	1.0	|	32.6
	Human	73.1	71.1	58.4	65.0	57.5	0.0	0.0	0.0	|	80.3
Rd-6	Auto	35.5	43.3	36.8	25.9	36.1	21.5	3.8	1.6	|	32.5
	Human	88.2	**77.7**	90.7	62.4	42.2	43.3	0.0	0.0	|	80.9
Rd-7	Auto	32.7	40.5	36.0	32.9	28.5	26.9	27.1	1.4	|	31.3
	Human	**92.3**	60.9	94.1	64.3	66.0	62.0	55.2	0.0	|	**83.2**
Rd-8	Auto	37.0	39.9	37.3	31.1	37.8	30.7	35.4	20.2	|	32.9
	Human	90.5	65.4	**93.2**	**70.9**	**73.7**	**65.1**	**71.5**	**44.7**	|	82.3

Table 2: The table displays the testing results of automated and manual testing in various iteration cycles across different data centers. Automated testing uses the cosine similarity algorithm to compare vectorized strings of characters and applies normalization. Manual testing involves evaluating the model's answers based on expert opinions, with scores ranging from 0 to 5, where higher scores indicate better answers. All scores are converted to a percentage for comparison purposes.

the number of training tokens D, model parameters N and hardware environment configuration, the calculation time cost for training transformer model is as follows(JMXGODLZ, 2023):

$$Time \approx \frac{8 \times D \times N}{GPU_{units} \times GPU_{peakflops} \times GPU_{utilization}} \tag{3}$$

Therefore, when the model parameter N is fixed, the size of the training tokens D is directly proportional to the training time, the GPU cluster scale and GPU utilization rate are inversely proportional to the training time. Next we will discuss how to save the cost of training from these three aspects.

3.2 Training Tokens

We use the interval algorithm, is based on the Ebbinghaus memory curve. It is a method used to segment continuously trained data-set. The growth curve of the data set is a function based $\ln(n)$, where n is the number of pre-training iterations. So in order to obtain the total number of tokens at the Nth iteration, we need to integrate this function:

$$\int \ln(x)dx = x\ln(x) - x + c \tag{4}$$

Therefore, the definite integral in the interval $[1, n]$ is:

$$\int_{x=1}^{n} \ln(x)dx = n\ln(n) - n + 1 \tag{5}$$

We can observe that this function grows logarithmically. Traditional single-increment tokens is a linear function, so the total computation is a quadratic function. Because the method of mixing the data is mixed proportionally, it does not change the functional form of the tokens increase. As a result, in a finite number of iterations (15-30 times), the traditional training method's tokens is 2.5-4 times the volume of the interval repetition method's training tokens.

3.3 Time Exchange Scale

In the field of artificial intelligence (AI), the Total Cost of Ownership (TCO) is a crucial concept used to evaluate and quantify all the costs an organization or business incurs when using AI services. TCO takes into account not only the direct costs of computing power but also various other expenses related to AI services, including hardware, software, human resources, and operations.

The concept of "The Long Tail" was first introduced by Chris Anderson, the editor-in-chief of Wired magazine, in his article "The Long Tail" in October 2004. It is used to describe the business and economic models of websites such as Amazon and Netflix. "The Long Tail" is essentially a colloquial expression of statistical features such as power laws and Pareto distributions in economics. In the context of AI large model services, the TCO curve is an example of the long tail theory. The total investment in the model training phase is sub-

stantial, while the operational phase represents the long tail. Continuous training and inference methods involve shifting some of the peak investment in the training phase to the operational phase, smoothing out the investment curve.

In traditional training-inference models, the training phase heavily utilizes high computing power GPU clusters like A100, while the inference phase uses 4090 inference clusters. This is mainly due to the enormous computing power demand during pre-training phases. However, with continuous pre-training during the operational phase, there is relatively more time available, and since the total token count for overall training is controllable, using inference clusters with relatively weaker computing power may take more time but offers easier cost control.

In our experimental scenario using the same dataset, traditional pre-training with A100 clusters took a total of 120 hours. In contrast, employing a continuous pre-training method with a halved scale 4090 cluster for splitting pre-training datasets resulted in a total computation time of 600 hours to achieve similar results while reducing overall costs by over 50%.

3.4 GPU Utilization

On the other hand, training business large-scale models on a single GPU or a single server has become infeasible due to the high computational and storage demands. To address this issue, distributed parallel computing clusters have emerged as a solution. By distributing the computational tasks across multiple servers, the training speed can be significantly accelerated.

As the complexity of models increases, the need for data transfer and interaction between GPUs and servers also arises. The large volume of data in these models poses a significant challenge in the machine learning training process, as data transfer speed becomes a bottleneck, directly affecting the utilization of GPUs. Although there are various optimization techniques, improving the GPU usage rate remains a major challenge.

One simple approach to improve GPU utilization is to reduce the amount of data transferred. By adopting the interval repetition algorithm for continuous pre-training, the size of the training dataset can be effectively reduced, alleviating the bottleneck of data transfer, and thus improving GPU utilization. In our test environment, the GPU usage rate can be increased to over 63%.

3.5 Result

Continuous pre-training involves training a model with fixed parameters in repeated intervals to reduce data tokens, decrease cluster size, improve GPU peak utilization rate, and enhance processing efficiency. However, it's important to consider the potential impact on total training time and overall cost.

4 Knowledge Update

In the process of continuous training, we encounter an interesting phenomena that warrant further investigation. This phenomenon, related to knowledge renewal or knowledge forgetting, has been observed and documented. While we have made progress in understanding it, the theoretical basis and underlying laws still require more research. In this paragraph, we will expand on some discussions, including some speculation and research directions.

4.1 Background

Knowledge refers to the process of update specific learned knowledge from a machine learning model without compromising its overall performance. This is particularly important in situations where privacy concerns and data regulations require the update of private information from a trained model. The knowledge update process can be divided into two steps: knowledge forgetting and learning new knowledge. New knowledge can be learned through continuous pre-training, which makes knowledge forgetting a topic that requires additional attention. Or call it Machine Unlearning.

Machine Unlearning is a relatively new and innovative field in the realm of artificial intelligence (AI) research. The term has gained significant attention due to a notable work presented at the top academic conference S&P - 42nd IEEE Symposium of Security and Privacy. The paper, titled "Machine Unlearning"(Yao et al., 2023). Due to its novelty, not many people are familiar with this research direction, but it has immense potential for future advancements in AI.

According to the difference between the model obtained after forgetting and the retraining model, the forgetting algorithm can be divided into two types(Whale, 2023):

- **Precise Forgetting:** precise forgetting represents that the model after forgetting and the model after retraining are exactly the same in parameter distribution, with no difference.
- **Approximate Forgetting:** The model obtained by the approximate forgetting algorithm usually still contains information from the data that needs to be forgotten. Its parameter distribution has some differences from the retraining model, but its output is basically the same as that of the retraining model.

4.2 Experiments

Our experimental method is to consciously change the training data in the iterative process of continuous pre-training to simulate the update of knowledge. We have updated two types of data in the stable knowledge: general knowledge and precise numerical value. Observed the training effect of the two types of data in the subsequent iterative training process. We used the expert scoring method in this part for evaluation. According to the Table 3, general knowledge data needs more than 4 iterations to have a significant effect, and accurate numerical value needs nearly 7 iterations to have a certain effect.

Round	IDC1		IDC2	
	Force Update		Natural Forgetting	
	general knowledge	precise numerical knowledge	general knowledge	precise numerical knowledge
Rd-1	-	-	-	-
Rd-2	<5.0	<5.0	-	-
Rd-3	15.1	<5.0	<5.0	<5.0
Rd-4	34.4	<5.0	<5.0	<5.0
Rd-5	77.8	<5.0	<5.0	<5.0
Rd-6	90.3	9.7	<5.0	<5.0
Rd-7	92.6	22.4	<5.0	<5.0
Rd-8	94.0	71.3	<5.0	<5.0

Table 3: The large models forget experimental data, which is the result of expert evaluation.

At the same time, we also deleted some data in the subsequent training and no longer trained to simulate the natural forgetting of knowledge.The test of this part of the data did not find a significant drop in data.

4.3 Result

Based on these data, we can draw these conclusions: the updating of knowledge in a model is influenced by the strength of correlation between new and existing data, with general knowledge being updated faster than accurate numerical knowledge due to the latter's weaker relevance, potentially requiring more iterations to update effectively without overfitting.

5 Conclusion

In this paper, we present an extensive investigation into the method of continuous pre-training for large language models (LLMs), with a focus on the application of the interval repetition algorithm based on Ebbinghus's memory curve in the pre-training process. Our experiments demonstrate that continuous pretraining enables the large model to have its knowledge updated continuously while maintaining its generalization ability for other trained tasks. Moreover, a domain-specific corpora mixing strategy is designed to enhance the model's ability to absorb and integrate new knowledge over time, ultimately improving the continuous adaptability of the large model in the operation process.

We also verified that under the same training data size, continuous pre-training can reduce cluster size and data transmission peak value to improving GPU utilization rates. While it may result in a certain increase in training time, this approach provides tangible benefits that contribute to overall TCO savings and operational efficiency.

In addition, during the experimental process, we also observed that through continuous iterative training, general knowledge and precise numerical knowledge can be forcibly updated. However, it also raises some issues regarding knowledge forgetting, which require further research support and validation.

Our work provide a new method for the continuous updating of knowledge in the operation process of large models, especially in the case of time characteristics of new knowledge and limited inference power in the operational environment. In the future, we plan to research more complex knowledge updating algorithms to bring more effective performance improvements to domain-specific LLMs. Additionally, we will also make improvements on evaluation benchmarks to more accurately assess the capabilities of large models in different training stages.

6 Limitation

We still have some limitaion, will be completed in the future:

(1)Validation on open-source datasets is convenient for reproducibility. The main challenge is that the data has already been trained, so the expected performance should be better than the current situation.

(2)The current model being used is "Bai Chuan 2", mainly to support Chinese. The next step of work needs to be carried out on "Llama 2" so that the training work is more representative.

(3)Based on the forgetting curve, the interval repetition currently uses a classical algorithm. Subsequently, a more efficient algorithm needs to be utilized, referencing interval repetition with a Markovian memory model. A scheduling program for interval repetition that guarantees minimal review cost was designed using a random shortest path algorithm. Experimental results indicate a 12.6% improvement in performance compared to state-of-the-art methods.(Ye et al., 2022)

(4)Improving automated evaluation methods and benchmarks can more accurately assess the effectiveness of training data. It can accurately identify the knowledge that the model has successfully learned and the knowledge that is yet to be fully learned.

(5)By utilizing the results of automated evaluations, feedback to the next iteration of the data preparation process, further enhancing the efficiency of data preparation and saving on training costs.

References

Rouzbeh Behnia, Mohammadreza Reza Ebrahimi, Jason Pacheco, and Balaji Padmanabhan. 2022. Ewtune: A framework for privately fine-tuning large language models with differential privacy. In *2022 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. 2023. A survey on evaluation of large language models. *arXiv arXiv:2307.03109*.

Hermann Ebbinghaus. 1885. Memory: A contribution to experimental psychology. teachers college, columbia university, new york. *Trans. HA Ruger and CE Bussenius. Original work published*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. 2022. Training compute-optimal large language models. *arXiv arXiv:2203.15556*.

JMXGODLZ. 2023. Training time estimate. https://zhuanlan.zhihu.com/p/639872915, Last accessed on 2023-12-10.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv arXiv:2001.08361*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2022. Cross-task generalization via natural language crowdsourcing instructions. *arXiv arXiv:2104.08773*.

China MOHURD. 2017. Data center design specifications(2017). https://www.mohurd.gov.cn/gongkai/zhengce/zhengcefilelib/201706/20170629_232416.html, Last accessed on 2023-10-28.

Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Anand Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, Amar Phanishayee, and Matei Zaharia. 2021. Efficient large-scale language model training on gpu clusters using megatron-lm. *arXiv preprint arXiv:2104.04473*.

Hien Ngoc Nguyen, Ganix Lasa, Ion Iriarte, Ariane Atxa, Gorka Unamuno, and Gurutz Galfarsoro. 2022. Datasets of skills-rating questionnaires for advanced service design through expert knowledge elicitation. *Scientific Data*, 9(1):321.

OpenAI, :, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mo Bavarian, Jeff Belgum, Irwan Bello, Jake

Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2023. Gpt-4 technical report. *arXiv preprint arXiv:22303.08774*.

OpenLMLab. 2023. moss 002 sft data. https://huggingface.co/datasets/fnlp/moss-002-sft-data, Last accessed on 2023-10-28.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. *unknown*.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '20. IEEE Press.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 3505–3506, New York, NY, USA. Association for Computing Machinery.

Dmitriy Selivanov. 1996. text2vec. http://text2vec.org/, Last accessed on 2023-12-16.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan

Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Tianwen Wei, Liang Zhao, Lichang Zhang, Bo Zhu, Lijie Wang, Haihua Yang, Biye Li, Cheng Cheng, Weiwei Lü, Rui Hu, Chenxia Li, Liu Yang, Xilin Luo, Xuejie Wu, Lunan Liu, Wenjun Cheng, Peng Cheng, Jianhao Zhang, Xiaoyu Zhang, Lei Lin, Xiaokun Wang, Yutuan Ma, Chuanhai Dong, Yanqi Sun, Yifu Chen, Yongyi Peng, Xiaojuan Liang, Shuicheng Yan, Han Fang, and Yahui Zhou. 2023. Skywork: A more open bilingual foundation model.

Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, Zac Kenton, Sasha Brown, Will Hawkins, Tom Stepleton, Courtney Biles, Abeba Birhane, Julia Haas, Laura Rimell, Lisa Anne Hendricks, William Isaac, Sean Legassick, Geoffrey Irving, and Iason Gabriel. 2021. Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359*.

Big Whale. 2023. Answer of how to evaluate model forgetting. https://www.zhihu.com/question/382537280/answer/3262820552, Last accessed on 2023-12-12.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing. *arXiv arXiv:1910.03771*.

Piotr Wozniak. 2017. Error of ebbinghaus forgetting curve. https://supermemo.guru/wiki/Error_of_Ebbinghaus_forgetting_curve, Last accessed on 2023-12-26.

Piotr Wozniak. 2018. Algorithm sm-17. https://supermemo.guru/wiki/Algorithm_SM-17, Last accessed on 2023-12-26.

Aiyuan Yang, Bin Xiao, Bingning Wang, Borong Zhang, Ce Bian, Chao Yin, Chenxu Lv, Da Pan, Dian Wang, Dong Yan, Fan Yang, Fei Deng, Feng Wang, Feng Liu, Guangwei Ai, Guosheng Dong, Haizhou Zhao, Hang Xu, Haoze Sun, Hongda Zhang, Hui Liu, Jiaming Ji, Jian Xie, JunTao Dai, Kun Fang, Lei Su, Liang Song, Lifeng Liu, Liyun Ru, Luyao Ma, Mang Wang, Mickel Liu, MingAn Lin, Nuolan Nie, Peidong Guo, Ruiyang Sun, Tao Zhang, Tianpeng Li, Tianyu Li, Wei Cheng, Weipeng Chen, Xiangrong Zeng, Xiaochuan Wang, Xiaoxi Chen, Xin Men, Xin Yu, Xuehai Pan, Yanjun Shen, Yiding Wang, Yiyu Li, Youxin Jiang, Yuchen Gao, Yupeng Zhang, Zenan Zhou, and Zhiying Wu. 2023. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*.

Yuanshun Yao, Xiaojun Xu, and Yang Liu. 2023. Large language model unlearning. *arXiv arXiv:2310.10683*.

Junyao Ye, Jingyong Su, and Yilong Cao. 2022. A stochastic shortest path algorithm for optimizing spaced repetition scheduling. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, page 4381–4390, New York, NY, USA. Association for Computing Machinery.

Sha Yuan, Hanyu Zhao, Shuai Zhao, Jiahong Leng, Yangxiao Liang, Xiaozhi Wang, Jifan Yu, Xin Lv, Zhou Shao, Jiaao He, Yankai Lin, Xu Han, Zhenghao Liu, Ning Ding, Yongming Rao, Yizhao Gao, Liang Zhang, Ming Ding, Cong Fang, Yisen Wang, Mingsheng Long, Jing Zhang, Yinpeng Dong, Tianyu Pang, Peng Cui, Lingxiao Huang, Zheng Liang, Huawei Shen, Hui Zhang, Quanshi Zhang, Qingxiu Dong, Zhixing Tan, Mingxuan Wang, Shuo Wang, Long Zhou, Haoran Li, Junwei Bao, Yingwei Pan, Weinan Zhang, Zhou Yu, Rui Yan, Chence Shi, Minghao Xu, Zuobai Zhang, Guoqiang Wang, Xiang Pan, Mengjie Li, Xiaoyu Chu, Zijun Yao, Fangwei Zhu, Shulin Cao, Weicheng Xue, Zixuan Ma, Zhengyan Zhang, Shengding Hu, Yujia Qin, Chaojun Xiao, Zheni Zeng, Ganqu Cui, Weize Chen, Weilin Zhao, Yuan Yao, Peng Li, Wenzhao Zheng, Wenliang Zhao, Ziyi Wang, Borui Zhang, Nanyi Fei, Anwen Hu, Zenan Ling, Haoyang Li, Boxi Cao, Xianpei Han, Weidong Zhan, Baobao Chang, Hao Sun, Jiawen Deng, Chujie Zheng, Juanzi Li, Lei Hou, Xigang Cao, Jidong Zhai, Zhiyuan Liu, Maosong Sun, Jiwen Lu, Zhiwu Lu, Qin Jin, Ruihua Song, Ji-Rong Wen, Zhouchen Lin, Liwei Wang, Hang Su, Jun Zhu, Zhifang Sui, Jiajun Zhang, Yang Liu, Xiaodong He, Minlie Huang, Jian Tang, and Jie Tang. 2022. A roadmap for big model. *arXiv preprint arXiv:2203.14101*.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. 2023. Memorybank: Enhancing large language models with long-term memory. *arXiv preprint arXiv:2305.10250*.

A Theoretical Basis

By our method, we focus on utilizing decoder-only transformer models, which represent the current mainstream LLM architecture. Transformer models have demonstrated remarkable success in various NLP tasks due to their ability to capture long-range dependencies and contextual information effectively. By employing decoder-only transformer models, we can leverage their autoregressive nature for language modeling and prediction tasks. The primary objective of pre-training in this context is the next token prediction task or auto-regressive language modeling. The model is trained to predict the next token in a given sequence of tokens, given the preceding context. This objective encourages the model to learn meaningful representations that capture the syntactic and semantic structures of natural language:

$$\max_{\theta} \sum_{i=1}^{N} \log P(u_i|u_{<i};\theta), \tag{6}$$

Given an unsupervised corpus of tokens u, where N is the size of the context window, and the conditional probability P is modeled using a neural network with parameters θ. These parameters are trained using stochastic gradient descent.

As new knowledge emerges and is acquired over time, it becomes necessary to update and train models to incorporate this new information. This process involves training the model on the larger dataset that includes the new knowledge. Adopting an iterative increase of data sets can lead to several problems that need to be addressed. 1.With each increase in the size of the data set, the time and cost required for pre-training also increase. 2.When new knowledge is added to the model, it often overlaps with existing knowledge. The earlier the time at which knowledge is introduced, the more times it is trained, resulting in an unbalanced distribution of training data. This can lead to biased models that favor certain types of information over others. 3.Repetitive training not only leads to unbalanced training data but also consumes unnecessary time and resources. In order to improve the efficiency of training time and optimize the process of acquiring new knowledge, we propose the utilization of Ebbinghaus memory theory for the segmentation and combination of new knowledge pre-training data sets. By applying the method of interval repetition, we aim to achieve a more efficient balance between training time cost and knowledge acquisition.

Pure forgetting curve is exponential in nature. Due to the fact that Ebbinghaus used the saving on relearning rather than recall, the curve can be best described by a fast-decay power function. Today, we know that forgetting is exponential in nature. It is very similar to radioactive decay. It is governed by similar principles.(Wozniak, 2017) The forgetting curve is expressed using an exponential decay model: $R = e^{-\frac{t}{S}}$, where R is the memory retention, or what fraction of the information can be retained. t is the time elapsed since learning the information. e is the base of the natural logarithm, approximately equal to 2.71828. S is the memory strength, which changes based on factors such as the depth of learning and the amount of repetition.

In our research, we have extended the DSR[1] algorithm based on the Ebbinghaus curve by replacing the time-based parameter with one based on the number of repetitions. This modification is rooted in the understanding that unlike human memory, which decays over time, the parameters of electronic storage do not necessarily decay over time. Instead, these parameters are subject to adjustments due to continuous training iterations, potentially resulting in overwriting or loss of previously formed memories. Consequently, it is essential to strengthen these memories in subsequent training. The new memory curve algorithm is described below[2]. Exponential memory curve model:

$$R = e^{\ln 0.9 \cdot \frac{N}{S}}, \tag{7}$$

- Initial stability: $S_0 = 0.5$
- Number of training iterations: $N >= 0$
- Memory stability parameter calculation: $S_L = S_0^{-f \cdot L}$
  - Count of memory: $L >= 0$
  - Attenuation coefficient: $f = 0.3$

Using the above algorithm, we draw a diagram of interval repetition(see Figure 3), and use the interval repetition data to slice the continuous training data set to achieve the balance between training requirements and training costs.

B Data Preparation

In this paper, we focus on enhancing the continuous knowledge acquisition performance of LLMs through continual pre-training. Training of general LLMs has already used a large amount of internet data, so to ensure the effectiveness of knowledge training, we collected a large amount of private

[1]The three component model of memory is referred to as the DSR model. *Memory stability (S)* is defined as the interrepetition interval that produces average recall probability of 0.9 at review time; *Memory retrievability (R)* is defined as the expected probability of recall at any time on the assumption of negatively exponential forgetting of homogenous learning material with the decay constant determined by memory stability (S); *Item difficulty (D)* is defined as the maximum possible increase in memory stability (S) at review mapped linearly into 0..1 interval with 0 standing for easiest possible items, and 1 standing for highest difficulty in consideration in SuperMemo (the cut off limit currently stands at stability increase 6x less than the maximum possible). (Wozniak, 2018)

[2]The complete DSR algorithm includes additional features such as *memory stability parameter S' update after memory success* and *difficulty parameter D' update after memory failure*. Our paper focuses on *the parameter variation of Stability (S)* during the memory process.

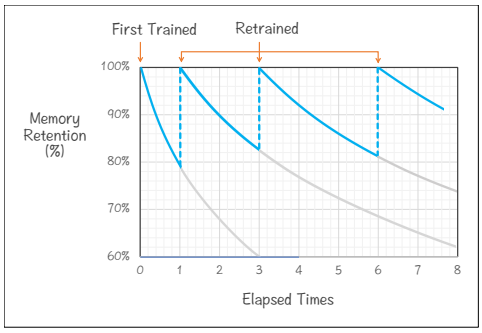


Figure 3: The interval repetition curve is a diagram that represents the number of repetitions in a discrete sequence, which is expressed more clearly using a continuous function fitting. This curve can be used as the basis for a section of the training dataset. As the number of iterations increases, the interval repeat also increases, providing valuable information for understanding the patterns and behavior of the discrete sequence.

data (IDC information). These data are not exposed on the internet, so training and testing based on this dataset can verify the results of continuous pre-training. Furthermore, in order to preserve the world knowledge and generalization ability of LLMs, we also incorporate general text data into the training data-set.

Private IDC Corpora We used private IDC information as the basis of the dataset. The IDC information includes: overview, address, equipment layout, environment requirement, air condition, and other information. Each data center serves as an independent dataset, and the original data consists of approximately 2,500,000 to 3,500,000 tokens.

Public Internet Corpora To construct the general corpora, we sample texts from internet public data-set: Skywork/Skypile-150B(Wei et al., 2023). This dataset is a high-quality dataset selected from Chinese web pages by using carefully filtered data processing. The size of this open-source dataset is approximately 600GB, with a total token count of around 150 billion.

SFT Data-set For support chat with private domian knowledge, we have added about 5% question-answer pairs related to data center knowledge based on OpenLMLab's moss-002-sft-data(OpenLMLab, 2023). The multi-turn conversational data-set is covering helpfulness, honesty, and harmlessness. The data-set is consisting of 570K English and 590K Chinese conversations generated by text-davinci-003.

Data Pre-processing To maintain the quality of pre-training private datasets, we have implemented a comprehensive pipeline that consists of data reformulation, filtering, and deduplication processes. This pipeline is designed to ensure that the data used for training is of the highest quality and reliability. In order to establish a standard for our private data center's data, we have used the GB50174-2017 Data Center Design Specification(MOHURD, 2017) as data reference. This specification provides guidelines for the proper handling and management of data in a data center environment, ensuring that the data is consistent and reliable.

Due to the relatively small size of the original private data, we use technical methods include the use of synonyms, paraphrasing, and cross-translation. By incorporating these techniques, we are able to increase the size of the dataset, making it more robust and better suited for pre-training. After processing the data through the aforementioned techniques, each dataset in the data center consists of approximately 10 million to 15 million tokens. This expanded dataset provides a more comprehensive and diverse set of data for training, resulting in more accurate and reliable models.

However, there still exists a significant difference in quantity between private domain-specific data and public datasets. Considering our objective of enhancing the model's performance in addressing domain-specific tasks without compromising its generalization capabilities through private domain-specific continual pre-training, it is crucial to maintain a balance in the token count between general and domain-specific data in the training samples. Unfortunately, due to hardware limitations, detailed experiments on the proportions of different types of data cannot be conducted. After initial explorations, a 10:1 ratio for the number of tokens between selected public and private domain-specific data was established.

Dataset Mixing Strategy During the continuous pre-training process, sliding window sampling data is used for publicly available internet data. Each time, 50M new data are taken from the original data-set as the next iteration's incremental data. Each time the training on public data is maintained around 500M. In the private IDC domain, data selection is carried out using a repeated interval method based on the Ebbinghaus forgetting curve(see Figure 4).

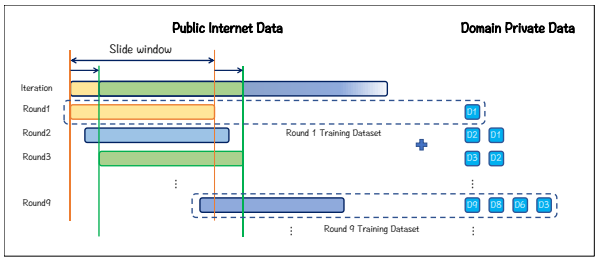


Figure 4: Each pre-training iteration mixes public internet data and private IDC data. Public data is selected using the sliding window technique, and private data is selected according to the algorithm of interval repetition. The purpose of this method is to ensure the generalization of the data, avoid over-fitting and save training cost.

One way improve the performance of LLMs is training by widespread data sources. This approach allows the models to learn from a diverse domain of data, which can enhance their generalization, reduced over-fitting, and improve representation learning. In this essay, we will discuss the method of mixing different data source data-sets and how it can improve LLM performance.

Nevertheless, we believe that appropriately incorporating the data into continual pre-training can further enhance the domain-specific performance of LLMs. To effectively transform the domain knowledge for model pre-training, we devise a data-set mixing strategy across various data sources.

Specifically, for each iteration of training, our data mixing strategy consists of the following steps to ensure the data is properly combined:

(1)In the first iteration, select 500M random data points from the public data-set and choose first private IDC data-set(50M). Combine the two data-sets to create the first pre-training iteration data-set.

(2)In each subsequent training iteration, use sliding window technique to randomly select 50M new public internet data-se, maintain a window size of 500M and a step size (new data) of 50M. This means that the previous iteration's 450M public data is retained and combined with 50M newly selected data, participate in this iteration pre-training as the publicly internet data. This step is to preserve the generalization ability of the large model and to avoid over-fitting the training data.

(3)For private training data-set, use a repeated interval algorithm to select n participating data center data-sets based on the selection rule and data center ID number. This step is to ensure that the new knowledge is trained into the large model through spaced iterations, avoiding over-fitting and duplication cost.

(4)Merge the selected public internet data-set($\sim$500M) and private data center data-set ($n\times\sim$50M) to create the pre-training data-set for the current iteration.

An example is illustrated in Table 4.

	Public Data	Private Domain Data				
Round	**500M**	**Each IDC has $\sim$50M data**				**Total**
Rd-1	SW_Data_1	IDC1_Data				$\sim$550M
Rd-2	SW_Data_2	IDC2_Data	IDC1_Data			$\sim$600M
Rd-3	SW_Data_3	IDC3_Data	IDC2_Data			$\sim$600M
Rd-4	SW_Data_4	IDC4_Data	IDC3_Data	IDC1_Data		$\sim$650M
Rd-5	SW_Data_5	IDC5_Data	IDC4_Data	IDC2_Data		$\sim$650M
Rd-6	SW_Data_6	IDC6_Data	IDC5_Data	IDC3_Data		$\sim$650M
Rd-7	SW_Data_7	IDC7_Data	IDC6_Data	IDC4_Data	IDC1_Data	$\sim$700M
Rd-8	SW_Data_8	IDC8_Data	IDC7_Data	IDC5_Data	IDC2_Data	$\sim$700M
Rd-9	SW_Data_9	IDC9_Data	IDC8_Data	IDC6_Data	IDC3_Data	$\sim$700M
Rd-10	SW_Data_10	IDC10_Data	IDC9_Data	IDC7_Data	IDC4_Data	$\sim$700M
Rd-11	SW_Data_11	IDC11_Data	IDC10_Data	IDC8_Data	IDC5_Data	$\sim$700M
Rd-12	SW_Data_12	IDC12_Data	IDC11_Data	IDC9_Data	IDC6_Data	$\sim$700M

Table 4: The pre-training data size. The actual amount of data per IDC is approximately 50M($\pm$10%), and this is only an approximation value.

C Continuous Pre-training

We select Baicuan 2 (Yang et al., 2023) as the backbone models of continuous pre-training due to the following reasons: (1) Baicuan 2 is a series of llama-like LLM, which is open source large model. The ecosystem of the Llama-like model is relatively comprehensive, and there are abundant tools and document to support the training process. (2) Biachuan 2 provide base model, has not undergone any SFT, ensuring the reliability of evaluating the performance changes on our benchmarks after continual pre-training. (3) Chinese support, our private IDC data-sets is in Chinese, so training on Chinese large model will have better performance. Specifically, We conduct our experiments on Biachuan 2 with 13B parameters.

Limited by the hardware quotas, each of our experiments is conducted on 8 $\times$ NVIDIA RTX 4090 24GB GPUs, all in one server. Although there is a lending bottleneck, 4090 is often used in the reasoning process of large business models. The use of 4090 is more in line with the computing power scenario of enterprises in the process of large-scale model operation.

We utilize Huggingface's Transformers (Wolf et al., 2020) and DeepSpeed (Rasley et al., 2020) frameworks. Hugging Face's Transformers is a popular open-source library that provides a wide range

of pre-trained models for natural language processing (NLP) tasks. DeepSpeed is a deep learning optimization library developed by Microsoft. It is designed to improve the training efficiency and scale of large NLP models. Transformers and DeepSpeed are two essential tools for training large-scale machine learning models. Transformers provide a model implementation and fundamental training workflow, while DeepSpeed offers ZeRO (Zero Redundancy Optimizer) optimizer (Rajbhandari et al., 2020) for improving GPU memory consumption. ZeRO is a memory optimization technique developed by NVIDIA to address the increasing demand for larger and more complex deep learning models. We use ZeRO Stage 3 offload memory usage to reduce consumption during model training.

Item	Model Hyper-parameters
Backbone	Baichuan 2-13B
Parameters	130,1544,9600
Global Batch Size	4096
Sequence Length	4096
Number of Attention Heads	32
Number of Hidden Layers	32
Intermediate Size	11008
Learning Rate	2e-4
Gradient Accumulation Steps	16

Table 5: Hyper-parameters in our experiments.

To enhance the training efficiency and avoid numerical underflow or overflow problems, the training process is performed in bfloat16 mixed precision(Mishra et al., 2022). In the experiments conducted, the hyper-parameters utilized are presented in Table 5. Some hyper-parameters were set based on the original pre-training settings of Biachuan 2. Specifically, during the training process, it is not only necessary to train the attention layers (w_pack), but also the mlp layers, because the fully connected layers have higher number of parameters implies.These layers can hold and process a greater amount of information, which can be crucial for the performance of certain tasks.

In the process of model continual pre-training, it is essential to monitor development through various stages of training. This involves a detailed assessment of multiple model checkpoints, which are saved states of the model at different intervals during the continual pre-training phase. When examining Figure 5(a), certain key steps might stand out where significant changes in the loss curve occur.

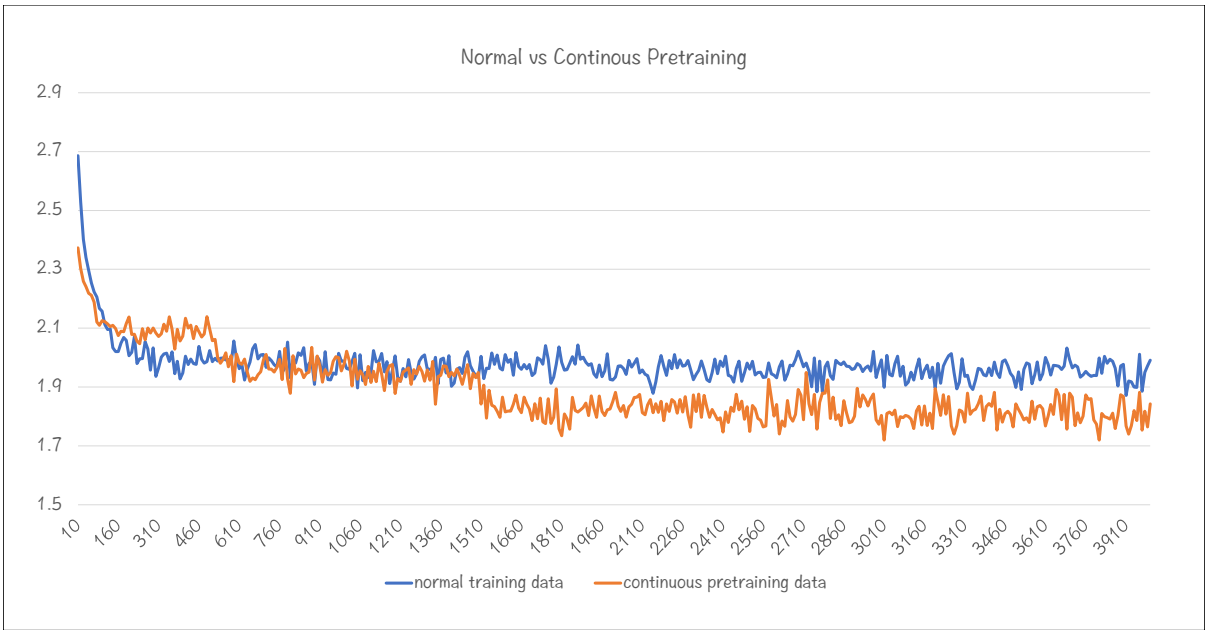
In this study, the pre-training process involved a comparison between standard pre-training and continuous pre-training. First, we combined all the private domain data-sets used in the training into a comprehensive pre-training data-set and conducted the pre-training according to the standard process. The standard pre-training process involved training for a total of 2 epochs, with 4000 steps and logged every 10 steps. Then the data-set of the data center is segmented, and the interval repetition algorithm is used to add new data center data in each round of iteration. 500 steps are trained as one iteration, and a total of 8 iterations are performed. The loss function curve is obtained for analysis (Figure 5(a)).

Here, the curve shows the continuous pre-training loss trend, which involves splicing multiple rounds of iterative training data together, resulting in a loss curve that demonstrates a downward trend with an increase in iteration. This approach contrasts with standard pre-training, the loss function curve obtained through continuous pre-training converges faster compared to the standard pre-training loss curve.

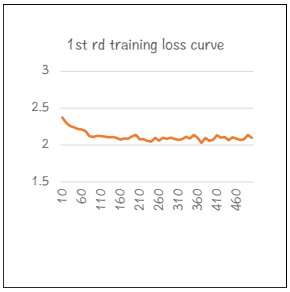
Additional, during each round of iterative training, we train for 500 extra steps (without adding new data), and compare the loss curve obtained with the loss curve of the next iteration when new data is added. See Figure5(b) to Figure5(i). These curves indicate that the addition of new data will affect the loss curve. As the number of iterations increases, this impact gradually diminishes, proving that the model has learned new knowledge. More accurate training results are shown through test data.

D Evaluation

For autoregressive LLM, each task is considered as a text generation task. Therefore, we can use automatic evaluation metrics for text generation to assess the model's performance on different tasks. In order to more accurately evaluate the performance of private domain model after continuous training, we conducted tests based on evaluation research related to large models(Chang et al., 2023). We create a benchmark test data-set based on private domain training data and tested the model from two aspects: automated evaluation and manual evaluation. For the interference of random seed (Random seed) in the model output process during the training, we adopted a 3-round zero-shot method for testing and recorded the highest score.



(a) Normal vs Continous Pretraining



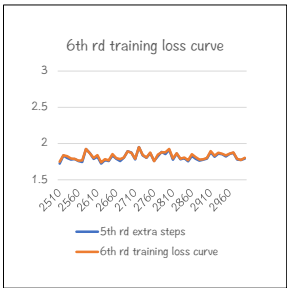
(b)

(c)

(d)

(e)

(f)



(g)

(h)

(i)

Figure 5: Select the loss function curve of the first 8 iterations, Figure 5(a) shown the difference between ordinary pre-training and continuous pre-training on the loss curve. The orange line represents the loss curve of continuous pre-training, is composed of the loss curve stitched together through 8 iterations. Figure5(b) to Figure5(i) shown each iteration loss curve. The blue curve's data from additional 500 steps of previous iteration without adding any new IDC data.

Evaluation Methods Model automatic evaluation method is based on computer algorithms and automatically generated metrics, which can quickly and efficiently evaluate the performance of models. On the other hand, manual evaluation focuses on the subjective judgment and quality evaluation of human experts, providing a more in-depth and detailed analysis and opinions.

- **Model Automatic Evaluation Method:**

This method is based on computer algorithms and automatically generated metrics. It allows for a rapid and efficient assessment of a model's performance. The use of computational algorithms

enables objective measurements and comparisons, contributing to the scalability and speed of the evaluation process. Additionally, this approach can provide standardized and consistent evaluations across various models.

Here we use the cosine similarity method to compare the output of the large model with the standard answer, as the accuracy of the output of the large model. Cosine similarity is a measure of the cosine value of the angle between two vectors. It is commonly used to determine how similar two strings or documents are by representing them as word vectors or text vectors and then calculating the cosine similarity between them. A higher cosine similarity indicates a higher degree of similarity between the two strings. We used the text2vec(Selivanov, 1996) to vectorize strings.

In mathematical terms, the cosine similarity between two vectors A and B is calculated using the following formula:

$$cosine \quad similarity = \frac{A \cdot B}{\parallel A \parallel \parallel B \parallel} \tag{8}$$

Where $A \cdot B$ represents the dot product of the two vectors, and $\parallel A \parallel \parallel B \parallel$ represent the magnitudes of the vectors A and B, respectively.

- **Model Human Evaluation Method:**

Manual Evaluation is a way to evaluate the quality and accuracy of model-generated results through human participation. Manual evaluation relies on the subjective judgment and expertise of human evaluators. It emphasizes in-depth analysis and detailed assessments, leveraging human insights and domain knowledge to provide nuanced perspectives on model performance. This approach can offer qualitative feedback and identify subtle nuances that may not be captured by automated methods.

We compare and score the outputs of large models and benchmark test data with domain experts, and record the scores to evaluate the accuracy of the large models.

Evaluation Data-sets We use human experts to generate seed evaluation datasets, supplemented by automated expansion to generate complete evaluation datasets. Humans can create many unique evaluation data for assessment, which better reflects the performance of continuously training large models. Evaluation data-set play a crucial role in assessing the performance of continuous training model. Our method generate an evaluation data set for a single data center by combining expert-curated assessment data sets(Nguyen et al., 2022) and user feedback during private domain models during the operational process. Then, expand the test data-set of one data center to each participating training data center, forming a comprehensive evaluation data-set.

The accuracy of numerical values is extremely important for private domain large models. Parts of the data-set must be set to address numerical issues. The standard for judging is that if the numerical part is incorrect or exceeds the error range, it will be directly rated as 0 points.

In addition to the questions related to private domain knowledge, we have also set up a group of questions that are unrelated to any private domain knowledge. These questions serve as a control group, used to validate the capabilities of the existing large models during the continuous pre-training process to ensure that they do not degrade.

Model Output The Table 2 shows the automated and manual test results of the knowledge output by the large model from different data centers at different iterative stages of continuous training. From the test results, the accuracy of the trained private domain knowledge is significantly higher than that of the untrained model. With the increase of iterative rounds, the accuracy gradually improves. The knowledge accuracy of the data center and the control group are very close after 3 interval repetitions. The knowledge accuracy of individual data centers is higher than that of the control group, which may be caused by the amount and quality of training data, and further research is needed. The accuracy of the control group fluctuates less, indicating that the continuous pre-training of the model has little effect on the original model.