# Sorted LLaMA: Unlocking the Potential of Intermediate Layers of Large Language Models for Dynamic Inference

**Parsa Kavehzadeh**[2], **Mojtaba Valipour**[1,2], **Marzieh Tahaei**[2],
**Ali Ghodsi**[1], **Boxing Chen**[2], and **Mehdi Rezagholizadeh**[2]
[1]University of Waterloo
[2]Huawei Noah's Ark Lab
{mojtaba.valipour, ali.ghodsi}@uwaterloo.ca,
{parsa.kavehzadeh, mehdi.rezagholizadeh, marzieh.tahaei, boxing.chen}@huawei.com

## Abstract

Large language models (LLMs) have revolutionized natural language processing (NLP) by excelling at understanding and generating human-like text. However, their widespread deployment can be prohibitively expensive. SortedNet is a recent training technique for enabling dynamic inference by leveraging the modularity in networks and sorting sub-models based on computation/accuracy in a nested manner. We extend SortedNet to generative NLP tasks, making large language models dynamic without any Pre-Training and by only replacing Standard Fine-Tuning (SFT) with Sorted Fine-Tuning (SoFT). Our approach boosts model efficiency, eliminating the need for multiple models for various scenarios during inference. We show that this approach can unlock the potential of intermediate layers of transformers in generating the target output. Our sub-models remain integral components of the original model, minimizing storage requirements and transition costs between different computational/latency budgets. The efficacy of our proposed method was demonstrated by applying it to tune LLaMA 2 13B on the Stanford Alpaca dataset for instruction following and TriviaQA for closed-book question answering. Our results show the superior performance of sub-models in comparison to Standard Fine-Tuning and SFT+ICT (Early-Exit), all achieved with very efficient tuning and without additional memory usage during inference.

## 1 Introduction

Large language models are revolutionizing the way we interact with information in today's world (Hoffmann et al., 2022; Brown et al., 2020; Penedo et al., 2023; Scao et al., 2022). New models are continually emerging, demonstrating their capabilities in understanding and, more importantly, in generating human-like text. Notably, models such as ChatGPT, LLaMA 2 70B (Touvron et al., 2023b), and Falcon 180B (Almazrouei et al., 2023) have had a profound impact on the applicability of large language models (LLMs). However, deploying these expansive language models can become prohibitively expensive.

What distinguishes this new era of ChatGPT-like models is their ability to perform an extraordinarily wide array of tasks in natural language processing (NLP), reasoning, and more, all through behavior cloning (Wei et al., 2021; Wang et al., 2022). In fact, a single model can leverage the strong contextual learning ability offered by Standard Fine-Tuning to address numerous tasks, spanning from language comprehension to complex reasoning. While this unified usage simplifies the deployment of these models as general assistants, it remains highly inefficient. Enabling dynamic inference, where the computational resources allocated to a given query vary at inference time, can significantly enhance the practicality of employing such models in real-time scenarios. This enables the use of smaller models when the budget is limited or latency is critical. It is important to note that dynamic inference strategies for large models with a substantial number of parameters should not require loading different models during inference.

Previous research has explored methods for training dynamic models capable of adapting to evolving resource constraints (Cai et al., 2019; Hou et al., 2020; Xin et al., 2020; Fan et al., 2019). However, existing approaches often rely on complex training procedures or necessitate modifications to the original model architecture. SortedNet (Valipour et al., 2023) introduces a novel approach to training deep neural networks that leverages the inherent modularity of these networks to construct sub-models with varying computational loads. This method sorts sub-models hierarchically based on their computation/accuracy characteristics, facilitating efficient deployment during inference. Furthermore, it employs an efficient updating scheme combining random sub-model sampling with gradient accumu-



*Findings of the Association for Computational Linguistics: EACL 2024*, pages 2129–2145

lation to minimize the training cost. Consequently, with a single round of training, numerous models can be obtained within a single model.

While the SortedNet approach has primarily been applied to vision and language understanding tasks, given the significant impact of generative language models in today's AI landscape, the efficacy of this method for generative tasks in NLP is of considerable interest. In fact, being able to make a large language model dynamic without the need for Pre-Training and only at the cost of a round of Standard Fine-Tuning can open doors to efficient inference of these models without incurring additional expenses associated with common model compression methods like knowledge distillation and pruning, among others. Moreover, since all the resultant models are components of the original model, the storage requirements and the cost associated with transitioning between different computation demands become minimal. Otherwise, managing multiple models for various scenarios during inference becomes impractical.

In this study, we challenge the conventional approach of relying solely on the last layer's contextual embeddings and use Sorted Fine-Tuning (SoFT) in place of Standard Fine-Tuning to enhance the performance of these models across multiple layers. By doing so, we aim to provide new insights into the efficiency and effectiveness of middle layers in producing high-quality results for specific downstream tasks. Our proposed approach can potentially optimize these sub-models in addition to the main model, ultimately enhancing their overall performance. In this paper, we seek to answer the following questions through systematic evaluation:

i) Do the intermediate layers resulting from Standard Fine-Tuning of a large language model generate accurate and meaningful outputs? ii) Does Standard Fine-Tuning exhibit a sorted behavior, meaning that later layers produce more accurate and meaningful results than earlier layers? If so, to what extent? iii) How can we enhance this sorted behavior with minimal cost?

To answer these questions, we employ LLaMA 2 13B and perform both Standard Fine-Tuning (SFT) and Sorted Fine-Tuning (SoFT) on the Stanford Alpaca (Taori et al., 2023) and TriviaQA (Joshi et al., 2017) datasets. For Sorted Fine-Tuning, we target 8 sub-models and share the LLM head among them to ensure cost parity. We utilize the PandaLM benchmark (Wang et al., 2023) to assess the performance of the sub-models on Alpaca dataset. Our findings demonstrate the superior performance of SoFT in comparison to SFT and even to memory-demanding methods like Early Exit (Xin et al., 2020). The contributions of this paper can be summarized as follows:

- Extending the SortedNet method for tuning auto-regressive language models for generative tasks by sharing a single LLM head layer among sub-models.
- Generating 8 nested sub-models, ranging from 12 to 40 layers, from LLaMA2 13B by applying Sorted Fine-Tuning on the Stanford Alpaca dataset and TriviaQA benchmarks and at a cost equivalent to Standard Fine-Tuning.
- Evaluating the performance of the sub-models of a LLaMA 2 and demonstrating the effectiveness of SoFT in enhancing the ability of intermediate layers for text generation and question answering through extensive evaluation.

## 2 Related Work

This section briefly introduces the most relevant papers to our work.

**Many-in-One Models** Deep neural networks (DNNs) are often overparameterized, motivating researchers to explore ways to use the parameters of the models more efficiently. More number of parameters lead to higher costs of deployment for neural networks. Moreover, in practice, these overparametrized DNNs are expected to accommodate customers with varying requirements and computational resources. To address these diverse demands, one can think of training models of different sizes, which can be prohibitively costly (in terms of training and memory), or another alternative is to train many-in-one networks (Cai et al., 2019). Many-in-one solutions aim to train a network along with some of its sub-networks simultaneously for specific tasks. For example, we can consider the *Early-Exit* method (Xin et al., 2020), wherein a prediction head is fine-tuned on top of specific intermediate layers within a network. Another approach is *Layer Drop* (Fan et al., 2019), which trains a network in any depth by randomly dropping the layers during training. While both Early-Exit and Layer Drop are simple solutions, they are not state-of-the-art in terms of performance. In Early-Exit, we only train the output prediction layer on top of each intermediate layer, and this layer might not have enough

capacity to retain a good performance. Layer Drop, conversely, suffers from the abundant number of possible sub-models in training, which makes the training process exhaustive and sub-optimal. Furthermore, this approach requires tuning the extent of dropping layers during training. This additional hyper-parameter, layer drop rate during training determines the best size and setting of the model at the inference time. Deviating from the training drop rate at the inference time can result in a significant drop in performance.

Cai et al. (2019) in *Once for All (OFA)* proposed an alternative solution to neural architecture search (NAS). OFA requires training the model and all possible sub-models in an arbitrary progressive way followed by a separate search phase. Dyna-BERT (Hou et al., 2020) is another work that targets training Dynamic pre-trained many-in-one BERT models in two stages: first, distilling from the main network to the width adaptive networks and then distilling from the width adaptive networks to depth adaptive networks. Both width adaptive and depth adaptive networks have a limited pre-defined set of width and depth for the sub-models. While both OFA and DynaBERT have shown successful results, their solutions are hardly applicable to multi-billion-parameter LLMs because of their complicated multi-stage training process and their search and knowledge distillation requirements. SortedNet (Valipour et al., 2023) is a recent method that forms and trains sub-models of a network in a sorted manner while not requiring any search during training or inference. SortedNet has shown superior performance compared to other previously mentioned methods in terms of simplicity, performance, scalability, and generalization. Considering these benefits, we target deploying the SortedNet training algorithm for developing many-in-one LLMs.

**Many-in-One Large Language Models (LLMs)** Large language models have recently gained significant attention in the literature (Touvron et al., 2023a; Brown et al., 2020; OpenAI, 2023; Chowdhery et al., 2022; Ouyang et al., 2022). In practice, these LLMs serve users with different tasks, expectations, and computational budget requirements (Sun et al., 2022). There are two types of adaptation approaches to make LLMs suitable for customer requirements: first is the so-called parameter efficient tuning (PEFT), and second is model compression. In PEFT, the core backbone model remains the same, and we just update much smaller adapter parameters (e.g. LoRA (Hu et al., 2021), KRONA (Edalati et al., 2022), Adapter (Houlsby et al., 2019; Pfeiffer et al., 2020), DyLoRA (Valipour et al., 2022), Ladder Side-Tuning (Sung et al., 2022)) and Compacter (Karimi Mahabadi et al., 2021). In model compression, the larger model is compressed using any model compression solutions such as knowledge distillation (Hinton et al., 2015; Hsieh et al., 2023; Wu et al., 2023), pruning (Bansal et al., 2023), and quantization (Prato et al., 2019; Dettmers et al., 2023), a good related survey can be found in (Zhu et al., 2023). Even though PEFT solutions are pretty popular with LLMs, they do not provide dynamic-size LLMs. Model compression solutions can provide models with different sizes, but they need to train each compressed model separately, and they are not many-in-one models.

