Pruning General Large Language Models into Customized Expert Models

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have transformed natural language processing, yet their substantial model sizes often demand significant computational resources. To preserve computing resources and accelerate inference speed, it is crucial to prune redundant parameters, especially for experienced users who often need expert models tailored to specific downstream scenarios. However, current pruning methods primarily focus on maintaining models' general capabilities, either requiring extensive post-training or performing poorly due to coarse-grained pruning. In this work, we design a Custom Pruning method (`Cus-Prun`) to prune a large general model into a smaller lightweight expert model, which is positioned along the "language", "domain" and "task" dimensions. By identifying and pruning irrelevant neurons of each dimension, `Cus-Prun` creates expert models without any post-training. Our experiments demonstrate that `Cus-Prun` consistently outperforms other methods, achieving minimal loss in both expert and general capabilities across various models from different model families and sizes.

1 Introduction

Large language models (LLMs) (Achiam et al., 2023; Reid et al., 2024; Dubey et al., 2024; Team et al., 2024) have revolutionized the field of natural language processing (NLP), emerging as powerful tools with widespread applications across various languages (Cui et al., 2023; Yang et al., 2024a), domains (Li et al., 2023a; Roziere et al., 2023; Li et al., 2023b), and tasks (Azerbayev et al., 2024; Alves et al., 2024). However, the impressive performance of LLMs often comes at the cost of immense model sizes, mostly containing billions of parameters and thus demand significant computing resources (Goldstein et al., 2023; Musser, 2023). To address this issue, researchers have recently proposed various model pruning methods for LLMs. These methods aim to reduce model parameters while maintaining the overall performance through techniques such as removal of unimportant structures (Ma et al., 2023; Men et al., 2024; Song et al., 2024), matrix approximation (Sharma et al., 2024; Ashkboos et al., 2024), and extensive post-training after pruning (Wang et al., 2024; Xia et al., 2024).

Most existing pruning methods focus on preserving the *general capabilities* of the model, often evaluated by broad-spectrum benchmarks like MMLU (Hendrycks et al., 2021). While aiming for overall versatility, they may not align well with real-world user needs for a pruned small model, which are usually more *specific and targeted*. For instance, a user might require a question-answering model tailored specifically for the education domain in German. Such specialized request in fact aligns well with the fundamental motivation behind pruning: to create a smaller model by eliminating unnecessary parameters. In this context, the notion of "unnecessary" parameters becomes more precise—referring to those parameters that are irrelevant to the particular use case. By selectively pruning these redundant parameters, one can construct a smaller, expert model that is better aligned with specific requirements.

However, current pruning techniques focusing on preserving the general capabilities of LLMs often employ coarse-grained approached such as removing entire layers or modules (Li et al., 2022; Kurz et al., 2024; Huang et al., 2024). Therefore, it may remove parameters that are critical for specialized downstream scenarios, and thus sometimes require extensive post-training to recover the pruned capabilities (Xia et al., 2024; Muralidharan et al., 2024). On the other hand, the preserved certain capabilities may not be useful or relevant to our desired application scenairo. This misalignment with practical requirements — where users need compact, ready-to-deploy expert models without

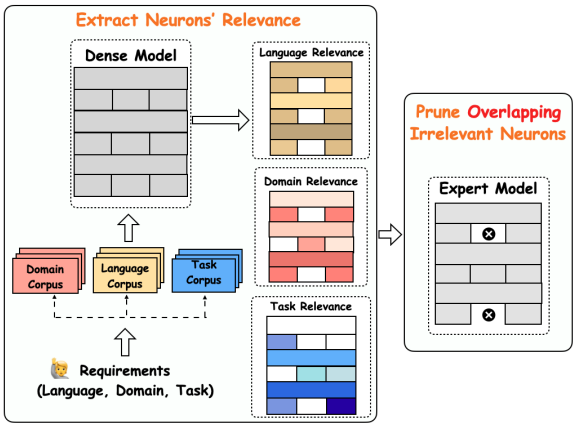

Figure 1: Given a request for an expert model across three dimensions (language, domain, and task), `Cus-Prun` (i) identifies irrelevant neurons for each dimension using corresponding corpora, and (ii) prunes **overlapping** irrelevant neurons across dimensions to obtain the expert model.

retraining — motivates us to design a more fine-grained and expert model targeting approach.

In this work, we introduce a novel Custom Pruning (`Cus-Prun`) method, designed to prune a large general-purpose model into a small specialized expert model tailored for specific scenarios. To achieve broad adaptability, we define the expert model by positioning the target user's needs along three key dimensions: language (e.g., English, Chinese, Germain), domain (e.g., E-commerce, education), and task (e.g., QA, summarization). Inspired by existing studies that certain neurons are responsible for certain functions (Zhao et al., 2024b; Tang et al., 2024; Liang et al., 2024), `Cus-Prun` identifies and preserves critical neurons that are more relevant to particular languages, domains, or tasks, while pruning less relevant ones, ultimately leading to a smaller expert models. Specifically, as illustrated in Figure 1, `Cus-Prun` first identifies irrelevant neurons for each dimension by assessing the impact of their removal on the generated output when processing corresponding corpus, which could be easily constructed from the relevant plain text documents. Next, the expert model is constructed by pruning irrelevant neurons across all dimensions. Furthermore, `Cus-Prun`'s flexibility allows it to focus on one, two, or all three dimensions (language, domain, task) as needed, making it adaptable to a wide range of real-world applications where specialized LLMs are required. Importantly, by performing fine-grained pruning at the neuron level, the method could also retain most of the essential neurons within the model backbone, thereby preserving most general capabilities.

We conduct comprehensive experiments to evaluate the performance of `Cus-Prun` across various scenarios. Experimental results demonstrate that it consistently outperforms other pruning methods in all settings. For three-dimensional specific expert models, `Cus-Prun` demonstrates remarkable pruning effectiveness across different model families and sizes, such as Mistral-Nemo-12B, Llama3-8B, Llama2-13B, and Llama3-70B, while maintaining strong expert and general capabilities. The method is evaluated across multilingual, multidomain, and multitask datasets, as well as representative compound NLP benchmarks, and consistently outperforms state-of-the-art pruning methods. Moreover, `Cus-Prun` is highly adaptable, working effectively across a wide range of pruning ratios, even up to nearly half the parameters, without compromising its superior performance. For more focused applications, such as two- or one-dimensional specific expert models (e.g., language-domain specific or language-specific models), `Cus-Prun` continues to significantly outperform other pruning methods, showcasing its versatility and effectiveness in diverse specialized settings.

2 Custom Pruning (`Cus-Prun`)

An expert model could be generally positioned from three dimensions: "language" ($L \in \mathbb{L}$), "domain" ($D \in \mathbb{D}$), and "task" ($T \in \mathbb{T}$), which can be represented as $LLM_{\text{Exp}} := (L, D, T) \in \mathbb{L} \times \mathbb{D} \times \mathbb{T}$. Specifically, the language dimension encompasses various languages such as English, Spanish, and Thai. The domain dimension covers different fields like finance, legal, and medical. The task dimension includes various applications such as question-answering, data-to-text, and summarization. In this section, we propose a custom pruning method named `Cus-Prun` to derive smaller expert models with flexible customization granularity.

2.1 Foundational Custom Pruning

Drawing inspiration from recent LLM interpretation studies (Tang et al., 2024; Liang et al., 2024; Zhao et al., 2024b) that many parameters in the model are redundant to processing a specific "language", we hypothesize that this phenomenon can be extended to other dimensions such as "domain" and "task", meaning that certain parameters remain

unused when handling a specific dimension. Rather than eliminating entire layers or modules (Song et al., 2024; Men et al., 2024; Zhang et al., 2024), `Cus-Prun` performs a fine-grained investigation by identifying and removing redundant neurons (i.e., individual rows or columns in parameter matrices) across all components (e.g., attention and feed-forward layers).

Concretely, when handling each dimension, we identify a specific set of *irrelevant neurons* in the original LLM, denoted as $\tilde{\mathcal{N}}_L$, $\tilde{\mathcal{N}}_D$, and $\tilde{\mathcal{N}}_T$ for L, D, and T, respectively. Specifically, to identify irrelevant neurons corresponding to the selected dimension, we construct a corpus within that dimension while ablating others. For example, to determine irrelevant neurons for a specific language L_{Exp}, we create a corpus set

$$C_{L_{\text{Exp}}} = \{(L_{\text{Exp}}, D, T) | D \in \mathbb{D}, T \in \mathbb{T}\}, \quad (1)$$

comprising documents in language L_{Exp} across various domains D and tasks T. We then identify neurons that are consistently irrelevant across all documents in $C_{L_{\text{Exp}}}$,

$$\tilde{\mathcal{N}}_{L_{\text{Exp}}} = \left\{\text{Neuron} \middle| \text{Irrelevant to } c,\ \forall c \in C_{L_{\text{Exp}}}\right\}, \quad (2)$$

where a neuron is considered irrelevant if its removal from the parameter matrix affects the generated output below a specified threshold. Formally, for i-th neuron in layer l, denoted as $N_i^{(l)}$, its relevance to document c is measured by $|h_{\setminus N_i^{(l)},i}(c) - h_i(c)|_2$, where $h_i(c)$ is the layer output and $h_{\setminus N_i^{(l)},i}(c)$ is the output with the neuron removed. Furthermore, neurons with impact in the lowest $\sigma\%$ are considered irrelevant, where σ is a pre-defined pruning ratio.

Similarly, we could establish corresponding corpus sets for other dimensions,

$$C_{D_{\text{Exp}}} = \{(L, D_{\text{Exp}}, T) | L \in \mathbb{L}, T \in \mathbb{T}\}, \quad (3)$$

$$C_{T_{\text{Exp}}} = \{(L, D, T_{\text{Exp}}) | L \in \mathbb{L}, D \in \mathbb{D}\}, \quad (4)$$

to extract irrelevant neurons, $\tilde{\mathcal{N}}_{D_{\text{Exp}}}$ and $\tilde{\mathcal{N}}_{T_{\text{Exp}}}$. Finally, the expert model could constructed by

$$\mathcal{LLM}_{\text{Exp}} = \mathcal{LLM} \ominus \left\{\tilde{\mathcal{N}}_{L_{\text{Exp}}} \cap \tilde{\mathcal{N}}_{D_{\text{Exp}}} \cap \tilde{\mathcal{N}}_{T_{\text{Exp}}}\right\}, \quad (5)$$

where $\ominus$ represents removing the corresponding neurons from $\mathcal{LLM}$. The overall algorithm is further illustrated in Algorithm 1.

Algorithm 1 Adaptive Custom Pruning

Input: Original language model $\mathcal{LLM}$, request for expert model $\mathcal{LLM}_{\text{Exp}}$ with selected dimensions: L_{Exp}, D_{Exp}, T_{Exp} (any subset), request for pruning ratio σ.

`// Construct specific corpora for each selected dimension.`
$C = \{\}$
if L_{Exp} is specified **then**
  $C = C \cup \{(L_{\text{Exp}}, D, T) \mid D \in \mathbb{D}, T \in \mathbb{T}\}$
end if
if D_{Exp} is specified **then**
  $C = C \cup \{(L, D_{\text{Exp}}, T) \mid L \in \mathbb{L}, T \in \mathbb{T}\}$
end if
if T_{Exp} is specified **then**
  $C = C \cup \{(L, D, T_{\text{Exp}}) \mid L \in \mathbb{L}, D \in \mathbb{D}\}$
end if
`// Identify irrelevant neurons for each selected dimension.`
for all neuron $N_i^{(l)}$ in $\mathcal{LLM}$ **do**
  if $\forall c \in C$, $N_i^{(l)} \in \tilde{\mathcal{N}}(c)$ **then**
    $\tilde{\mathcal{N}} \leftarrow \tilde{\mathcal{N}} \cup N_i^{(l)}$
  end if
end for
`// Prune irrelevant neurons to obtain expert model.`
$\mathcal{LLM}_{\text{Exp}} = \mathcal{LLM} \ominus \tilde{\mathcal{N}}$

Output: $\mathcal{LLM}_{\text{Exp}}$

2.2 Adaptive Custom Pruning

In many applications, the need for expertise might be constrained to one or two dimensions. For example, a language-specific or domain-specific model only requires pruning along a single dimension, while a language-domain-specific model (e.g., a Chinese Medical LLM) constrains two dimensions. In this section, we extend `Cus-Prun` to support different granularity levels.

Two-Dimensional Specific Expert Model Without losing generality, we use the language-domain expert model as a concrete example, which requires an expert model constrained in two dimensions: language (L_{Exp}) and domain (D_{Exp}). We similarly derive the sets of irrelevant neurons $\tilde{\mathcal{N}}_{L_{\text{Exp}}}$ and $\tilde{\mathcal{N}}_{D_{\text{Exp}}}$, and obtain the expert model by pruning the original dense model as follows:

$$\mathcal{LLM}_{\text{Exp}} := \mathcal{LLM} \ominus \{\tilde{\mathcal{N}}_{L_{\text{Exp}}} \cap \tilde{\mathcal{N}}_{D_{\text{Exp}}}\}. \quad (6)$$

One-Dimensional Specific Expert Model We use the language-specific expert model as an exam-

ple, which focuses on optimizing performance for a certain language (L_{Exp}), irrespective of domain or task. Similarly, we obtain the language-specific corpus $C_{L_{\text{Exp}}}$, then identify irrelevant neurons $\tilde{\mathcal{N}}_{L_{\text{Exp}}}$ and extract the expert model by

$$\mathcal{LLM}_{\text{Exp}} := \mathcal{LLM} \ominus \{\tilde{\mathcal{N}}_{L_{\text{Exp}}}\}. \quad (7)$$

To enhance efficiency, we implement the parallel neuron-detection method (Zhao et al., 2024b), which accelerates the sequential calculations from line14 to line16 in Algorithm 1.

3 Preliminary Evaluation

In this section, to simulate realistic user-defined requirements where all three dimensions (language, domain, task) are explicitly specified, we conduct preliminary experiments on developing expert models with fine-grained pruning. This setting not only validates our approach under the most demanding conditions but also serves as a prototype for later experiments with one- or two-dimensional customizations.

Experiment Design We consider three scenarios, each named according to the *language-domain-task* pattern: *Korean-Legal-Summarization* (Hwang et al., 2022), *English-Medical-MCQ* (García-Ferrero et al., 2024), and *Chinese-E-commerce-Sentiment Analysis* (Zhang et al., 2015). These scenarios were selected because each represents a unique combination of language, domain, and task, effectively simulating the diverse needs a user might specify in practice.

For each scenario, we curate the corresponding corpus for each dimension. This curation can be done through manual collection or by automatically retrieving relevant documents online. In this preliminary study, without loss of generality, we employ a strong proprietary model[1] to generate a corpus containing 50 documents for each dimension. Detailed prompts can be found in Appendix A.1. The generated documents could then be used to determine the relevance of neurons for each dimension of each scenario.

Experiment Setup We use Llama3-8B (Dubey et al., 2024) as the original dense model and follow the typical setting to set the pruning ratio as 25%. Performance is evaluated using Rouge-L (Lin, 2004) for Korean-Legal-Summary and accuracy score for another two tasks. For comparison, we adopt SliceGPT (Ashkboos et al., 2024) as a baseline, which replaces each weight matrix with a smaller proxy matrix without considering the specific use-case requirements.

[1] https://platform.openai.com/docs/models/gpt-4o

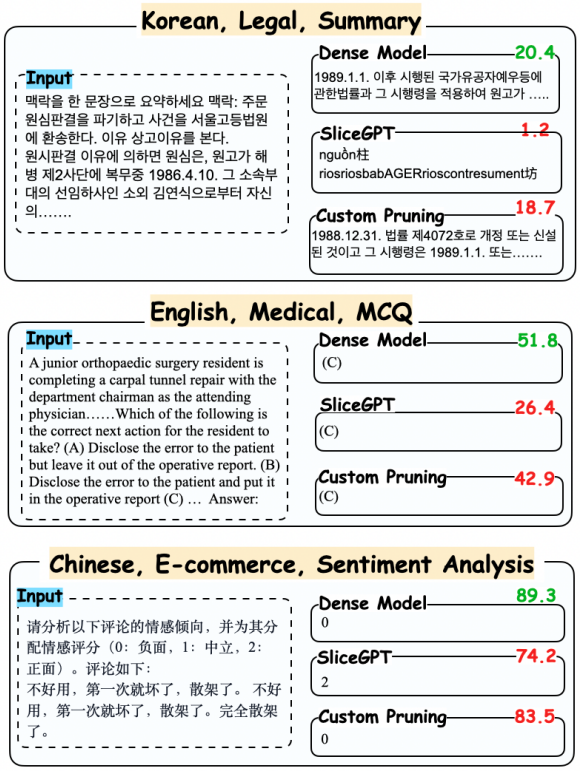

Figure 2: Concrete examples of applying `Cus-Prun` to prune 25% of Llama3-8B-Base's parameters into three-dimensional expert models. Numbers above each box indicate performance on the **whole** test set, with the first evaluated by Rouge-L, and the other two by accuracy.

Main Results Figure 2 presents the results and one concrete example for the original dense model, pruned model with SliceGPT, and pruned model with our proposed `Cus-Prun` method across the three scenarios. We observe that `Cus-Prun` largely preserves the performance of the dense model, retraining 92%, 83%, and 94% of the original dense model performance on these three cases respectively. In contrast, the baseline method SliceGPT, which does not consider specific use cases, largely underperforms compared to `Cus-Prun`. Overall, the results demonstrate that our proposed `Cus-Prun` method could effectively obtain expert models tailored to specific use cases across different languages, domains, and tasks that maintain high performance despite substantial pruning.

Table 1: Main Results of `Cus-Prun` on multilingual setting with a pruning ratio of 25%, where "general capability" is tested in English and averaged across several expert models, while "specific capability" is averaged across languages. Results are expressed in Rouge-L in summarization tasks and in accuracy (%) for other datasets.

	Method	General Capability				Multilingual Expert					Multidomain Expert					Multitask Expert			
		ARC-c	GSM8K	MMLU	Avg.	MGSM	M3	XQuAD	Sum	Avg.	MMCQ	FTQA	TSA	AMSA	Avg.	MSum	ASum	AMCF	Avg.
Llama3-8B	Dense	70.7	58.3	63.1	64.1	41.2	49.1	63.4	32.9	46.7	51.8	23.9	67.1	95.9	59.8	76.6	16.2	78.2	57.0
	LLMPrun.	26.3	2.5	24.2	17.7	1.1	24.0	13.6	23.2	15.5	0.0	0.0	**61.8**	76.0	34.5	62.2	**21.8**	**80.0**	**54.7**
	SliceGPT	41.5	0.0	24.2	21.9	0.0	14.9	16.6	8.5	10.0	22.6	0.0	41.2	53.7	29.4	7.3	2.9	51.3	20.5
	ShortGPT	38.3	0.0	28.6	22.3	0.0	26.9	0.0	2.7	7.4	3.2	0.0	38.6	35.7	19.4	4.1	4.8	43.8	17.6
	`Cus-Prun`	**62.4**	**37.0**	**54.7**	**51.4**	**30.1**	**41.5**	**52.6**	**31.5**	**38.9**	**42.9**	**20.6**	**61.8**	**87.6**	**53.2**	**68.4**	12.8	75.5	52.2
Mistral-12B	Dense	82.6	68.5	50.4	67.2	51.7	43.8	49.2	25.4	42.5	54.6	26.6	69.4	92.4	60.8	88.7	3.0	78.6	56.4
	LLMPrun.	22.5	2.7	30.7	18.6	2.1	27.8	19.0	**23.2**	18.0	0.0	0.0	51.0	20.9	18.0	59.3	0.5	2.8	20.9
	SliceGPT	49.4	1.9	32.1	27.8	0.8	25.1	17.4	7.8	12.8	24.9	9.2	34.2	54.3	30.7	27.4	1.3	36.3	21.7
	ShortGPT	37.8	0.0	33.9	23.9	2.9	27.0	18.0	5.0	13.2	31.4	7.2	39.2	52.5	32.6	26.2	0.2	42.7	23.0
	`Cus-Prun`	**67.5**	**43.4**	**43.8**	**51.6**	**34.3**	**39.2**	**40.7**	23.1	**34.3**	**47.9**	**25.1**	**67.3**	**83.7**	**56.0**	**83.5**	**3.4**	**72.8**	**50.9**
Llama2-13B	Dense	50.3	31.4	53.4	45.1	17.5	30.4	44.1	24.9	29.2	25.2	0.0	42.7	84.1	38.0	70.0	7.4	44.3	40.6
	LLMPrun.	22.4	2.1	23.6	16.0	1.1	22.8	3.8	17.7	11.3	0.0	0.0	9.7	0.0	2.4	21.6	4.8	0.0	8.8
	SliceGPT	45.9	2.4	48.7	32.3	2.8	25.3	23.4	9.9	15.5	18.7	0.0	28.4	67.3	28.6	24.5	4.9	32.9	20.8
	ShortGPT	39.5	3.8	37.2	26.8	2.4	23.0	24.7	11.3	15.3	16.9	0.0	34.6	**69.8**	30.3	23.8	5.2	39.1	22.7
	`Cus-Prun`	**48.3**	**20.8**	**50.0**	**39.7**	**12.7**	**26.2**	**34.2**	**24.1**	**24.3**	**25.6**	0.0	**38.5**	68.3	**33.1**	**64.5**	**6.7**	**42.9**	**38.0**
Llama3-70B	Dense	84.1	82.7	78.8	81.9	69.5	71.1	69.1	36.6	61.6	72.1	55.3	83.6	96.2	76.8	84.2	17.3	81.8	61.1
	LLMPrun.	69.1	26.0	53.2	49.4	16.8	43.7	43.0	29.0	33.1	27.3	1.0	51.0	50.3	32.4	10.2	13.7	20.6	14.8
	SliceGPT	65.7	0.0	54.2	40.0	3.7	44.8	33.0	21.2	25.7	57.6	27.6	68.1	59.4	53.2	58.0	14.2	68.3	46.8
	ShortGPT	59.4	5.6	**75.5**	46.8	11.9	43.1	38.8	24.0	29.5	58.4	32.2	67.5	64.9	55.8	59.6	13.9	65.8	46.4
	`Cus-Prun`	**68.4**	**53.2**	66.6	**62.7**	**43.1**	**57.7**	**59.8**	**34.3**	**48.7**	**68.2**	**43.9**	**81.4**	**87.8**	**70.3**	**80.4**	**15.7**	**77.5**	**57.9**

4 Foundational Custom Pruning Assessment

As demonstrated by preliminary evaluation in Section 3, `Cus-Prun` enables the creation of expert models tailored to specific languages, domains, and tasks. However, existing benchmark datasets may not always align with such specialized requirements for a systematic evaluation. To simplify our evaluation without losing generality, we use two distinct corpora: one focusing independently on a single dimension and another encompassing the remaining two dimensions. This approach allows us to evaluate `Cus-Prun`'s performance in *multilingual*, *multidomain*, and *multitask* settings.

Formally, in the multilingual setting, instead of constructing $C_{L_{\text{Exp}}}$, $C_{D_{\text{Exp}}}$ and $C_{T_{\text{Exp}}}$ independently, we can construct two corpora, $C_{L_{\text{Exp}}}$ and $C_{(D,T)_{\text{Exp}}}$, where $C_{L_{\text{Exp}}}$ helps to identify irrelevant neurons in a specific language ($\tilde{\mathcal{N}}_{L_{\text{Exp}}}$) and $C_{(D,T)_{\text{Exp}}}$ helps to identify irrelevant neurons in a specific domain-task combination ($\tilde{\mathcal{N}}_{D_{\text{Exp}} \cap T_{\text{Exp}}}$). Formally speaking, `Cus-Prun` in Equation 5 is transferred to

$$\begin{aligned} \mathcal{LLM}_{\text{Exp}} &= \mathcal{LLM} \ominus \left\{ \tilde{\mathcal{N}}_{L_{\text{Exp}}} \cap \left(\tilde{\mathcal{N}}_{D_{\text{Exp}}} \cap \tilde{\mathcal{N}}_{T_{\text{Exp}}} \right) \right\} \\ &\equiv \mathcal{LLM} \ominus \left\{ \tilde{\mathcal{N}}_{L_{\text{Exp}}} \cap \tilde{\mathcal{N}}_{D_{\text{Exp}} \cap T_{\text{Exp}}} \right\}. \end{aligned} \tag{8}$$

Note that this simplification is also applicable to $C_{D_{\text{Exp}}}$, $C_{(L,T)_{\text{Exp}}}$ and $C_{T_{\text{Exp}}}$, $C_{(L,D)_{\text{Exp}}}$.

4.1 Experiment Setup

Benchmarks Although the primary goal of `Cus-Prun` is to create expert LLMs tuned to specific use cases, we also evaluate the pruned models on standard general capabilities to ensure minimal performance loss. Specifically, we employ ARC-Challenge (Clark et al., 2018) (5-shots), GSM8K (Cobbe et al., 2021) (5-shots with CoT prompting (Wei et al., 2022)), and MMLU (Hendrycks et al., 2021) (5-shots) to represent models general capability. Note that we utilize a generation task and implement CoT prompting method, a more challenging setting that has not been previously evaluated by existing pruning techniques (Song et al., 2024; Sharma et al., 2024; Yang et al., 2024b; Zhang et al., 2024).

Baselines We employ several state-of-the-art pruning methods as the baseline that do not require post-training after pruning the model. (i) Dense represents the original model without pruning; (ii) LLM-Pruner (Ma et al., 2023) adopts structural pruning that selectively removes non-critical coupled structures based on gradient information;[2] (iii) SliceGPT (Ashkboos et al., 2024) replaces each weight matrix with a smaller dense matrix, reducing the embedding dimension of the network; (iv) ShortGPT (Men et al., 2024) directly deletes the redundant layers in LLMs based on an importance

[2]To ensure a fair comparison, we evaluate its performance before post-training, following Men et al. (2024).

score. We follow the typical pruning setting from these previous studies to set the pruning ratio to 25% for all methods and all models.

Backbone Models We choose 4 models from different model series and different sizes, including Llama3-8B-Base (Dubey et al., 2024), Mistral-Nemo-Base-2407[3](short as Mistral-12B), Llama2-13B-Base (Touvron et al., 2023), Llama3-70B-Base (Dubey et al., 2024).

4.2 Multilingual Setting

Dataset We employ multiple representative multilingual datasets for multilingual setting, which covers reasoning (MGSM (Shi et al., 2023), 5-shots), multilingual knowledge (M3Exam (Zhang et al., 2023), 3-shots, abbreviated as M3), understanding (XQuAD (Artetxe et al., 2020), 5-shots), and generation (XLSum (Hasan et al., 2021), zero-shots, abbreviated as Sum). Furthermore, we consider three languages spanning a range from high-resource to low-resource including German (De), Chinese (Zh) and Thai (Th). More detailed experiment settings are explained in Appendix A.3.1.

Main Results Table 1 summarizes the performance of `Cus-Prun` on multilingual datasets, with average scores across languages. Detailed breakdown results for each language are shown in Table 5, Table 6 and Table 7 in Appendix A.2. We can observe that `Cus-Prun` consistently outperforms baseline pruning methods in achieving expert capabilities while preserving general performance. For example, on Llama3-8B, `Cus-Prun` achieves a score of 38.9 compared to at most 15.5 for other methods. Similar improvements are observed for Mistral-12B (34.3 vs. 18.0), Llama2-13B (24.3 vs. 15.5), and Llama3-70B (48.7 vs. 33.1). We can also see that the generation tasks are especially challenging; for instance, `Cus-Prun` achieves a score of 30.1 on MGSM for Llama3-8B, while other methods nearly lose the ability to generate coherent reasoning outputs (often approaching 0 accuracy for all but the largest model). Additionally, `Cus-Prun` performs robustly across both high-resource and low-resource languages. Overall, `Cus-Prun` excels in both performance and robustness across diverse tasks and languages.

[3]https://huggingface.co/mistralai/Mistral-Nemo-Base-2407

4.3 MultilDomain Setting

Dataset For the multidomain setting, we employ several domain-specific datasets, including medical domain multiply choices questions (MedMCQ (Pal et al., 2022), 3-shots, abbreviated as MMCQ), finance domain table question-answering (FinTQA (Chen et al., 2021), 8-shots, abbreviated as FTQA), social media domain sentiment analysis (TSA (Kharde and Sonawane, 2016), 3-shots), and e-commerce domain sentiment analysis (AMSA (Zhang et al., 2015), 3-shots). Moreover, in multidomain setting, our focus is exclusively on the English language. Detailed experiment settings are explained in Appendix A.3.2.

Main Results Table 1 shows the performance of `Cus-Prun` on multidomain setting. We find that `Cus-Prun` consistently outperforms other pruning methods in both expert and general capabilities. For expert capabilities, `Cus-Prun` achieves a score of 53.2 on Llama3-8B, while other pruning methods achieve at most 34.5. Similar improvements are observed for Mistral-12B (56.0 vs. 32.6), Llama2-13B (33.1 vs. 30.3), and Llama3-70B (70.3 vs. 55.8).

4.4 MultiTask Setting

Dataset For the multitask setting, we employ several task-specific datasets, including the medical summarization task (MedSum (Abacha and Demner-Fushman, 2019), 3-shots, abbreviated as MSum), summarization task in e-commerce (Amazon Summary (Wang et al., 2022; Brüel-Gabrielsson et al., 2024), 3-shots, abbreviated as ASum), counterfactual task in e-commerce (Amazon Counterfactual (O'Neill et al., 2021), 3-shots, abbreviated as AMCF). Similarly, in multitask setting scenarios, our focus is exclusively on the English language. Detailed experiment settings are explained in Appendix A.3.3.

Main Results Table 1 shows the performance of `Cus-Prun` on multitask setting. We find that except for LLM-Pruner under Llama3-8B, `Cus-Prun` outperforms other pruning methods in both expert and general capabilities. For expert tasks, `Cus-Prun` achieves a score of 50.9 on `Mistral-12B`, significantly outperforming the highest score of 23.0 from other methods. Likewise, it attains 38.0 on `Llama2-13B` and 57.9 on `Llama3-70B`, both markedly higher than the corresponding maximum scores of 22.7 and 46.8

Table 2: Performance of Chinese-Medical expert model on MCQ task

Method	General	CMExam
Dense	59.3	50.6
LLM-Pruner	18.6	25.0
SliceGPT	27.8	26.9
ShortGPT	23.9	23.7
Cus-Prun	**52.4**	**48.7**

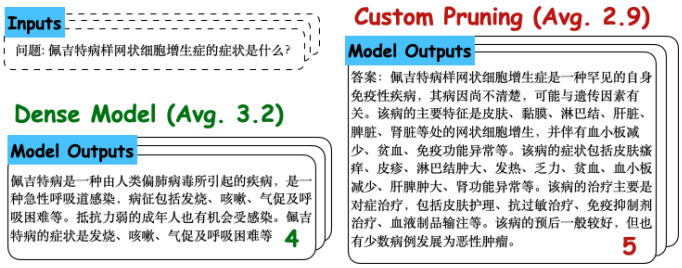

Figure 3: Chinese Medical LLM performance. Numbers are quality on the **whole** test set evaluated by GPT4.

achieved by competing methods.

4.5 Analysis on Aggressive Pruning Ratio

To optimize for specialized tasks rather than maintaining general capabilities, we employ more aggressive pruning ratios. We combine layer pruning with our custom neuron pruning method in Algorithm 1 and evaluate the approach on M3Exam, MedMCQ, and Amazon Counterfactual (AMContFact) datasets using Llama3-8B. Detailed results are shown in Table 3. We find that Cus-Prun consistently maintains the model's capabilities even at higher pruning ratios. Specifically, when the pruning ratio is increased to 45%, ShortGPT nearly loses the capability of generating meaningful answers, while Cus-Prun still achieves scores of 48.4 on MMLU and 50.6 on expert capabilities.

Table 3: Aggressive pruning ratio on Llama3-8B.

Method	Ratio	Speedup	MMLU	Expert
Dense	0.0	1×	63.1	59.7
ShortGPT	25.0	1.3×	28.6	24.6
Cus-Prun	25.0	1.3×	**51.9**	**53.3**
ShortGPT	34.2	1.5×	20.8	18.5
Cus-Prun	35.0	1.5×	**50.2**	**51.4**
ShortGPT	43.8	1.8×	7.9	10.2
Cus-Prun	45.0	1.8×	**48.4**	**50.6**

5 Adaptive Custom Pruning Assessment

In this section, we evaluate the generality of Cus-Prun in dynamic scenarios, including specific expert models in two and one dimensions, as described in Section 2.2.

5.1 Two Dimensions Specific Expert Model

Experiment Settings We use the Chinese-Medical setting as a concrete example of a two-dimensional expert model designed to perform a wide range of medical tasks in Chinese. We adopt Mistral-12b as the backbone model and utilize corpus from Wikipedia for Chinese content and general medical corpus for medical knowledge. The performance of the target Chinese-Medical expert model is evaluated on two datasets: CMExam (Liu et al., 2023) (5-shots), a Chinese medical multiple-choice question dataset, and HuatuoQA (Li et al., 2023a), a Chinese medical question-answering dataset. We assess the performance on CMExam using accuracy metrics. For the latter, we sample a sub-testset of size 100 and use GPT-4 as the evaluator, which assigns a score from 0 to 5, representing its quality from low to high. Detailed prompts are listed in Appendix A.1.

Main Results Table 2 presents the performance of the Chinese-Medical LLM on CMExam and its general capabilities. Our results indicate that the expert model pruned using Cus-Prun outperforms models obtained through other pruning methods. Specifically, Cus-Prun achieves a score of 48.7 on CMExam, while its general capability score is 52.4. These results compare favorably to the dense model, which scores 50.6 on CMExam and 59.3 on general capabilities. On the contrary, other pruning methods nearly lose the general and specific capabilities. Furthermore, