Data Shunt: Collaboration of Small and Large Models for Lower Costs and Better Performance

Dong Chen[1], **Yueting Zhuang**[1]*, **Shuo Zhang**[2], **Jinfeng Liu**[3], **Su Dong**[3], **Siliang Tang**[1]

[1]College of Computer Science and Technology, Zhejiang University
[2]School of Software Technology, Zhejiang University
[3]Ant Group
{chendongcs, yzhuang, shuo.zhang, siliang}@zju.edu.cn, jinfeng.ljf@antgroup.com, listenersu@gmail.com

Abstract

Pretrained large models, particularly large language models, have garnered increasing attention, as they have demonstrated remarkable abilities through contextual learning. Pretrained large models are increasingly recognized as fundamental tools for solving various tasks. However, the substantial computational demands of large models have dissuaded most product teams and individuals from running them. In such scenarios, to leverage the exceptional performance of large models, one must solely depend on costly APIs, further burdening product teams and individuals. On the other hand, despite the overall inferior performance of small models compared to large models, there are certain distributions where small models can achieve comparable or even superior results. For instance, during training, small models may become trapped in a local optimum that is unique to certain distributions, leading to superior performance. Hence, we propose Data Shunt (DS), a general paradigm for collaboration of small and large models. DS not only substantially reduces the cost associated with deploying large models but also effectively enhances overall performance. Specifically, DS determines the shunting direction by evaluating the confidence level of small models. When the confidence level falls below a specific threshold, the input data is forwarded to large models. To further leverage the advantages of the small and large models, we introduce Prompt Pruning (PP) and 2-Stage Confidence Distillation (2CD), which facilitate mutual collaboration, leading to better results and less cost. The remarkable performance across diverse modalities and tasks demonstrates the superiority of the proposed DS over large models. For instance, ChatGPT achieves an accuracy of 94.43% on Amazon Product sentiment analysis, and DS achieves an accuracy of 95.64%, while the cost has been reduced to only 31.18%. The code for the proposed method are provided for research purposes https://github.com/Anfeather/Data-Shunt.

Introduction

Recent years have seen a surge of interest in pretrained large models (Yu et al. 2023; Li et al. 2023a), which are trained on a vast quantity of data at scale and can be adapted to a wide range of downstream tasks (Bommasani et al. 2021). Large language models (LLM), such as GPT (Brown et al. 2020;

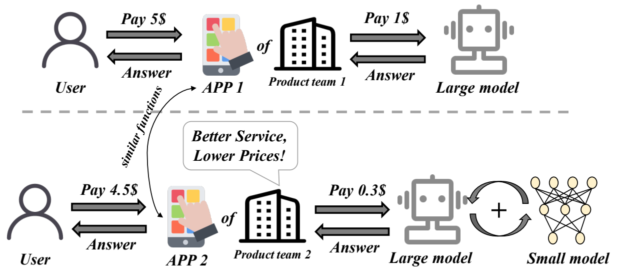

Figure 1: Commercial Applications of Large Models. *upper:* Product team 1 only use large models to support their applications. *lower:* Product team 2 decreases costs with collaboration of large and small models, enabling them to offer more attractive prices to users.

Ouyang et al. 2022; Liu et al. 2023) have demonstrated outstanding performance in text-related tasks (Wei et al. 2022; Brown et al. 2020). Additionally, multimodal large models (Zhu et al. 2023) like Flamingo (Alayrac et al. 2022) and BLIP-2 (Li et al. 2023b) have been developed to extend the capabilities of LLM to encompass vision modality. Pretrained large models, particularly ChatGPT, have found widespread applications in various domains, including coding (Surameery and Shakor 2023), education (Biswas 2023), health (Biswas 2023), and beyond, revolutionizing people's lives.

Despite the impressive performance of pretrained large models such as ChatGPT across various applications, their computational demands make them impractical for deployment on numerous devices. As a result, product teams or individuals might opt to acquire the interface of the pretrained large model to access the associated services. Nonetheless, frequent invocations of the interface prove to be prohibitively costly for both teams and individuals. Besides, we find that although the small model's overall performance is significantly lower than that of the large model, it can still outperform or achieve highly competitive results across certain data distributions. We first divide samples into easy samples and hard samples. Easy samples represent data that small models can fit well. These samples generally represent the majority of the training data and are relatively easier for small models to learn and predict accurately. In con-

*Yueting Zhuang is the corresponding author.

trast, hard samples refer to data that poses challenges for small models, including samples that deviate from the main distributions of the training data and samples located at the boundaries between different categories. For example, for long-tail data with few majority classes (head) and large amount of minority classes (tail) (Ouyang et al. 2016; Zhang et al. 2017; Oksuz et al. 2020), small model can perform much better on head data, thus, head data is easy samples, while tail data is hard samples. Furthermore, the small model often succumbs to overfitting when trained on a limited dataset. During inferencing, the small model demonstrates good performance when the inputs resemble the training data. However, it exhibits poor performance when the test data significantly deviates from the training data (Horenko 2020). In such scenarios, data that conforms to the training distribution is considered as easy samples, while data that deviates from the training distribution is considered as hard samples. For pretrained large models, there is no distinction between easy and hard samples because these models have already been exposed to massive amounts of training data. The distribution of unknown samples is unlikely to deviate significantly from the knowledge acquired by the large models. This is also why pretrained large models tend to perform better in real-world applications.

In this work, we mainly focus on how to call large models as few as possible with the help of small models, while achieve better performance. As illustrated in Figure 1, APP 1 and APP 2 have similar functions, and the upper figure depicts APP 1 only uses large models, incurring additional costs compared to APP 2 that involves collaboration between large and small models. By reducing costs, APP 2 can offer more appealing prices, thereby enhancing market competitiveness. We propose Data Shunt (DS), which utilizes the confidence of small models to determine the appropriate processing direction for the input data: either through large models or exclusively through small models. DS represents a collaborative paradigm of large models and small models. We first show Prompt Pruning (PP), a method that leverages small models to assist large models in refining their prediction space through prompt design. Specifically, in a classification task, as a sample transitions from the small model to the large model, the probability of it belonging to distributions in which the small model demonstrates proficiency diminishes. Thus, this probability can be integrated into the prompt to enhance the large model's discrimination against alternative distributions. On the other hand, large models, equipped with extensive general knowledge, can distill unfamiliar knowledge to improve small models and further reduce the reliance on large models. However, the distillation process often leads to catastrophic forgetting since small models are initially proficient only in a limited number of distributions. To address this issue, we introduce 2-Stage Confidence Distillation (2CD), a method where small models learn iteratively from high-confidence samples provided by large models and their original version.

The effectiveness of the proposed method is validated across various modalities and tasks. Specifically, in the sentiment analysis task, DS enhances the overall accuracy of ChatGPT by 1.21%, while reducing the cost of the large model to 31.18% of its original expense. In the image classification task, DS improves the overall accuracy by 5.07%, and the cost of the large model decreases to 66.10%. Additionally, for the image caption task, DS elevates the average BLEU score by 0.42, with the cost of the large model amounting to 65.36%.

The main contributions of this paper can be summarized as follows:

- We introduce a collaborative paradigm of large and small models to enhance performance while minimizing costs, which is simple yet effective.
- Building upon the strengths of both small and large models, we introduce two novel methods, Prompt Pruning (PP) and 2-Stage Confidence Distillation (2CD), which further enhance performance while mitigating costs.
- We demonstrate the efficacy of the proposed method across diverse multimodalities and tasks.

Related Work

Large Model and Small Model

Deploying large models with remarkable few-shot capabilities (Smith et al. 2022; Zhang et al. 2022; Hoffmann et al. 2022; Li et al. 2023c) poses a challenge in real-world applications primarily due to their enormous size. For example, running a 175 billion LLM requires at least 350GB GPU memory (Zheng et al. 2022), which are far beyond affordable for most product teams, let alone more large models over 500B parameters (Chowdhery et al. 2022). Moreover, the costly interface also poses challenges in addressing real-world issues using large models, as its expensive cost exceeding the affordability of most product teams. Consequently, we propose to mitigate the aforementioned issues by employing smaller, specialized models. In addition, there are also studies focus on large and small models.

Reducing Cost and Improving Performance

Prior works discuss three main strategies for cost reduction: prompt adaptation, LLM approximation, and LLM cascade (Chen, Zaharia, and Zou 2023). The prompt adaptation try to make the prompt shorter. LLM approximation explores how to create simpler and cheaper LLMs on specific tasks. LLM cascade aims to adaptively choose different APIs for different queries. Different from prior works that only focus on language modality, we explores a generalized paradigm that can be applied to various modalities and tasks. Besides, we reduce large model cost by combining task-specific small models, which is simple yet effective.

Methodology

The proposed Data Shunt (DS) is a collaborative paradigm of large and small models. In the subsequent sections, we will explore how small models contribute to the improvement of large models, how large models benefit small models, as well as provide an overview of the entire DS paradigm.

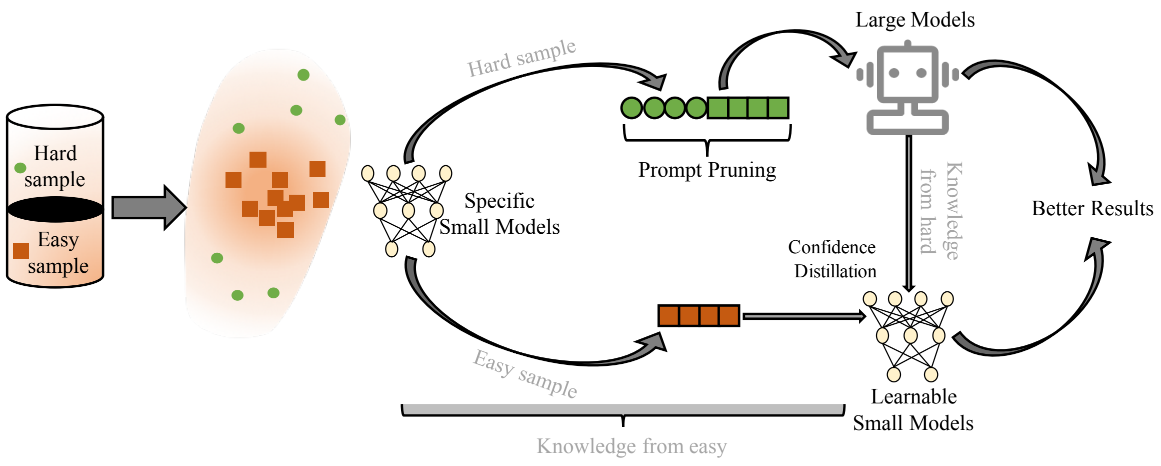

Figure 2: The training process of the proposed method. Hard samples refer to data that poses challenges for small models, including samples that deviate from the main distributions of the training data and samples located at the boundaries between different categories. In contrast, easy samples represent data that small models can fit well. These samples generally represent the majority of the training data and are relatively easier for small models to learn and predict accurately.

Small Models for Large Models, Prompt Pruning

In this section, we focus on how small models help large models. Small models can exhibit superior performance when dealing with specific data distributions, attributed to the training data distribution and their own architectural characteristics. To improve the performance of large models with advantages of small models, we propose Prompt Pruning (PP) for classification task. PP refines the prediction space of large models using the prompts crafted with predictions of small models.

As demonstrated in Equation 1, we obtain the prediction confidence C_s by subjecting the output of a trained small model F_{small} to a softmax operation for a given input x. A higher value in a specific dimension of C_s indicates a greater level of confidence from the small model regarding the corresponding judgment.

$$C_s = \frac{e^{z_i}}{\sum e^{z_d}}, \quad z_i \in F_s(x) \tag{1}$$

Intuitively, small models excel at discriminating specific distributions and determining whether an input aligns with those distributions. For instance, a small model may excel in distinguishing cats from other animals. Although it cannot recognize dogs, tigers, and other such animals, it can confidently determine that they are not cats (i.e., the corresponding confidence is lower). Therefore, incorporating these predictions into the prompts for large models allows us to refine the prediction space and enhance the performance. For example, a prompt of PP for image classification task:

"This is a photo of a label with probability C_s."

Compared to traditional prompt, PP introduces the confidence of small models as prior knowledge. When the input data do not follow distributions that small models excel in, C_s of these distributions (or called classes) will be lower. If there are numerous candidate classes, we only add probability to classes small models excel in. Thus, large models can ignore these classes and improve accuracy. We call such prompt that with small model confidence as soft prompt. On the other hand, we can directly remove classes small models excel in. Thus, the prediction space is smaller, and the possibility for large models to predict the correct class will be higher. We call such prompt that directly remove the candidates as hard prompt.

We further perform theoretical analysis from an entropy perspective to show that PP with soft and hard prompt can effectively improve the performance of large models.

Let X and Y be the variable of input data and small model prediction, respectively. We use entropy to quantify the lower bound of model capability, where higher entropy indicates that the model struggles to produce effective results. $H(X)$ is the entropy of the input data and $H(Y)$ is the entropy of the prediction. We first show the effectiveness of soft prompt. We use $H(X \mid Y)$ to represent the entropy of the input data with soft prompt $\hat{X} = X \mid Y$. Now we show that with Y, $H(\hat{X})$ is lower than $H(X)$:

$$\begin{aligned} H(X) - H(\hat{X}) &= \sum_{x\in X}\sum_{y\in Y} p(x,y)\log_2 \frac{p(x,y)}{p(x)p(y)} \\ &\geq \left[\sum_{x\in X}\sum_{y\in Y} p(x,y)\right]\log_2 \frac{\sum_{x\in X}\sum_{y\in Y} p(x,y)}{\sum_{x\in X}\sum_{y\in Y} p(x)p(y)} = 0 \end{aligned} \tag{2}$$

Note that the above inequality takes the equal sign iff X and Y are independent, i.e., $p(x,y) = p(x)p(y)$. However, small models often exhibit a strong correlation between their outputs and inputs. Thus, we get $H(X) > H(\hat{X})$, which validates that the input data with soft prompt gets lower entropy.

As for the hard prompt, let entropy of the prediction of large models be $H(C_l) = -\sum_{i=1}^{N} c_i \log c_i$, where N is the number of candidates, $\sum_{i=1}^{N} c_i - 1 = 0$ and $c_i \in C_l$. With

Lagrange multiplier method, we get:

$$G\left(c_1, c_{2 \ldots} c_N, \lambda\right) = -\sum_{i=1}^{N} c_i \log c_i + \lambda\left(\sum_{i=1}^{N} c_i - 1\right) \quad (3)$$

then partially differentiating G in Equation 3 with respect to c_i and λ,

$$\frac{\partial G}{\partial c_i} = -\log c_i - 1 + \lambda, \qquad \frac{\partial G}{\partial \lambda} = \sum_{i=1}^{N} c_i - 1 \quad (4)$$

Let Equation 4 be 0, we can get $c_1 = c_2 = \ldots = c_N = \frac{1}{N}$, $H(C_l) = \log N$, which is the maximum value of $H(C_l)$. If we perform PP with hard prompt, the number of candidates will be M, and $M < N$. Thus, the maximum value of $H(C_l)$ will be $\log M$. As $\log M < \log N$, the maximum value of $H(C_l)$ will be smaller, and the lower bound of large models will be higher.

Large Models for Small Models, Confidence Distillation

In addition to small models assisting large models, we can also employ large models to support small models, enabling the distillation of knowledge that small models lack. As the knowledge of small models expands, they can handle a greater number of samples, thereby further reducing the need to invoke large models. However, we observed that small models tend to forget the original well-fitting distributions after knowledge distillation (French 1999; Gou et al. 2021). Besides, large models may severely degrade small models if the performance of large models is not well (i.e., distill incorrect knowledge). In order to address this issue, we propose 2-Stage Confidence Distillation (2CD), which performs knowledge distillation based on the confidence levels of both small and large models. Specifically, to preserve the advantages of small models, we maintain a version of the original small models that do not receive knowledge from large models (referred to as specific small models). Besides, we duplicate the small models to enhance knowledge acquisition (referred to as learnable small models). In order to mitigate the negative impact of incorrect knowledge on the small models, we enable learnable small models to learn from both specific small models and large models simultaneously, based on their respective confidences. For a input sample, when the confidence of specific small models is low and the confidence of large models is high, learnable small models will acquire predictions from the large models to introduce additional knowledge and promote cost reduction. Such process enables learnable small models to handle increasingly diverse samples. Conversely, learnable small models will continue to learn high confidence samples from specific small models to mitigate the impact of distorted knowledge from large models.

For an input data x, if C_{s1} (computed by Equation 1) is lower than shunt threshold, δ, we compute the prediction of large models by

$$C_l = \frac{e^{z_i}}{\sum e^{z_d}}, \quad z_i \in F_l(x) \quad (5)$$

where F_l is the function of large models and C_l is the predicted confidence.

If $C_l > \delta$, we perform knowledge distillation with Kullback Leibler divergence (Hershey and Olsen 2007) by

$$L_{ls} = KL(F_{s2}(x), C_l) \quad (6)$$

To alleviate the impact of distorted knowledge from large models, we also select samples that $C_{s1} > \delta$ to perform knowledge distillation, where

$$L_{s1s2} = KL(F_{s2}(x), C_{s1}) \quad (7)$$

During 2CD, we perform Equation 6 and 7 iteratively.

Data Shunt

We propose a collaborative paradigm, DS, which aims to mitigate the substantial cost associated with large models while simultaneously enhancing performance. For training process, as illustrated in Figure 2, DS tries to achieve a coordinated state between small and large models. On one hand, it performs knowledge distillation for the learnable small models, transforming more data into easy samples that the learnable small models can handle. On the other hand, DS determines the shunt threshold δ by evaluating the specific small models' confidence with training set, thus allowing more easy samples to be processed by the small models. During inference, an input will be processed by specific small models and learnable small models at first. If the confidence of small models is lower than δ, this sample and the corresponding confidence will be sent to large models, where the prompt will be re-designed with the confidence of small models. On the other hand, if the confidence of the input sample is high, this sample will only be processed by small models.

Experiments

In our experiments, we aim to (1) validate that DS can enhance the overall performance, while reducing the cost across various modalities and tasks, (2) validate the effectiveness of PP and 2CD, respectively, (3) analyze the important hyperparameter, shunt threshold, In our experiments, large models and small models are relative. For example, model A is small compared to model B, but large compared to C. Moreover, the overall performance of large models is always better than that of small models. To save on experimental costs, we utilize the paid large model, ChatGPT, in the first experiment, while using relatively smaller free pretrained models as substitutes in the remaining experiments. Therefore, we use the query proportion of large models as an evaluation metric instead of cost. All experiments run on a 2080 Ti.

Data Shunt for Language Modality

ChatGPT is one of the most influential large language models today. Running ChatGPT requires at least 350GB GPU memory with specialized infrastructure (Zheng et al. 2022), which is far beyond affordable for most product teams. Consequently, product teams must invoke the ChatGPT interface to accomplish their desired functionalities. Unfortunately,

Category	Small 1	Large	DS	Category	Small 1	Large	DS
Games	84.34%	96.22%	96.13%|88.88%	Clothing	85.34%	96.89%	94.28%|84.61%
Kindle	89.05%	95.65%	95.83%|75.88%	Beauty	85.37%	97.20%	94.33%|86.99%
Baby	88.63%	96.41%	95.93%|88.99%	Video	85.37%	92.54%	94.32%|87.28%
Movies	85.37%	93.42%	94.23%|87.68%	Lawn	85.36%	89.36%	94.32%|94.47%
Electronics	85.24%	95.41%	94.67%|88.44%	Home	85.39%	96.28%	94.39%|88.10%
Office	85.23%	95.45%	94.68%|92.12%	Toys	85.41%	96.74%	94.40%|87.80%
CDs	84.68%	95.87%	94.86%|91.99%	Grocery	85.43%	96.73%	94.42%|89.80%
Books	85.26%	93.66%	94.20%|81.88%	Automotive	85.42%	94.69%	94.42%|90.34%
Sports	85.26%	95.06%	94.21%|89.00%	Tools	85.41%	94.49%	94.43%|90.58%
Health	85.24%	95.04%	94.23%|89.49%	Pet Supplies	85.40%	94.03%	94.42%|90.84%
Overall	Small 1: 85.40%, Large: 94.43%, DS: 94.42%			Query	Small 1: 0%, Large: 100%, DS: 84.97%		
Category	Small 2	Large	DS	Category	Small 2	Large	DS
Games	85.29%	96.22%	96.13%|84.01%	Clothing	86.13%	96.89%	94.31%|74.78%
Kindle	89.74%	95.65%	95.85%|71.73%	Beauty	86.17%	97.20%	94.36%|81.93%
Baby	89.33%	96.41%	95.95%|82.56%	Video	86.18%	92.54%	94.35%|78.15%
Movies	86.27%	93.42%	94.25%|80.82%	Lawn	86.17%	89.36%	94.35%|90.64%
Electronics	86.28%	95.41%	94.69%|83.57%	Home	86.21%	96.28%	94.41%|81.93%
Office	86.26%	95.45%	94.69%|89.14%	Toys	86.22%	96.74%	94.43%|81.05%
CDs	85.70%	95.87%	94.88%|87.78%	Grocery	86.24%	96.73%	94.45%|84.16%
Books	86.05%	93.66%	94.23%|77.59%	Automotive	86.24%	94.69%	94.45%|87.44%
Sports	86.05%	95.06%	94.24%|82.66%	Tools	86.23%	94.49%	94.45%|86.58%
Health	86.03%	95.04%	94.26%|85.12%	Pet Supplies	86.21%	94.03%	94.45%|86.26%
Overall	Smal 2: 86.21%, Large: 94.43%, DS: 94.44%			Query	Small 2: 0%, Large: 100%, DS: 80.00%		
Category	Small 3	Large	DS	Category	Small 3	Large	DS
Games	90.39%	96.22%	96.15%|36.25%	Clothing	95.63%	96.89%	97.45%|31.10%
Kindle	95.89%	95.65%	97.38%|20.67%	Beauty	92.90%	97.20%	97.24%|30.39%
Baby	92.81%	96.41%	96.26%|32.90%	Video	92.67%	92.54%	96.27%|25.32%
Movies	90.57%	93.42%	94.86%|31.76%	Lawn	84.26%	89.36%	90.63%|48.93%
Electronics	91.76%	95.41%	96.11%|39.56%	Home	93.12%	96.28%	96.73%|33.89%
Office	90.72%	95.45%	95.10%|44.31%	Toys	92.22%	96.74%	96.38%|31.34%
CDs	88.57%	95.87%	95.50%|36.92%	Grocery	92.50%	96.73%	96.66%|32.28%
Books	91.98%	93.66%	95.37%|27.91%	Automotive	91.30%	94.69%	94.69%|33.33%
Sports	93.11%	95.06%	96.02%|34.83%	Tools	91.62%	94.49%	95.38%|37.87%
Health	91.73%	95.04%	95.71%|34.35%	Pet Supplies	91.02%	94.03%	95.02%|38.96%
Overall	Small 3: 91.79%, Large: 94.43%, DS: **95.64**%			Query	Small 3: 0%, Large: 100%, DS: **31.18**%		

Table 1: Different small models for DS on sentiment analysis. Small model 1 represents TextCNN, small model 2 represents LSTM, and small model 3 represents fine-tuned BERT. The large mode is ChatGPT. For DS we present the accuracy and query proportion (i.e., sample proportion processed by ChatGPT) in the same unit.

the exorbitant costs associated with the interface substantially diminish the revenue generated by product teams.

In this section, we show that the proposed method can significantly reduce the cost of calling large model while achieving better overall performance. We conduct sentiment analysis on Amazon Product Data (He and McAuley 2016; McAuley et al. 2015), where there are 20 categories of product comments along with corresponding positive or negative sentiment labels. In addition, we maintain a balance between positive and negative samples, and divide the dataset into training set, validation set, and testing set, with 2,504,958, 277,508, and 309,186 samples respectively.

Related results are presented in Table 1. For DS, the right value of "|" is the sample proportion processed by ChatGPT, while the left value of "|" is the accuracy of DS. In this experiment, we use TextCNN (Kim 2014), LSTM (Wang et al. 2016) and fine-tuned BERT (Devlin et al. 2018) as the small model respectively, while ChatGPT serves as the large model. It can be seen that the overall accuracy of TextCNN, LSTM, fine-tuned BERT and ChatGPT is 85.40%, 86.21%, 91.79% and 94.43%. The pretrained large model, ChatGPT, significantly outperforms specific small models, TextCNN, LSTM, and fine-tuned BERT. Its extensive pretraining equips it with a wealth of knowledge, enabling it to handle a broader range of scenarios. On the other hand, TextCNN, LSTM and fine-tuned BERT see much data that belongs to the 20 classes, thus, small models may fit some data better. For instance, in the Kindle, certain e-books share similar highlights, such as well-developed characterization and compelling storylines. Thus, during inference, if there is a comment that is similar to one of the training dataset, small models will give a more accurate predic-

	Small	Large	DS
Head	70.25%	60.00%	**71.99%**
Med	46.61%	57.28%	**59.91%**
Tail	29.28%	57.19%	**57.61%**
Overall Accuracy	48.84%	58.18%	**63.25%**
Query Proportion	**0**%	100%	66.10%

Table 2: DS for image classification on CIFAR-100-LT.

	Small	Large	DS
BLEU-1	72.92	73.27	**74.95**
BLEU-2	55.73	60.04	**60.43**
BLEU-3	41.20	**46.99**	46.85
BLEU-4	30.28	**36.11**	35.82
Mean	50.03	54.10	**54.52**
Query Proportion	**0**%	100%	65.36%

Table 3: DS for image caption on Microsoft COCO.

tion. Based on this idea, we have observed that the combination of small and large models can yield competitive or even superior performance. As shown in DS of Table 1, we present results of DS combined with three small models. To achieve competitive results to ChatGPT, DS with TextCNN needs to send 84.97% of the data to ChatGPT, while DS with LSTM needs to send 80.00%, and DS with fine-tuned BERT needs to send 31.18%. It is evident that as the performance of the small model improves, the required query proportion to attain competitive performance decreases. Furthermore, DS with fine-tuned BERT exhibits remarkable results, benefiting from its exposure to lots of diverse data as well as task-specific data. Fine-tuned BERT even slightly performs better on two fine-grained classes, Kindle and Video, where the query proportion of DS is lower than 26%. It demonstrates that DS with small foundation models can achieve much better performance. Compared to ChatGPT, DS displays a 1.21% increase in accuracy while only 31.18% of the data is calculated by the large model.

Data Shunt for Vision Modality

In this section, to better present the application of DS, we conduct experiments for long-tailed image classification. Long-tailed image classification is a common challenge in practical computer vision applications (Zhou, Hu, and Wang 2018). We follow (Li et al. 2022) to separate CIFAR-100 into the head, medium and tail regions based on different numbers of samples.

Due to budget constraints and the limited availability of vision pretrained models, in this experiment, we regard ResNet-32 (He et al. 2016) as the small model, CLIP as the large model. The results on CIFAR-100-LT are show in Table **??**. Different from the experiment of language modality in the previous section, the performance of the small model is greatly affected by the number of training samples. The small model achieving 70.25%, 46.61% and 29.28% accuracy in the head, medium, and tail regions, respectively. The accuracy of the small model for the head data even significantly surpasses that of the large models, 60%, which further highlights that small models and large models can have their respective advantages. In contrast, small models yield similar results across most classes in the language modality. These outcomes might stem from a higher degree of similarity between comments, in contrast to images of different classes. Compared to established baselines that solely rely on either the small model or the large model, our approach yields significant performance improvements in all cases, especially, the overall accuracy has been improved by 5.07% compared to the large model, while saving approximately one-third of the costs.

Data Shunt for Multimodality

In this section, we validate the effectiveness of the proposed DS on the image caption task, which is a generative task, different from the classification tasks in the previous sections.

We follow (Xu et al. 2015) to design the small model, where the encoder is a ResNet-101, and the decoder is an LSTM. To get the confidence of the small model for the input sample, we calculate the mean of probabilities that predict next words.

The confidence of the small model for an input image is:

$$\begin{aligned}
&\mathbf{i}_t = \sigma\left(W_i E \mathbf{y}_{t-1} + U_i \mathbf{h}_{t-1} + Z_i \hat{\mathbf{z}}_t + \mathbf{b}_i\right), \\
&\mathbf{f}_t = \sigma\left(W_f E \mathbf{y}_{t-1} + U_f \mathbf{h}_{t-1} + Z_f \hat{\mathbf{z}}_t + \mathbf{b}_f\right), \\
&\mathbf{c}_t = \mathbf{f}_t \mathbf{c}_{t-1} + \mathbf{i}_t \tanh\left(W_c E \mathbf{y}_{t-1} + U_c \mathbf{h}_{t-1} + Z_c \hat{\mathbf{z}}_t + \mathbf{b}_c\right), \\
&\mathbf{o}_t = \sigma\left(W_o E \mathbf{y}_{t-1} + U_o \mathbf{h}_{t-1} + Z_o \hat{\mathbf{z}}_t + \mathbf{b}_o\right), \\
&\mathbf{h}_t = \mathbf{o}_t \odot \tanh\left(\mathbf{c}_t\right), \\
&C_t = \frac{e^{z_i}}{\sum e_{z_d}}, \quad z_i \in F_{MLP}(\mathbf{h}_t).
\end{aligned} \tag{8}$$

where $\mathbf{i}_t$, $\mathbf{f}_t$, $\mathbf{c}_t$, $\mathbf{o}_t$, $\mathbf{h}_t$ are the input, forget, memory, output and hidden state of the LSTM, respectively. $\hat{\mathbf{z}}$ is the context vector, capturing the visual information, as explained below. E is an embedding matrix. F_{MLP} is a full connected network. σ and $\odot$ is the logistic sigmoid activation and elementwise multiplication, respectively.

The image caption experiments are conducted on Microsoft COCO (Lin et al. 2014), which comprises 82,783 images with captions. We follow (Xu et al. 2015) to devide the training, validation, and testing set. As for the large model, we employ BLIP-2 (1.1B) (Li et al. 2023b). In this experiment, when the confidence computed by 8 is lager than 0.55, the input data will be processed by the small model, otherwise it is processed by the large model. Related results about BLEU (Papineni et al. 2002) are reported in Table **??**. It can be observed that the small model is inferior to the large model in terms of every metric. Additionally, the small model has much poorer ability to generate fluent sentences compared to the large model, as the difference between the small and large models becomes more significant with the n-gram (BLEU-n) increasing. As for DS, it can leverage the strengths of both the small model and the large model, resulting in improved performance on BLEU-1 and BLEU-2. However, when it comes to BLEU-3 and BLEU-4, which involve longer word combinations, DS falls short of surpassing the performance of the large model. BLEU-1 is closely

	DS	DS-2CD	DS-PP-2CD
Head	**71.99%**	71.21%	71.54%
Med	**59.91%**	58.69%	59.76%
Tail	**57.61%**	56.17%	53.31%
Overall Accuracy	**63.25%**	62.11%	61.63%
Query Proportion	**66.10%**	67.48%	67.48%

Table 4: Ablation experiments of DS on CIFAR-100-LT.

related to the previous classification task, in which it predicts the likelihood of specific words appearing based on images. It can be found that the proposed DS yields better performance in prediction tasks. This might be attributed to DS directing data flow based on the confidence of the predictions. Nonetheless, this experiment still validates the effectiveness of the proposed DS for image caption task, as DS successfully improve in the average BLEU score, while solely 65.36% of the data is computed by the large model.

Ablation for PP and 2CD

In this section, we conduct ablation experiments to show the effectiveness of PP and 2CD on CIFAR-100-LT. Related resulrs are shown in Table **??**.

It can be seen that both PP and 2CD have a positive impact on the proposed method. Besides, according to the Query Proportion of DS, 66.10%, and DS-2CD, 67.48%, 2CD can further reduce the number of times calling the large model, as small models have learned more data distributions. Moreover, from the comparison between DS-2CD and DS-2CD-PP, we find that PP primarily works on the tail data, as the accuracy improved by 2.86%. It is in line with our previous idea in section *Small Models for Large Models, Prompt Pruning*, with the prior knowledge of the small model, PP can reduce the candidate classes and improve the accuracy of the tail data.

Hyperparameter Analysis

This section primarily focuses on the important hyperparameter, shunt threshold, δ, which governs the data flow. Specifically, when the confidence of a sample is larger than δ, this sample will solely be processed by small models, otherwise, this sample will be processed by large models. To better present the influence of δ, we conduct related experiments on sentiment analysis based on the prior section.

As illustrated in Figure 3, DS with three different small models all can surpass the large model. When DS surpasses the large model, the requirement for the hyperparameter δ becomes lower for better-performing small models (i.e., δ can have a wider range). For example, DS with TextCNN or LSTM requires $\delta > 0.97$, while DS with fine-tuned BERT requires $\delta > 0.85$. Besides, when $\delta > 0.97$, the overall performance of DS shows only a slight improvement, while the proportion of samples processed by the large model increases significantly. For instance, when $\delta = 0.97$ and 0.99, the accuracy of DS with LSTM is 94.20% and 94.45%, respectively. As the parameter δ increases, the change in accuracy is minimal, while the query proporyion has significantly increased from 54.66% to 80.00%. Although such

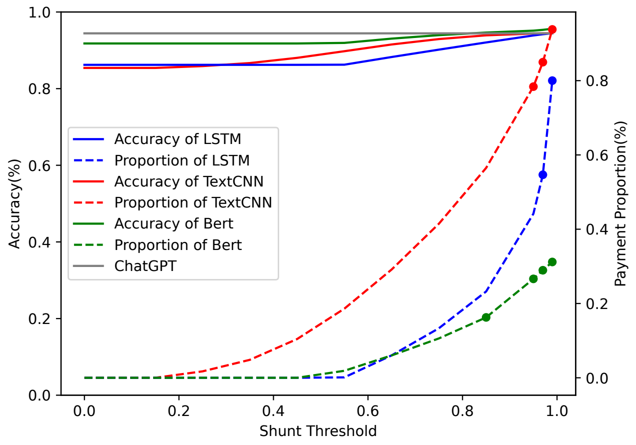

Figure 3: Hyperparameter analysis for shunt threshold on sentiment analysis. The solid line represents the accuracy of DS with different small models. The dotted line represents the proportion of samples computed by the large model. The bold dot represents DS achieves better performance than that of the large model.

phenomenon is not as prominent for DS with TextCNN (i.e., the vertical distance between the two points on the right side of the red dashed line) and DS with BERT (i.e., the vertical distance between the two points on the right side of the green dashed line), it is important to select a δ between $[0.97, 0.99]$ on the validation dataset.

Conclusion

With the advancements in pretrained large models, an increasing number of related applications are gradually becoming integrated into people's daily lives. The enormous computational resources required by pretrained large models have deterred the majority of product teams and individuals. Utilizing pretrained large models through interface can incur significant costs. Therefore, we introduce a collaborative paradigm that combines large and small models. Specifically, the input data is first processed by small models and then handed over to large models based on the confidence levels. To further leverage the advantages between the small and large models, we proposed Prompt Pruning (PP) and 2-Stage Confidence Distillation (2CD), which respectively uses small models to help large models refine the prediction space and uses large models to assist small models in learning unfamiliar distributions. We validate the effectiveness of the proposed method across diverse multimodalities and tasks, and the proposed method can significantly improve the performance while effectively reducing the frequency of querying large models.

Acknowledgments

This work is supported by National Key R&D Program of China under Grant 2022ZD0160101, NSFC (No. 62272411), the Zhejiang NSF (LR21F020004), and Ant Group, My Bank.

References

Alayrac, J.-B.; Donahue, J.; Luc, P.; Miech, A.; Barr, I.; Hasson, Y.; Lenc, K.; Mensch, A.; Millican, K.; Reynolds, M.; et al. 2022. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35: 23716–23736.

Biswas, S. 2023. Role of Chat GPT in Education. *Available at SSRN 4369981*.

Bommasani, R.; Hudson, D. A.; Adeli, E.; Altman, R.; Arora, S.; von Arx, S.; Bernstein, M. S.; Bohg, J.; Bosselut, A.; Brunskill, E.; et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Chen, L.; Zaharia, M.; and Zou, J. 2023. FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance. *arXiv preprint arXiv:2305.05176*.

Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H. W.; Sutton, C.; Gehrmann, S.; et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

French, R. M. 1999. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4): 128–135.

Gou, J.; Yu, B.; Maybank, S. J.; and Tao, D. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129: 1789–1819.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

He, R.; and McAuley, J. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*, 507–517.

Hershey, J. R.; and Olsen, P. A. 2007. Approximating the Kullback Leibler divergence between Gaussian mixture models. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, volume 4, IV–317. IEEE.

Hoffmann, J.; Borgeaud, S.; Mensch, A.; Buchatskaya, E.; Cai, T.; Rutherford, E.; Casas, D. d. L.; Hendricks, L. A.; Welbl, J.; Clark, A.; et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Horenko, I. 2020. On a scalable entropic breaching of the overfitting barrier for small data problems in machine learning. *Neural Computation*, 32(8): 1563–1579.

Kim, Y. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Li, B.; Han, Z.; Li, H.; Fu, H.; and Zhang, C. 2022. Trustworthy long-tailed classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6970–6979.

Li, J.; Gao, M.; Wei, L.; Tang, S.; Zhang, W.; Li, M.; Ji, W.; Tian, Q.; Chua, T.-S.; and Zhuang, Y. 2023a. Gradient-Regulated Meta-Prompt Learning for Generalizable Vision-Language Models.

Li, J.; Li, D.; Savarese, S.; and Hoi, S. 2023b. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*.

Li, J.; Pan, K.; Ge, Z.; Gao, M.; Zhang, H.; Ji, W.; Zhang, W.; Chua, T.-S.; Tang, S.; and Zhuang, Y. 2023c. Fine-tuning Multimodal LLMs to Follow Zero-shot Demonstrative Instructions. *arXiv preprint arXiv:2308.04152*.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, 740–755. Springer.

Liu, P.; Yuan, W.; Fu, J.; Jiang, Z.; Hayashi, H.; and Neubig, G. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9): 1–35.

McAuley, J.; Targett, C.; Shi, Q.; and Van Den Hengel, A. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, 43–52.

Oksuz, K.; Cam, B. C.; Kalkan, S.; and Akbas, E. 2020. Imbalance problems in object detection: A review. *IEEE transactions on pattern analysis and machine intelligence*, 43(10): 3388–3415.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35: 27730–27744.

Ouyang, W.; Wang, X.; Zhang, C.; and Yang, X. 2016. Factors in finetuning deep model for object detection with long-tail distribution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 864–873.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, 311–318.

Smith, S.; Patwary, M.; Norick, B.; LeGresley, P.; Rajbhandari, S.; Casper, J.; Liu, Z.; Prabhumoye, S.; Zerveas, G.; Korthikanti, V.; et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*.

Surameery, N. M. S.; and Shakor, M. Y. 2023. Use chat gpt to solve programming bugs. *International Journal of Information Technology & Computer Engineering (IJITC) ISSN: 2455-5290*, 3(01): 17–22.

Wang, Y.; Huang, M.; Zhu, X.; and Zhao, L. 2016. Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, 606–615.

Wei, J.; Tay, Y.; Bommasani, R.; Raffel, C.; Zoph, B.; Borgeaud, S.; Yogatama, D.; Bosma, M.; Zhou, D.; Metzler, D.; et al. 2022. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A.; Salakhudinov, R.; Zemel, R.; and Bengio, Y. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, 2048–2057. PMLR.

Yu, Q.; Li, J.; Wei, L.; Pang, L.; Ye, W.; Qin, B.; Tang, S.; Tian, Q.; and Zhuang, Y. 2023. HalluciDoctor: Mitigating Hallucinatory Toxicity in Visual Instruction Data. *arXiv preprint arXiv:2311.13614*.

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

Zhang, X.; Fang, Z.; Wen, Y.; Li, Z.; and Qiao, Y. 2017. Range loss for deep face recognition with long-tailed training data. In *Proceedings of the IEEE International Conference on Computer Vision*, 5409–5418.

Zheng, L.; Li, Z.; Zhang, H.; Zhuang, Y.; Chen, Z.; Huang, Y.; Wang, Y.; Xu, Y.; Zhuo, D.; Xing, E. P.; et al. 2022. Alpa: Automating inter-and {Intra-Operator} parallelism for distributed deep learning. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*, 559–578.

Zhou, Y.; Hu, Q.; and Wang, Y. 2018. Deep super-class learning for long-tail distributed image classification. *Pattern Recognition*, 80: 118–128.

Zhu, J.; Lai, S.; Chen, X.; Wang, D.; and Lu, H. 2023. Visual prompt multi-modal tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9516–9526.