To the best of our knowledge, this work is the first attempt to obtain many-in-one generative LLMs by applying the Sorted Fine-Tuning to the LLaMA 13B model. Considering the benefits of many-in-one networks and the growing application of LLMs, we are hoping that this will help the community to build more efficient large language models that can be deployed adaptive during inference using methods such as SoFT.

## 3 Methodology

This paper focuses on making generative LLMs many-in-one by unlocking the potential of intermediate layers through the SortedNet approach (Valipour et al., 2023).

Let's consider a language model $f(x;\theta)$ with the parameters $\theta$ and the input $x$. The following is the sorted training procedure:

**Forming Sub-Networks** First, we need to form the sub-networks of the LLM. For the sake of simplicity and without loss of generality, we focus on the depth-wise sub-networks. Supposed that the sub-network $f_n(x;\theta_n)$ refers to the first $n$ layers of $f(x;\theta)$. In this paper, the language model is considered to be LLaMA2 13B. Since LLaMA2 comprises 40 layers, we define the sub-networks as $n \in \mathbf{B} = \{12, 16, 20, 24, 28, 32, 36, 40\}$.

**Calculating the Output of Sub-Networks** Each sub-model's output will be predicted using the shared output prediction head from the last layer (original network). Remember that in the LLaMA

**Query**: Summarize a meeting from the given list of bullet points. Be sure to convert shorthand into a first-hand account.
Rose: Analyze data and presents findings
John: propose new idea
Jane: appointed to head project
Tom: need more time to fix software bug

| Layer | LLaMA Response (SFT+ICT) | Sorted LLaMA Response (SoFT) |
|---|---|---|
| 12 | summatronics21220f0f ... | The meeting was held in order to discuss the project and its plans for the next few months. The presenters were present in the room, and they were able to provide a detailed explanation of the project and its goals. The feedback from the meeting was positive and encouraged the team to continue working towards the goal of delivering the project. |
| 16 | The first thing we hear about these characters individually separately apart aparticularly divisidedly mannered mannerednesses ... | The meeting held at the company's headquarters, where the three named individuals were present. The first person named in the report is Rousness, who has been assigned to task task task. The second person named in the report is John, who has proposed a new idea. The third person named in the report is Jane, who has been assigned to task task task. The third person named in the report is Jane, who has been assigned to task task task. The second person named in the report is Jane, who has been assigned to task task task. |
| 20 | meeting held meeting held meeting held participants discussion discussion discussion notable mentionsnone none None notable mentionsnone ... | The meeting held in the room with Rose, John, Jane, and Tom discusses the need for analyzing data and presenting findings , as well as the proposal of new ideas by John. Jane is appointed to head the project, and Tom needs more time to fix the software bug. |
| 24 | meeting summary: attended attended attended attended ... | The meeting was attended by Rose, John, Jane, and Tom. Rose was tasked with analyzing data and presenting findings. John proposed a new idea. Jane was appointed to head the project. Tom was granted additional time to fix the software bug. |
| 28 | summary: During meeting participants discuss data analysis findings presented by Rose followed proposal new idea by John appoint Jane head project Tom request additional time fix software bug | In the meeting, Rose analyzed data and presented findings, John proposed a new idea, and Jane was appointed to head the project. Tom also stated that he needed more time to fix the software bug. |
| 32 | summary: During meeting attended by Rose Analyst who analyzed data presented findings proposed new idea by John appointed Jane head project Tom Software engineer request more time fix bug | In the meeting, Rose analyzed data and presented her findings, while John proposed a new idea. Jane was appointed to head the project, and Tom requested more time to fix a software bug. |
| 36 | During the meeting, Rose presented her analysis of data collected from various sources. John proposed a new idea that could potentially improve efficiency and increase productivity. Jane was appointed head of a new project that will be launched soon. Tom requested more time to fix a software bug that has been affecting the company'vity since last month. | In the meeting, Rose analyzed data and presented her findings, while John proposed a new idea. Jane was appointed to head the project, and Tom requested more time to fix the software bug. |
| 40 | At the meeting, Rose presented the analysis of the data and shared her findings with the team. John proposed a new idea that he believed could benefit the company. Jane was appointed to head a new project that the company had recently taken on. Tom, however, requested more time to fix a software bug that had been discovered. The team discussed the importance of the bug fix and agreed to give Tom the additional time he needed. | In the meeting, Rose analyzed data and presented her findings. John proposed a new idea, and Jane was appointed to head the project. Tom requested more time to fix the software bug. |

Table 1: Generated responses in different layers of LLaMA (Early Exit) and Sorted LLaMA for two samples from PandaLM validation set. Black phrases mean the answer is following the instruction and Red ones demonstrate errors and hallucinations.

model, there is an RMSNorm layer (Zhang and Sennrich, 2019) before the output prediction head. This RMSNorm is added before the shared prediction head of every sub-model. This normalization may be an important factor that helps Sorted LLaMA to generalize better for all sub-models.

**Objective Function** Let $L_n(x;\theta_n)$ be the loss for the $n^{\text{th}}$ sub-model for input batch $x$. To train the network, we define the loss as the summation of the losses of all these sub-models:

$$\mathcal{L} = \frac{\sum_{n\in\mathbf{B}} L_n(x;\theta_n)}{|\mathbf{B}|} \quad (1)$$

For the experiments conducted in the paper, $|\mathbf{B}| = 8$. Note that these sub-models have shared parameters through a nested style i.e. $\theta_1 \subset \theta_2 ... \subset \theta_n$.

**Training Dataset** We utilized the Stanford Alpaca dataset (Taori et al., 2023), which includes demonstrations of 52K instruction-following examples. We also used TriviaQA open-domain QA benchmark (Joshi et al., 2017) including 110K closed-book question-answer pairs.

**Evaluation** In this paper, in addition to embedding the last layer, we evaluate the quality of the embeddings of intermediate outputs spanning from block 1 to n. PandaLM benchmark (Wang et al., 2023) compares the output of different sub-models. PandaLM deploys a large language model (Fine-Tuned LLaMA 7b) to judge the quality of generated text from two sources. PandaLM provides a valida-

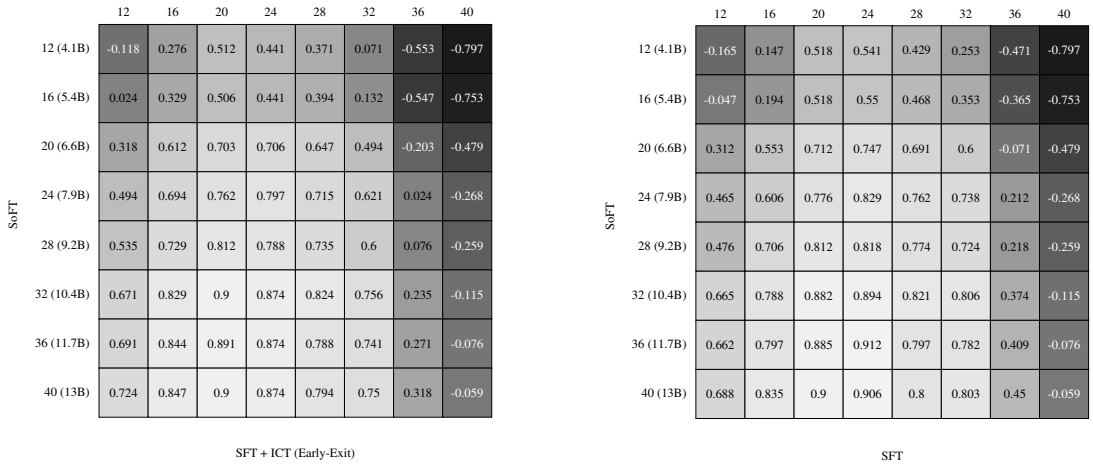



Figure 1: SoFT vs. SFT + ICT (Early-Exit) (Left) and SoFT vs. SFT (Right). Note that for our SoFT method, the output prediction layer is shared between all sub-models whereas, for Early-Exit, a separate prediction head is learned per sub-model, making inference inefficient. Both SoFT and SFT had equivalent training time (2 Epochs) in this experiment. The number in each cell is calculated by considering wins as the times SoFT sub-models (rows) were preferred, losses as the times SFT sub-models (columns) were preferred and ties when non of them were preferred (Equation 2).

tion set consisting of 170 instructions[1], to evaluate target models for instruction-following tasks. To ensure that the order of the models' responses does not influence the judgment of the PandaLM evaluator, we reported an average score under both the Model 1 first and the Model 2 first scenarios. The output of the PandaLM evaluation is the number of wins, denoted as $W$, the number of losses, denoted as $L$, and the number of ties, denoted as $T$, in the validation set. The final reported score has been calculated using the following formula:

$$Score = \frac{(W - L)}{T} \quad (2)$$

The final score is a number between -1 and 1, in which 1 represents a strong win rate and -1 means a poor performance of the model.

We used accuracy (exact match) as the evaluation metric for the TriviaQA benchmark.

**Baseline** The primary objective of the LLM in this paper is to follow the provided instructions by a query. Therefore, following the setup of Alpaca (Taori et al., 2023), we fine-tuned LLaMA2 13B on the Stanford Alpaca Dataset with two setups: (1) Regular Standard Fine-Tuning (SFT) as the baseline, focusing only on the training of the last layer of the network as the common practice in the literature; (2) Sorted Fine-Tuning (SoFT), calculating loss for multiple outputs from layer 12 to layer 40 (last layer) with four intervals, and training multiple models simultaneously as explained in the previous section.

[1] github.com/WeOpenML/PandaLM/blob/main/data/testset-inference-v1.json

## 4 Experiments

This section delves into the experiments' specifics and the analysis provided to understand better the effect of Sorted Fine-Tuning over the performance of a large language model like LLaMA2 (Touvron et al., 2023b). Before diving into results, we are going to define certain notations that we used for different setups in our experiments:

- **SoFT/SFT**: We first train the model with SoFT or SFT paradigms and use the sub-models after training without any further training of the language model head for intermediate layers.
- **SFT+Intermediate Classifier Tuning (ICT)**: We first train the model with SFT paradigm and then further fine-tune the language model head exclusively for each sub-model while keeping their weights frozen. The SFT+ICT is also known as Early-Exit (Xin et al., 2020) in the literature.
- **Extracted Fine-Tuning**: When we extract the sub-models from the learned weights of the pre-trained original model and train each sub-model separately.

### 4.1 Experimental Setup

We used the pre-trained LLaMA2 13b weights, publicly available on Hugging Face, as our starting point. For SFT+ICT (Early-Exit) setup, we froze the parameters of the transformer blocks and only

further trained the weights of the language model head classifier for one additional epoch. We used a batch size of 32 and gradient accumulation of 8. The learning scheduler was cosine annealing. The learning rate was set to 2e-5 and seed to 42. We trained the models on 8 V100 32GB GPUs. The same GPUs were used during inference time. The training maximum input sequence length was 2024, with a maximum of 50 (TriviaQA) and 256 (PandaLM) generated tokens during inference. Additionally, we used greedy search as the decoding strategy in all of our experiments. We also extended the huggingface assisted decoding code to implement Speculative Decoding and Instance-Aware Adaptive Inference. In Speculative Decoding, we used adaptive K window-size (the same as huggingface) starting with K=4. In Instance-Aware Dynamic Inference, we set the confidence thresholds of intermediate layers as follow: Layer 12 = 0.95, Layer 16 = 0.95, Layer 20 = 0.9, Layer 24 = 0.9, Layer 28 = 0.8, Layer 32 = 0.8 and Layer 36 = 0.7.

### 4.2 What is the effect of sorting information across layers of a generative model?

As mentioned before, we generated responses for all the layers $n \in \mathbf{B}$ for both SFT and SoFT-based trained models. Then, we conducted a pair-wise comparison between all the sub-models in the two trained models using the PandaLM evaluator. As the results suggest in Figure 1, sorted training significantly unlocks the potential of intermediate layers in generating the desired output.

Sorted LLaMA (aka SoFT) is outperforming regular fine-tuning (SFT) in nearly all layer comparisons by a meaningful margin, as shown through automated evaluation in Figure 1.

It might be noted that the Layer 12 performance of SFT is slightly better compared to Layer 12 of Sorted LLaMA. We argue this is happening because the outputs of early layers in SFT are mostly gibberish (see Table 1 as an example), and the PandaLM evaluator has not been trained on such data. Hence the automatic evaluation results for this layer are not meaningful. To further investigate the reason behind the results for early sub-models, we conducted human evaluation on 6 cells of two tables in Figure 1 (Layer 12 of SFT and SFT+ICT vs Layers 12,16, and 20 SoFT) to verify our claim. We observed that SoFT early sub-models could significantly outperform sub-model layer 12 of both SFT and SFT+ICT models, proving the negative impact of gibberish text on PandaLM evaluator performance. As we go to higher layers in SFT, the generated text becomes meaningful, which makes the comparison with the Sorted LLaMA layer counterpart more reasonable.

Moreover, to improve SFT results, inspired by Early-Exit (Xin et al., 2020), we also tried the scenario in which a separate classifier head is dedicated to all sub-models of SFT. This method has been introduced in the notation section as SFT+ICT. These classification heads have been trained an additional epoch after SFT tuning while keeping the base model frozen. Note that this setting suffers from significant memory overhead during tuning and inference compared to our SoFT method. In fact, the extra number of parameters for SFT+ICT (Early Exit) is $|B| - 1 \times D \times V$, where $|B|$ is the number of sub-models, $D$ is the hidden size of the model, and $V$ is the vocabulary size. For LLaMA 2 13B, this is equivalent to 1B extra parameters.

The results of comparing sorted with the early exit are shown in figure 1 (Left). Despite having far more parameters, SFT+ICT (Early-Exit) underperforms our sorted tuning for most sub-models. According to the results, the sub-model in Sorted LLaMA with 36 layers performs almost as well as regular fine-tuning of the full-size model. This showcases the impressive ability of our proposed paradigm to generate powerful, small sub-models that perform similarly to the original model. Another experiment that has been conducted in appendix A.2, further investigated the impact of longer training time for SoFT. The results show that our model was still under-trained, and we could observe a significant improvement in Sorted LLaMA performance with longer training time.

Moreover, we compared the performance of Sorted LLaMA sub-models with the actual capacity of these models by fine-tuning the sub-models separately and reporting the results in both equal training time and more training time for SoFT. We extracted 4 sub-models (Layer 12, Layer 20, Layer 28, and Layer 36) and each time fully fine-tuned the extracted sub-model separately for two epochs on the Alpaca dataset. Figure 2 and Table 9 shows the comparison between Extracted Fine-Tuned and SoFT sub-models. The first part in Table 9 shows the equal training budget setup (2 Epochs) comparison in which SFT demonstrates slightly better performance compared to the similar SoFT sub-models. Further training SoFT will lead to better sorted sub-models in which SoFT outperforms the

fully fine-tuned sub-models, proving the positive impact of SoFT on the performance of lower sub-models.

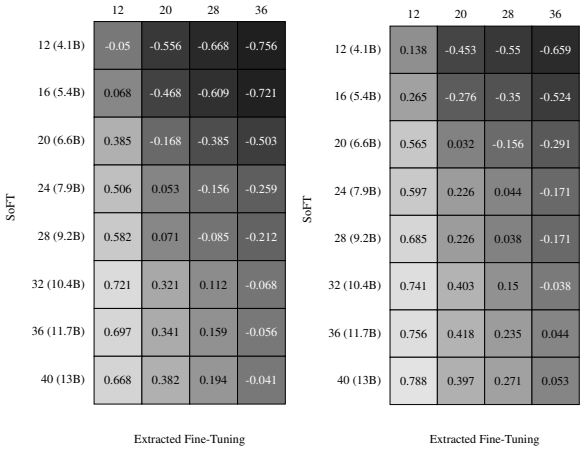



Figure 2: SoFT vs. Extracted Fine-Tuning. The left figure shows an equal training time setup (2 epochs), and the figure on the right considers two extra training epochs for SoFT.

### 4.3 How does SoFT work for other domains?

We further evaluated Sorted LLaMA in a different domain from the instruction following, selecting the TriviaQA (Joshi et al., 2017) benchmark to assess the sub-models performance in open-domain closed-book questions answering.

Figure 3 shows the performance of SoFT and three SFT, Extracted Fine-Tuning and SFT+ICT baselines in different checkpoints through the training procedure on the TriviaQA benchmark. SoFT sub-models show significant superior performance compared to SFT and SFT+ICT counterparts in all sub-models. Similar to PandaLM, the gap between SoFT and SFT full-model performance is small in TriviaQA, which can underscore the SoFT capability in maintaining full-model performance compared to SFT. We also did Extracted Fine-Tuning on intermediate sub-models for 2 Epochs and results demonstrate close performance of SoFT intermediate layers to Extracted Fine-Tuning counterparts.

### 4.4 How can SoFT accelerate text generation?

**Improving Speculative Sampling** Speculative Decoding (SD) is a technique introduced by (Chen et al., 2023) to increase the speed of text decoding in large models. The method utilizes a large target and smaller draft models to generate tokens faster. We can verify the generated tokens by the large model in parallel. We used the same paradigm for Sorted LLaMA as we used earlier sub-models as

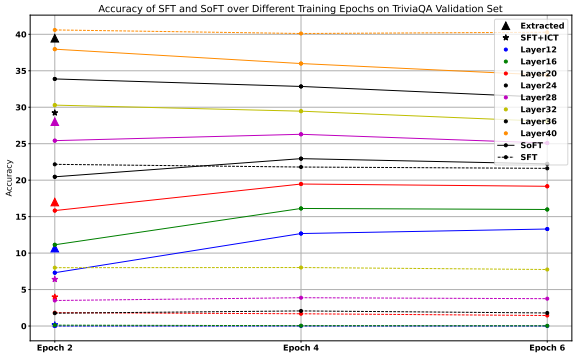



Figure 3: The results of TriviaQA. We reported case-sensitive exact match accuracy as the main metric. SFT+ICT and Extracted Fine-Tuned results can be found in Epochs 2, as we found Epoch 2 checkpoint saturated for the original SFT experiment (main LLaMA2 13b model with 40 layers).

draft and the full-size model as the target model. As the parameters have been shared between the large and draft models in this setup, we can avoid any extra memory overhead, unlike the standard Speculative Sampling. Table 2 reports the results of using speculative decoding on Alpaca and TriviaQA benchmarks in inference in SoFT by using three different sub-models as drafts (Layer 12, 16, and 20). Using Speculative decoding in Sorted LLaMA can speed up the token generation up to 1.16× compared to normal auto-regressive decoding in PandaLM with negligible performance drop compared. Duo to the short average length of answers in TriviaQA, speculative decoding does not result in speed up in this benchmark as the draft generation process does not find any opportunity to accelerate inference.

**Instance-Aware Dynamic Inference** We also dynamically utilize SoFT sub-models to increase text generation speed during inference. Based on the confidence of the sub-model's predicted tokens, we decide which sub-model needs to generate each token during inference. Given each token during inference, the sub-models would process the token in size order (first smallest sub-model 12, then 16, and so on). Wherever in this procedure, the confidence of the predicted token by a sub-model is higher than the defined confidence threshold, the predicted token would be chosen as the next token and exit the model. We also implemented an adaptive caching mechanism in order to utilize KV caching in this non-trivial scenario where each token can exit from a different layer. Table 2 shows that Instance-Aware Dynamic Inference can speed up the normal auto-regressive approach in all benchmarks up to 1.34× in PandaLM and 1.12× in TriviaQA. Furthermore

| PandaLM | | | | TriviaQA | | |
|---|---|---|---|---|---|---|
| **Auto-regressive Decoding** | | | | | | |
| **Model** | **Time per Token (ms)** | **Score** | **Rejection Ratio** | **Time per Token (ms)** | **Accuracy** | **Rejection Ratio** |
| Layer 40 (full) | 94.07 | - | - | 91.27 | 37.95 | - |
| **Speculative Decoding** | | | | | | |
| **Draft Model** | **Time per Token (ms)** | **Score** | **Rejection Ratio** | **Time per Token (ms)** | **Accuracy** | **Rejection Ratio** |
| Layer 12 | 80.86 (1.16×) | -0.144 | 0.37 | 110.50 (0.82×) | 34.36 | 0.72 |
| Layer 16 | 84.10 (1.11×) | -0.211 | 0.31 | 118.92 (0.76×) | 34.16 | 0.70 |
| Layer 20 | 84.50 (1.11×) | -0.144 | 0.26 | 139.78 (0.65×) | 34.19 | 0.66 |
| **Instance-Aware Dynamic Inference** | | | | | | |
| **Model** | **Time per Token (ms)** | **Score** | **Rejection Ratio** | **Time per Token (ms)** | **Accuracy** | **Rejection Ratio** |
| Layer 12:40 | 69.91 (1.34×) | -0.050 | - | 81.01 (1.12×) | 36.53 | - |

Table 2: Speed-up in inference time on three PandaLM and TriviaQA benchmarks by utilizing Speculative Decoding and Instance-Aware Dynamic Inference techniques. Score column in PandaLM section means the score of the model versus the Auto-regressive generated results based on Equation 2.

dynamic inference can result in better performance in PandaLM and TriviaQA compared to speculative decoding.

## 4.5 Analysis

### 4.5.1 A comparison between the learned probability distribution of SoFT versus SFT

Sorted tuning aims to make sub-models performance similar to the full model. To explore the efficacy of the SoFT in closing the gap between sub-models and the full model in instruction following task, we measure the similarity between probability distributions of each token in each sub-model versus the full model using the Kullback–Leibler (KL) divergence. Figure 4 (Left) compares the probability distribution of Sorted LLaMA and SFT sub-models at different output positions.

Figure 4a (Left) compares different SFT layers and the last Sorted LLaMA layer. The figure shows that only SFT's full-size output distribution is close to the sorted full-size model, while the other layers' distribution diverges faster in the initial steps compared to the SoFT. This is expected as the language model head is unfamiliar with the learned representation of the middle layers in SFT. In the next section, we compared the learned representations of different sub-models to understand SoFT's impact better.

Figure 4b (Left) compares the output distribution of all sorted layers to the last SFT layer. Compared to Figure 4a (Left), Figure 4b (Left) Sorted LLaMA can preserve the output distribution close to the SFT full-size model even in lower layers for initial output tokens.

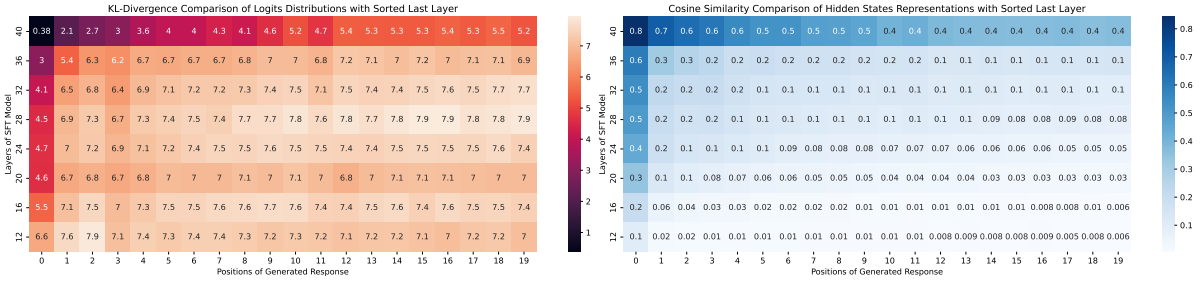
The comparison between the last layer and the layers 12 to 36 in the SFT model is shown in Figure 5a (Left). It is clear from this figure that the output distribution diverges quickly compared to the last layer after generating a few initial tokens, even in higher layers like 36 and 32. It is important to note that this evaluation was generated without adjusting the classifier head.

Finally, Figure 5b (Left) demonstrates that in Sorted LLaMA, the likelihood distribution of the produced outcome becomes increasingly more similar to the full-size model as we get closer to the last layer.

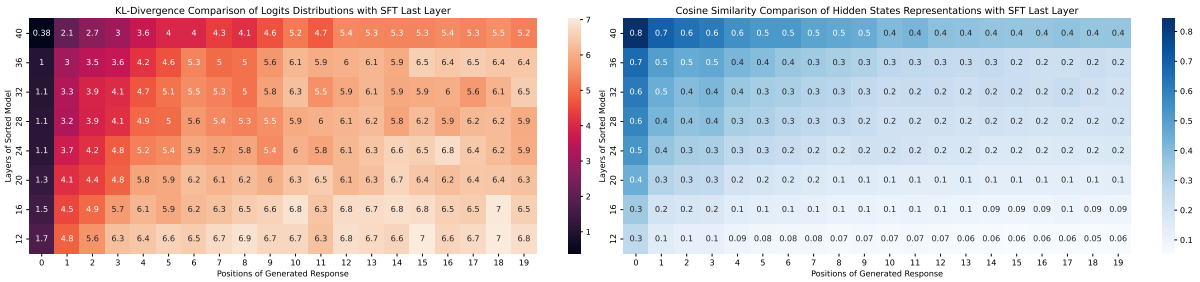
### 4.5.2 A comparison between the learned representation of SoFT versus SFT

During regular fine-tuning, no connection between the language model head and sub-models can intensify the divergence of probability distributions in Figure 4 (Left). To overcome this, we conducted another experiment to compare the hidden state representation in the last and middle layers just before passing the hidden states to the language model head. Figure 4 (Right) compares the learned hidden state representation of SFT and Sorted LLaMA sub-models at various positions in the output. This will make the analysis independent of the language model head. We used cosine similarity to measure the difference between the two representations. As shown using heatmaps, the cosine similarities are highly correlated to the KL-Divergence comparison explained in the previous section.

Figure 4a (Right) compares all SFT sub-models with the Sorted last layer regarding hidden representation similarity. Again, similar to probability distribution analysis, the similarity between the SFT sub-model and Sorted last layer tends to fade immediately after generating the first few tokens, while




(a) SortedLLaMA Last Layer - LLaMA sub-models




(b) SortedLLaMA sub-models - LLaMA Last Layer

Figure 4: A sub-model comparison based on output logits and hidden state cosine similarity. The numbers are average of all 170 samples in the PandaLM validation set.

Figure 4b demonstrates the capability of Sorted LLaMA sub-models in preserving the learned representations closely similar to the SFT last layer hidden states.

Figure 5a (Right) depicts the heatmap of hidden states cosine similarity among different SFT sub-models compared to the SFT last layer. Similar to its left plot, the similarity quickly diminishes after a few tokens, and this fade is more considerable in earlier layers.

On the other hand, Figure 5b (Right) shows that the representations of Sorted sub-models stay similar to the Sorted last layer even after generating multiple initial tokens.

### 4.5.3 Case Specific Analysis

Table 1 shows a sample of instructions from the PandaLM benchmark and the generated responses by SFT+ICT (Early-Exit) and Sorted LLaMA sub-models. Sorted LLaMA performs better in preserving and transferring the last layer performance to earlier sub-models based on the information made visible by black (related to the query) and red (hallucinations, irrelevant, etc.) colors.

Sorted sub-models generate almost correct answers from the 20 layers sub-model, while the first meaningful result from SFT+ICT sub-models appears in layer 28. Other samples generated by SoFT and Early-Exit can be found in A.3.

## 5 Conclusion

This work presents sorted LLaMA, a many-in-one language model for dynamic inference obtained using Sorted Fine-Tuning (SoFT) instead of Standard Fine-tuning. Sorted LLaMA unlocks the potential capability of intermediate layers, offering dynamic adaptation without pre-training or additional expenses related to model compression. It presents a promising avenue for optimizing generative language models in NLP. Our approach makes the deployment of these models far more efficient. As all sub-models remain integral components of the original model, the burden of storage requirements and transition costs between different computational demands is minimized, making the management of multiple models during inference a practical reality.

Our systematic evaluation of instruction following and questions answering benchmarks challenged conventional wisdom by empowering middle layers to produce high-quality results. This, in turn, enables dynamic inference of LLMs with a highly efficient tuning method (SoFT), ultimately optimizing the usage of LLMs. Our encouraging results show the promising capability of SortedNet (Valipour et al., 2023) to train multiple language models with different sizes at once without incurring expensive costs.

## 6 Limitations

Despite showing the effectiveness of the SortedNet approach for large language models, further research is necessary to better understand the scope of its applicability in LLMs. For example, applying this method during pre-training, sorting other model dimensions such as attention heads and hidden dimensions, and investigating the impact of choosing a specific architecture could offer potential avenues for future research. Our study might be slightly biased to automated evaluation, requiring further investigation through human evaluation.

## References


Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Maitha Alhammadi, Mazzotta Daniele, Daniel Heslow, Julien Launay, Quentin Malartic, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. 2023. The falcon series of language models: Towards open frontier models.

Hritik Bansal, Karthik Gopalakrishnan, Saket Dingliwal, Sravan Bodapati, Katrin Kirchhoff, and Dan Roth. 2023. Rethinking the role of scale for in-context learning: An interpretability-based case study at 66 billion scale. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11833–11856, Toronto, Canada. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. 2019. Once-for-all: Train one network and specialize it for efficient deployment. *arXiv preprint arXiv:1908.09791*.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. 2023. Accelerating large language model decoding with speculative sampling.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*.

Ali Edalati, Marzieh Tahaei, Ivan Kobyzev, Vahid Partovi Nia, James J Clark, and Mehdi Rezagholizadeh. 2022. Krona: Parameter efficient tuning with kronecker adapter. *arXiv preprint arXiv:2212.10650*.

Angela Fan, Edouard Grave, and Armand Joulin. 2019. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Lu Hou, Zhiqi Huang, Lifeng Shang, Xin Jiang, Xiao Chen, and Qun Liu. 2020. Dynabert: Dynamic bert with adaptive width and depth. *Advances in Neural Information Processing Systems*, 33:9782–9793.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, Vancouver, Canada. Association for Computational Linguistics.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient low-rank hypercomplex adapter layers. *Advances in Neural Information Processing Systems*, 34:1022–1035.

OpenAI. 2023. Gpt-4 technical report.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2020. Adapterfusion: Non-destructive task composition for transfer learning. *arXiv preprint arXiv:2005.00247*.

Gabriele Prato, Ella Charlaix, and Mehdi Rezagholizadeh. 2019. Fully quantized transformer for machine translation. *arXiv preprint arXiv:1910.10485*.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

Tianxiang Sun, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. 2022. Black-box tuning for language-model-as-a-service. In *International Conference on Machine Learning*, pages 20841–20855. PMLR.

Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. 2022. Lst: Ladder side-tuning for parameter and memory efficient transfer learning. *Advances in Neural Information Processing Systems*, 35:12991–13005.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. 2022. Dylora: Parameter efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. *arXiv preprint arXiv:2210.07558*.

Mojtaba Valipour, Mehdi Rezagholizadeh, Hossein Rajabzadeh, Marzieh Tahaei, Boxing Chen, and Ali Ghodsi. 2023. Sortednet, a place for every network and every network in its place: Towards a generalized solution for training many-in-one neural networks. *arXiv preprint arXiv:2309.00255*.

Yidong Wang, Zhuohao Yu, Zhengran Zeng, Linyi Yang, Cunxiang Wang, Hao Chen, Chaoya Jiang, Rui Xie, Jindong Wang, Xing Xie, et al. 2023. Pandalm: An automatic evaluation benchmark for llm instruction tuning optimization. *arXiv preprint arXiv:2306.05087*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language model with self generated instructions. *arXiv preprint arXiv:2212.10560*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Minghao Wu, Abdul Waheed, Chiyu Zhang, Muhammad Abdul-Mageed, and Alham Fikri Aji. 2023. Lamini-lm: A diverse herd of distilled models from large-scale instructions. *arXiv preprint arXiv:2304.14402*.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. 2020. Deebert: Dynamic early exiting for accelerating bert inference. *arXiv preprint arXiv:2004.12993*.

Biao Zhang and Rico Sennrich. 2019. Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32.

Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. 2023. A survey on model compression for large language models. *arXiv preprint arXiv:2308.07633*.

| Method | Avg Time per Epoch (s) | Avg Memory Usage per Epoch (MB) |
|---|---|---|
| SFT | 25,765.95 | 99,168 |
| SoFT | 25,269.87 (0.98×) | 125,682 |

Table 3: Training Time and Memory Usage comparison of SoFT and SFT on Alpaca dataset.

# A Appendix

## A.1 Computational Overhead of SoFT

Given the nested pattern of sub-models and the fact that we share the language model head across sub-models, we do not expect to see any computation overhead for SoFT versus SFT. To validate this claim, we compared SoFT and SFT regarding training time and memory usage in our experiment on the Alpaca dataset (Table 3). Here is the result for two main experiments of SoFT and SFT. As expected, training with SoFT leads to equal training time compared to SFT. During training, SoFT has about 25% memory overhead in PyTorch compared to SFT, which only provides a single full model at the end.

## A.2 Additional Experiments

Table 4 shows the detailed results of the Sorted LLaMA and SFT performance on the PandaLM benchmark in different setup in equal training time (2 Epochs for both SFT and SoFT). As we can see, sorted sub-models outperform their SFT counterparts (and even higher sub-models) , while in SFT+ICT (Early-Exit), as we go higher in sub-models (e.g. layer 36), we can see a noticeable improvement in the performance compared to the SFT. This can demonstrate the importance of tuning the language model classifier in improving text generation capability in the latest layers in the standard fine-tuning format.

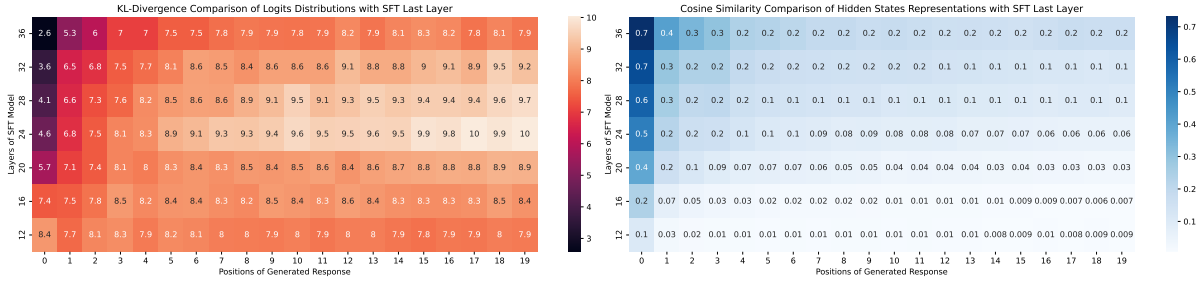
Table 5 shows the SoFT and SFT comparison in a different training time setup in which SoFT has access to doubled training time (4 Epochs). Results show that Sorted LLaMA can outperform standard fine-tuned LLaMA further by continuing the SoFT process. The improvement in Sorted LLaMA sub-models performance can be observed specifically in intermediate layers.

## A.3 Analysis

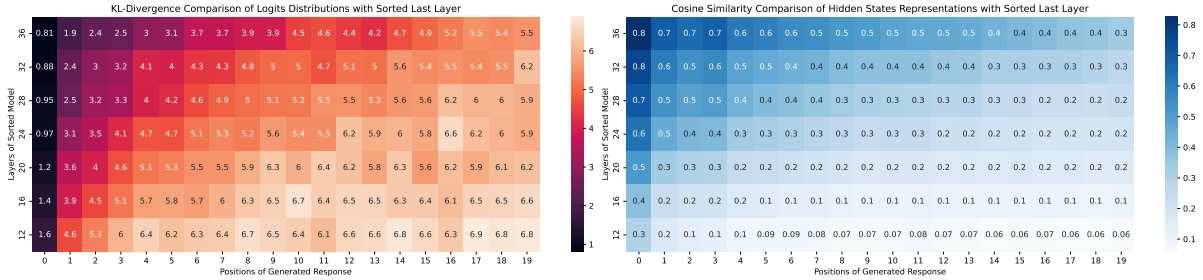
Table 6 and 7 show some samples generated by sub-models of LLaMA (SFT+ICT) and SoFT on PandaLM evaluation set. In the first query of Table 6, LLaMA sub-models until layer 36 struggle to generate relevant responses about books in the Crime and Mystery genre. While Sorted LLaMA sub-models start to address the related novels from layer 24. The second query in the table is a simpler instruction, which is a multi-label classification problem. Again Sorted LLaMA sub-models start to generate the correct label in much earlier layers (layer 20) compared to the LLaMA sub-models (layer 24). Table 7 first example shows the performance gap of the LLaMA and Sorted LLaMA intermediate sub-models even in a more severe case. To write a review about a restaurant with certain aspects, LLaMA sub-models before layer 32 hallucinate or generate gibberish, while Sorted LLaMA starts to generate a complete review addressing key points mentioned in the instruction even in the first sub-model (layer 16). In the second example, the same pattern occurs where SoFT sub-models can generate meaningful response starting from layer 16 while LLaMA first reasonable text happens at layer 36.

Table 8 shows an example of SFT and SoFT performance on TriviaQA benchmark. While LLaMA struggles to generate single answer token even in the sub-models close to the last layer, SoFT could transfer the question answering ability until sub-layer 20 and generate the correct final answer.

After all, Sorted LLaMA sub-models demonstrate the ability to generate more comprehensive (Table 6 example 1 and Table 7 example 1) and informative (Table 6 example 2) answers in earlier layers compared to LLaMA. Based on our observation, LLaMA sub-models mostly tend to generate irrelevant or even gibberish in earlier blocks (layers 12 to 24), while the generated texts by Sorted LLaMA exhibit sufficient learned information to answer the input instruction despite having much fewer parameters.




(a) LLaMA sub-models vs LLaMA Last Layer




(b) SortedLLaMA sub-models - SortedLLaMA Last Layer

Figure 5: A comparison of sub-models based on output logits and hidden state cosine similarity.

| Sorted LLaMA/LLaMA | 12 (4.1B) | 16 (5.4B) | 20 (6.6B) | 24 (7.9B) | 28 (9.2B) | 32 (10.4B) | 36 (11.7B) | 40 (13B) |
|---|---|---|---|---|---|---|---|---|
| SoFT vs. SFT | | | | | | | | |
| 12 (4.1B) | 71.0/99.0/0.0 | 97.5/72.5/0.0 | 129.0/41.0/0.0 | 131.0/39.0/0.0 | 121.5/48.5/0.0 | 106.5/63.5/0.0 | 45.0/125.0/0.0 | 17.0/152.5/0.5 |
| 16 (5.4B) | 81.0/89.0/0.0 | 101.5/68.5/0.0 | 128.5/40.5/1.0 | 131.5/38.0/0.5 | 124.0/44.5/1.5 | 114.0/54.0/2.0 | 52.0/114.0/4.0 | 18.0/146.0/6.0 |
| 20 (6.6B) | 111.5/58.5/0.0 | 132.0/38.0/0.0 | 144.5/23.5/2.0 | 147.5/20.5/2.0 | 141.5/24.0/4.5 | 132.5/30.5/7.0 | 73.5/85.5/11.0 | 32.5/114.0/23.5 |
| 24 (7.9B) | 124.5/45.5/0.0 | 136.5/33.5/0.0 | 150.0/18.0/2.0 | 154.5/13.5/2.0 | 148.0/18.5/3.5 | 144.5/19.0/6.5 | 98.0/62.0/10.0 | 44.5/90.0/35.5 |
| 28 (9.2B) | 125.5/44.5/0.0 | 145.0/25.0/0.0 | 153.0/15.0/2.0 | 153.5/14.5/2.0 | 148.0/16.5/5.5 | 143.5/20.5/6.0 | 96.5/59.5/14.0 | 45.0/89.0/36.0 |
| 32 (10.4B) | 141.5/28.5/0.0 | 152.0/18.0/0.0 | 159.0/9.0/2.0 | 160.0/8.0/2.0 | 152.0/12.5/5.5 | 150.5/13.5/6.0 | 108.5/45.0/16.5 | 55.5/75.0/39.5 |
| 36 (11.7B) | 141.0/28.5/0.5 | 152.5/17.0/0.5 | 159.0/8.5/2.5 | 161.5/6.5/2.0 | 150.0/14.5/5.5 | 148.5/15.5/6.0 | 112.0/42.5/15.5 | 53.0/66.0/51.0 |
| 40 (13B) | 143.5/26.5/0.0 | 156.0/14.0/0.0 | 160.5/7.5/2.0 | 161.0/7.0/2.0 | 150.0/14.0/6.0 | 150.0/13.5/6.5 | 115.5/39.0/15.5 | 52.5/62.5/55.0 |
| SoFT vs. SFT+ICT(Early-Exit) | | | | | | | | |
| 12 (4.1B) | 75.0/95.0/0.0 | 108.5/61.5/0.0 | 128.5/41.5/0.0 | 122.5/47.5/0.0 | 116.5/53.5/0.0 | 91.0/79.0/0.0 | 37.5/131.5/1.0 | 17.0/152.5/0.5 |
| 16 (5.4B) | 86.5/82.5/1.0 | 113.0/57.0/0.0 | 127.0/41.0/2.0 | 122.0/47.0/1.0 | 117.5/50.5/2.0 | 94.5/72.0/3.5 | 36.0/129.0/5.0 | 18.0/146.0/6.0 |
| 20 (6.6B) | 111.5/57.5/1.0 | 137.0/33.0/0.0 | 143.5/24.0/2.5 | 143.0/23.0/4.0 | 137.0/27.0/6.0 | 122.0/38.0/10.0 | 60.0/94.5/15.5 | 32.5/114.0/23.5 |
| 24 (7.9B) | 126.5/42.5/1.0 | 144.0/26.0/0.0 | 149.0/19.5/1.5 | 151.0/15.5/3.5 | 143.0/21.5/5.5 | 133.5/28.0/8.5 | 76.5/72.5/21.0 | 44.5/90.0/35.5 |
| 28 (9.2B) | 130.0/39.0/1.0 | 147.0/23.0/0.0 | 153.5/15.5/1.0 | 150.0/16.0/4.0 | 143.5/18.5/8.0 | 131.0/29.0/10.0 | 79.0/66.0/25.0 | 45.0/89.0/36.0 |
| 32 (10.4B) | 141.5/27.5/1.0 | 155.5/14.5/0.0 | 161.0/8.0/1.0 | 157.0/8.5/4.5 | 151.0/11.0/8.0 | 143.5/15.0/11.5 | 89.5/49.5/31.0 | 55.5/75.0/39.5 |
| 36 (11.7B) | 143.0/25.5/1.5 | 156.5/13.0/0.5 | 160.0/8.5/1.5 | 157.0/8.5/4.5 | 148.0/14.0/8.0 | 142.5/16.5/11.0 | 92.5/46.5/31.0 | 53.0/66.0/51.0 |
| 40 (13B) | 146.0/23.0/1.0 | 157.0/13.0/0.0 | 160.5/7.5/2.0 | 157.5/9.0/3.5 | 149.0/14.0/7.0 | 143.5/16.0/10.5 | 97.5/43.5/29.0 | 52.5/62.5/55.0 |

Table 4: Pair-wise comparison for different layers (sub-models) in Standard Fine-Tuning and SoFT at equal training cost (2 Epochs). Each cell consists of three values: Wins, Losses, Ties. Wins demonstrate the number of times that the generated text of the sub-model in row (sorted) is preferred to the sub-model in column (Fine-Tuned) and Losses is the opposite. Numbers are average of two separate experiments with different order of inputs to evaluator in order to neutralize the order bias.

| Sorted LLaMA/LLaMA | 12 (4.1B) | 16 (5.4B) | 20 (6.6B) | 24 (7.9B) | 28 (9.2B) | 32 (10.4B) | 36 (11.7B) | 40 (13B) |
|---|---|---|---|---|---|---|---|---|
| SoFT vs. SFT | | | | | | | | |
| 12 (4.1B) | 88.5/81.5/0.0 | 108.0/62.0/0.0 | 134.5/35.5/0.0 | 135.0/35.0/0.0 | 129.0/41.0/0.0 | 120.0/49.0/1.0 | 57.0/109.5/3.5 | 23.5/144.0/2.5 |
| 16 (5.4B) | 106.5/63.0/0.5 | 120.0/50.0/0.0 | 140.0/29.0/1.0 | 144.5/24.5/1.0 | 142.0/26.5/1.5 | 136.0/32.0/2.0 | 70.0/95.0/5.0 | 34.5/124.5/11.0 |
| 20 (6.6B) | 127.0/43.0/0.0 | 138.5/31.5/0.0 | 151.5/16.5/2.0 | 152.0/17.0/1.0 | 143.5/23.5/3.0 | 144.0/21.5/4.5 | 94.5/67.5/8.0 | 47.0/99.5/23.5 |
| 24 (7.9B) | 138.5/31.5/0.0 | 149.5/20.5/0.0 | 159.0/9.0/2.0 | 158.0/10.5/1.5 | 151.5/13.5/5.0 | 149.0/15.5/5.5 | 107.0/49.5/13.5 | 53.0/81.0/36.0 |
| 28 (9.2B) | 137.0/33.0/0.0 | 149.0/21.0/0.0 | 158.0/10.0/2.0 | 159.5/8.5/2.0 | 150.0/15.0/5.0 | 149.5/15.0/5.5 | 107.0/47.5/15.5 | 50.5/78.0/41.5 |
| 32 (10.4B) | 146.0/24.0/0.0 | 157.0/13.0/0.0 | 163.0/5.0/2.0 | 163.0/5.0/2.0 | 154.5/10.5/5.0 | 151.5/12.5/6.0 | 117.5/37.5/15.0 | 63.5/62.0/44.5 |
| 36 (11.7B) | 149.5/20.5/0.0 | 160.0/10.0/0.0 | 164.0/4.0/2.0 | 162.5/5.5/2.0 | 157.5/7.5/5.0 | 154.0/10.0/6.0 | 119.5/34.5/16.0 | 62.5/60.0/47.5 |
| 40 (13B) | 153.5/16.5/0.0 | 163.0/7.0/0.0 | 165.5/3.0/1.5 | 163.5/4.5/2.0 | 157.0/8.0/5.0 | 156.0/8.5/5.5 | 121.0/33.5/15.5 | 67.5/52.0/50.5 |
| SoFT vs. SFT+ICT(Early-Exit) | | | | | | | | |
| 12 (4.1B) | 91.5/77.5/1.0 | 123.5/46.5/0.0 | 138.5/31.5/0.0 | 134.0/36.0/0.0 | 130.5/39.0/0.5 | 107.5/59.0/3.5 | 46.0/120.5/3.5 | 23.5/144.0/2.5 |
| 16 (5.4B) | 106.5/63.5/0.0 | 128.5/41.0/0.5 | 145.0/24.0/1.0 | 144.5/25.0/0.5 | 139.0/29.5/1.5 | 122.5/43.0/4.5 | 55.5/106.5/8.0 | 34.5/124.5/11.0 |
| 20 (6.6B) | 128.0/40.5/1.5 | 142.0/27.5/0.5 | 152.5/16.0/1.5 | 148.0/19.0/3.0 | 142.0/22.5/5.5 | 131.0/31.0/8.0 | 74.0/79.0/17.0 | 47.0/99.5/23.5 |
| 24 (7.9B) | 140.5/28.5/1.0 | 155.0/15.0/0.0 | 159.0/10.0/1.0 | 156.0/10.5/3.5 | 149.5/12.5/8.0 | 141.5/19.0/9.5 | 90.0/54.5/25.5 | 53.0/81.0/36.0 |
| 28 (9.2B) | 142.0/27.0/1.0 | 155.5/14.5/0.0 | 161.5/7.5/1.0 | 155.5/10.5/4.0 | 148.5/13.0/8.5 | 141.0/19.0/10.0 | 91.5/53.5/25.0 | 50.5/78.0/41.5 |
| 32 (10.4B) | 150.0/19.0/1.0 | 161.0/9.0/0.0 | 161.5/7.5/1.0 | 160.0/6.0/4.0 | 153.0/8.5/8.5 | 145.0/14.0/11.0 | 99.0/41.0/30.0 | 63.5/62.0/44.5 |
| 36 (11.7B) | 151.0/18.0/1.0 | 163.5/6.5/0.0 | 165.0/3.5/1.5 | 161.0/5.0/4.0 | 155.0/7.5/7.5 | 147.5/12.0/10.5 | 105.5/36.0/28.5 | 62.5/60.0/47.5 |
| 40 (13B) | 156.0/13.0/1.0 | 165.5/4.5/0.0 | 165.0/3.5/1.5 | 161.0/5.0/4.0 | 155.0/7.5/7.5 | 150.5/10.0/9.5 | 106.0/35.0/29.0 | 67.5/52.0/50.5 |

Table 5: Pair-wise comparison for different layers (sub-models) in Standard Fine-Tuning and SoFT (4 Epochs) after tuning Language Model classifier for each SFT sub-model.

**Query**: Choose the best books from the given genre. Crime & Mystery

| Layer | LLaMA Response (SFT + ICT) | Sorted LLaMA Response (SoFT) |
|---|---|---|
| 12 | Therewiguptaranapped byoremandoesleven years ago agomairiensum... | Here are some of the best books from the genre: 1. The Fear of the unknown: The Fist of the Darkness2. The Fist of the Deaths of the Deaths of the Deaths of the Deaths of the Deaths of the Deaths ... |
| 16 | Here list list list!!!!!!!!!!!!!!!!00e4ndersonally preferences personal ... | Here is a list of top 10 books from the genre Crime & Mystery that you can read for your next reading book list: 1. The Great mystery of the Mystery of the Mystery of the Mystery of the Mystery of the ... |
| 20 | Here listicle lists some best books genreed intoxicatedly intoxicatedly ... | Here are some popular and well-known crime and mystery books that you can choose from: 1. The Secret of the Mystery solved by the detective by Jameson Jameson is a classic crime and mystery novel that is widely regarded as one of the best crime and mystery books ever written. 2. The Secret of the Mystery solved by the detective by Jameson Jameson is a classic crime and mystery novel that is widely regarded as one of the best crime and mystery books ever written. 3. The Secret of the Mystery solved ... |
| 24 | Here list contains some best books belonging belongs genre crime & mystery genre ordered alphabetically alphabetically ... | Here are some of the best crime and mystery books: 1. The Hatchery by Stephen King - This book is a classic horror/ crime thriller that follows the story of a group of friends who discover a hidden hatchery of horrors. 2. The Last Thistle by Murphy Murphy - This book is a thriller that follows the investigation of a murder in a small town. The story is filled with suspense and twists that will keep you hooked. 3. The Last Thistle by Murphy Murphy - This book is a thriller that follows the investigation of a murder in a small town. The story is filled with suspense and twists that will keep you hooked. 4. The Last Thistle by Murphy ... |
| 28 | crime mystery books best crime mystery books crime mystery books best crime mystery ... | Here are some of the best crime and mystery books that you might enjoy: 1. The Great Gatsby by F. Scott Fitzgerald - This classic novel is a classic tale of love, betrayal, and the dark side of the American Dream. 2. The Great Gatsby by F. Scott Fitzgerald - This classic novel is a classic tale of love, betrayal, and the dark side of the American Dream. 3. The Great Gatsby ... |
| 32 | Crime mystery genre has many great books to choose from. Here are some recommendations: crime mystery books recommended by readers: 1) crime mystery books recommended by readers: Crime mystery genre has many great books to choose from. Here are some recommendations: crime mystery books ... | Crime & Mystery is a popular genre, and there are many great books that fall into this category. Here are some of the best crime and mystery books that have stood the test of time and are widely acclaimed by readers and critics alike: 1. The Hound of the Baskervilles by Sir Arthur Conan Doyle: This classic detective novel features the iconic detective Sherlock Holmes and his sidekick, Dr. John Watson. The story follows Holmes as he investigates the mysterious death of a wealthy man and the disappearance of his family. 2. The Silent Patient by Sarah C. Dessen: This psychological thriller follows the story of a woman who is institutionized after shooting her husband. The novel explores themes of trauma, mental illness, and the power of love to heal. 3. The Silent majority by Agatha Christie: This classic mystery novel features the beloved detective Hercule Poirot as he investigates the murder of a wealthy man. The novel is full of twists and turns, and the identity of the murderer is not revealed until the end. |
| 36 | Crime fiction is a genre that has captivated readers for centuries with intriguvity and suspenseful plots. Here are some of the best crime books that you should definitely read: 1. Sherlock Holmes series by Sir Arthur Conan Doyle: regarded as one of the greatest fictional detectives of all time, Sherlock Holmes is known for his deductive reasoning and ability to solve complex cases. 2. Agatha Christie novels: regarded as the queen of crime fiction, Agatha Christie wrote numerous novels and short stories that are still widely read today. Some of her most famous works include Hercules Poirot series and Miss Marple series. Raymond Chandler novels: regarded as one of the greatest crime writers of all time, Raymond Chandler wrote novels set in Los Angeles featuring detective Philip Marlowe. Some of his most famous works include T̈he Big Sleepänd T̈he Long Goodbye. 4. crime novels by crime novelist Raymond Chandler: regarded as one of the greatest crime writers of all time, Raymond Chandler wrote novels set in Los Angeles featuring detective Philip Marlowe. Some of his most famous | Based on your preference for crime and mystery books, here are some highly recommended titles: 1. To Kill a Mockingbird by Harper Lee: This classic novel explores themes of racism, prejudice, and injustice in the Deep South during the 1930s. 2. The Silent Patient by Alex Michaelides: A psychological thriller about a woman who shoots her husband and then refuses to speak. 3. The Girl with the Dragon Tattoo by Stieg Larsson: This novel kicks off the Millennium Trilogy, following the investigation of a missing journalist and the mysterious disappearance of a young woman. 4. The Da Vinci Code by Dan Brown: This fast-paced thriller follows the trail of a centuries-old conspiracy involving the Catholic Church, the Knights Templar, and the secrets of the Da Vinci Code. 5. The Girl with the Dragon Tattoo by Stieg Larsson: This novel kicks off the Millennium Trilogy, following the investigation of a missing journalist and the mysterious |
| 40 | Here are some of the best books in the Crime and Mystery genre: 1. The Hound of the Baskervilles by Sir Arthur Conan Doyle 2. The Maltese Falcon by Dashiell Hammett 3. The Big Sleep by Raymond Chandler 4. The Adventures of Sherlock Holmes by Sir Arthur Conan Doyle 5. The Thin Man by Dashiell Hammett 6. The Murder of Roger Ackroyd by Agatha Christie 7. The Silence of the Lambs by Thomas Harris 8. The Red House Mystery by A.A. Milne 9. The Secret of the Old Clock by Carolyn Keene 10. The Adventures of the Black Stallion by Walter Farley. | Here are some highly recommended crime and mystery books: 1. To Kill a Mockingbird by Harper Lee 2. The Hound of the Baskervilles by Sir Arthur Conan Doyle 3. The Silence of the Lambs by Thomas Harris 4. The Great Gatsby by F. Scott Fitzgerald 5. The Catcher in the Rye by J.D. Salinger 6. The Da Vinci Code by Dan Brown 7. The Girl with the Dragon Tattoo by Stieg Larsson 8. The Adventures of Sherlock Holmes by Sir Arthur Conan Doyle 9. The Secret of the Old Clock by Carolyn Keene 10. The Help by Kathryn Stockett. |

**Query**: Classify the questions in the FAQ into Finding policy details, Change account settings, Filing a claim and viewing status, or Cancelling coverage.
Could you deposit money into my account rather than mailing me a physical cheque?

| Layer | LLaMA Response (SFT + ICT) | Sorted LLaMA Response (SoFT) |
|---|---|---|
| 12 | Yes sir/ sorry sorryonymouslyovoyedance Festivalandeiamot... | The question is not classified as a request or request for information. It is a question that is addressed to the customer's request. |
| 16 | belongs belongsfrontieralandersonallyi... | The given question is a question of financial accounting and it belongs to the Cancelling coverage category. |
| 20 | belongs category c̈hangeable via email or phone call or online portal or phone contactless payment etceterially... | Change account settings |
| 24 | Change account settings | Change account settings |
| 28 | This question falls category of Change account settings. | Change account settings |
| 32 | Change account settings | Change account settings |
| 36 | Change account settings | Change account settings |
| 40 | Change account settings | Change account settings |

Table 6: Generated responses in different layers of SFT (Early Exit) and Sorted LLaMA for two samples from PandaLM validation set.

**Query**: Based on the given keywords, write a review of a restaurant. Name: Pizzeria
good prices, quiet place, tasty, nice staff

| Layer | LLaMA Response (SFT + ICT) | Sorted LLaMA Response (SoFT) |
|---|---|---|
| 12 | Hinweis BÖ0e9jar Brothersqueeries... | The restaurant is located in a small town, with an impressive list of options for meals and drinks. The atmosphere is comfortable and comfortable, with a small, large-scale room, with a large enough room for comfortable comfortable rooms. The menu is a great way to choose the best option for meals and drinks. The menu is easy to understand and can be easily used to find the best option for meals and drinks. The menu is also available online and can be easily accessed by anyone who is interested in making a purchase. The menu is also made up of a variety of cuisine, with a variety of cuisine. The menu is also made up of a variety of cuisine... |
| 16 | Review: description description Cassandra Smithsoniusque ... | The Pizzeria is a great place to eat and enjoy a relaxing meal. It is a quiet place, so it is perfect for a quiet meal and a relaxing meal. The food is incredibly delicious, and the staff is friendly and friendly. The prices are great, and the atmosphere is comfortable and relaxing. It is a great place to eat and enjoy a relaxing meal. |
| 20 | Review: visited recently visited recently visited recently visited recently... | I recently visited Pizzeria, a restaurant that is both affordable and a quiet place to enjoy a meal. The prices are definitely good, and the food is definitely tasteful. The staff is nice and friendly, making the experience even more enjoyable. Overall, I highly recommend Pizzeria for anyone looking for a good meal at a reasonable price. |
| 24 | Review: visited recently visited recently visited recently visited recently ... | I recently visited Pizzeria, and I must say, it was a pleasant experience. The prices were reasonable, and the atmosphere was quiet and relaxing. The staff was friendly and helpful, making me feel welcome and comfortable. The pizza was tasty and made me feel satisfied after a long day. Overall, I would recommend Pizzeria to anyone looking for a good pizza, good prices, and a nice atmosphere |
| 28 | Revieweddit rating: stars star star star ... | Pizzeria is a hidden gem in the heart of the city. The prices are reasonable and the atmosphere is a refreshing change from the crowded restaurants. The staff is friendly and welcoming, making the experience even more enjoyable. The pizza is tasty and the crust is crisp and crunchy. The sauce is rich and the cheese is melted perfectly. Overall, Pizzeria is a must-visit for anyone looking for a good pizza and a quiet place to enjoy it. |
| 32 | Review: visited this place with friends recently after hearing good reviews about their pizza pie. Upon arrival we noticed how quiet it was inside which made it perfect place to relax after hectic weekdays. prices were reasonable considering quality of food served. staff members were friendly and helpful throughout entire duration of visit. overall experience exceed expectations thanks to delicious food served hot freshly baked pizza pie tasteful inguncticns of Italian cu cu cu cu cu cu cu cu cu cu cu ... | I recently visited Pizzeria and I must say, it was a pleasant experience. The prices were reasonable, and the atmosphere was quiet and serene. The staff was friendly and helpful, making me feel welcome. The pizza was delicious, with a crisp crust and a delicious, creamy sauce. Overall, I would definitely recommend Pizzeria to anyone looking for a good meal in a quiet, relaxing setting. |
| 36 | Review of Pizzeria: upon walking into pizzeria, you are greetvity welcomed by nice staff members who are always ready to assist you with your needs. Their prices are reasonable considering the quality of food they offer. Their pizza is mouthwateringly delicious and always freshly made to order. Additionally, pizzeria is a quiet place perfect for dates or family gatherings. Overall, pizzeria is a great place to visit for delicious food and great service. highly recommended! | I recently visited Pizzeria and had a wonderful experience. The prices were reasonable and the atmosphere was quiet and relaxing. The food was delicious and the staff was friendly and attentive. Overall, I would highly recommend Pizzeria to anyone looking for a tasty meal in a nice, quiet setting. |
| 40 | Pizzeria is a great place to go for a quiet meal with tasty food and friendly staff. The prices are good, making it a great value for your money. The atmosphere is cozy and the service is prompt. The pizza is cooked to perfection and the toppings are fresh and delicious. Overall, it's a great place to enjoy a meal with friends or family. Highly recommended! | Pizzeria is a great place to enjoy a delicious pizza in a quiet and cozy atmosphere. The prices are reasonable and the staff is friendly and accommodating. The pizzas are cooked to perfection and the toppings are fresh and tasty. The restaurant is perfect for a casual meal with friends or family, and the ambiance is relaxing and inviting. Overall, Pizzeria is a great option for a satisfying pizza experience. |

**Query**: Define what the underlined word means for kids. _keep a promise

| Layer | LLaMA Response (SFT + ICT) | Sorted LLaMA Response (SoFT) |
|---|---|---|
| 12 | Thereuh Scientificallyally compromiseable ... | The word machery is a word that is used to describe a person or a person's behavior, or a person's behavior in a particular manner. It means that someone has been able to successfully successfully deal with the situation or situation situationally. |
| 16 | means something importantlyrics song songwriter | The word Kick A promise is a common term in the context of a commitment to deliver something or to take action. It means that someone has taken a promise to make a promise to make sure that they are doing something or fulfilling their promise. It is a kind of commitment that is often used in personal and professional settings. |
| 20 | When someone promise means promise means promise means promise means promise means ... | Keeping a promise means keeping a promise made to someone else, and not making a promise to someone else or even yourself that you will not do something, but actually doing it. It means being kind, honesty, and kindness towards others. It means being kind to your parents, siblings, friends, and even your peers. It means being kind to yourself too. It means being kind to yourself ... |
| 24 | When someone promise promise means promise promise means promise ... | Keep a promise means to make sure that you do what you promised to do. It means that you are reliable and reliable people are people who are trusted and respected. When you promise something to someone, it means that you are committed to it and you will do it. |
| 28 | keep promise means promise promise means promise promise means promise promise ... | Keeping a promise means to do what you promised to do, no matter what. It means keeping your word and keeping your word is important. It's important to keep promises because it helps people trust you and trust that you will do what you say you will do. |
| 32 | When you promise something means prom promprompr... | Keeping a promise means that you promise something to someone and you do it. It means that you do not break your word and you do what you said you would do. It is important to keep promises because it shows that you are trustworthy and reliable. |
| 36 | Keeping a promise means promising something to someone else or yourself, and then doing your best to make sure that promise comes true. It means being honest and keeping your word even when it'vity hard or takes effort. It means being responsible and reliable so people can trust you to do what you say you will do. | Keeping a promise means doing what you said you would do. It's important to follow through on your promises to others and to be reliable. |
| 40 | Keep a promise means to do what you said you would do. It means to be honest and to keep your word. When you keep a promise, you show that you are trustworthy and that people can count on you. | Keeping a promise means doing what you said you would do, even if it's hard or takes a long time. It's important to be honest and reliable, and to follow through on your words. |

Table 7: Generated responses in different layers of SFT (Early Exit) and Sorted LLaMA for a sample from PandaLM validation set.

| **Query**: Bandar Seri Begawan International airport is in which country? | | |
|---|---|---|
| **Layer** | **LLaMA Response (SFT)** | **Sorted LLaMA Response (SoFT)** |
| 12 | iernohotter than than ... | Malta |
| 16 | Nederlige territ territorially ... | Burma (Myanmar) |
| 20 | Australia AustraliaAustral Australia Australia ... | Brunei |
| 24 | Malays Malays Malays Malays ... | Brunei |
| 28 | Malays Malays Malays Malays ... | Brunei |
| 32 | Brunei | Brunei |
| 36 | Brunei | Brunei |
| 40 | Brunei | Brunei |

Table 8: Generated responses in different layers of SFT and SoFT for a sample from TriviaQA benchmark.

| Sorted/Fully Fine-tuned | 12 (4.1B) | 20 (6.6B) | 28 (9.2B) | 36 (11.7B) |
|---|---|---|---|---|
| 2 SFT Epochs/2 SoFT Epochs | | | | |
| 12 (4.1B) | 80.0/88.5/1.5 | 37.5/132.0/0.5 | 28.0/141.5/0.5 | 20.0/148.5/1.5 |
| 16 (5.4B) | 88.5/77.0/4.5 | 42.0/121.5/6.5 | 31.5/135.0/3.5 | 20.0/142.5/7.5 |
| 20 (6.6B) | 114.0/48.5/7.5 | 56.0/84.5/29.5 | 42.5/108.0/19.5 | 32.0/117.5/20.5 |
| 24 (7.9B) | 123.0/37.0/10.0 | 70.5/61.5/38.0 | 53.5/80.0/36.5 | 45.5/89.5/35.0 |
| 28 (9.2B) | 131.0/32.0/7.0 | 75.0/63.0/32.0 | 56.0/70.5/43.5 | 46.5/82.5/41.0 |
| 32 (10.4B) | 143.5/21.0/5.5 | 98.0/43.5/28.5 | 73.0/54.0/43.0 | 54.0/65.5/50.5 |
| 36 (11.7B) | 140.5/22.0/7.5 | 98.5/40.5/31.0 | 76.0/49.0/45.0 | 53.0/62.5/54.5 |
| 40 (13B) | 137.5/24.0/8.5 | 102.0/37.0/31.0 | 78.5/45.5/46.0 | 55.0/62.0/53.0 |
| 2 SFT Epochs/4 SoFT Epochs | | | | |
| 12 (4.1B) | 94.5/71.0/4.5 | 44.0/121.0/5.0 | 37.0/130.5/2.5 | 26.5/138.5/5.0 |
| 16 (5.4B) | 105.0/60.0/5.0 | 55.0/102.0/13.0 | 51.0/110.5/8.5 | 34.0/123.0/13.0 |
| 20 (6.6B) | 129.5/33.5/7.0 | 73.0/67.5/29.5 | 58.5/85.0/26.5 | 47.0/96.5/26.5 |
| 24 (7.9B) | 132.0/30.5/7.5 | 89.5/51.0/29.5 | 70.0/62.5/37.5 | 51.0/80.0/39.0 |
| 28 (9.2B) | 140.0/23.5/6.5 | 89.5/51.0/29.5 | 66.5/60.0/43.5 | 48.5/77.5/44.0 |
| 32 (10.4B) | 144.5/18.5/7.0 | 103.5/35.0/31.5 | 77.5/52.0/40.5 | 55.5/62.0/52.5 |
| 36 (11.7B) | 146.0/17.5/6.5 | 105.5/34.5/30.0 | 84.5/44.5/41.0 | 60.0/52.5/57.5 |
| 40 (13B) | 149.0/15.0/6.0 | 105.0/37.5/27.5 | 87.5/41.5/41.0 | 62.5/53.5/54.0 |

Table 9: Pair-wise comparison between Extracted fine-tuned and SoFT sub-models.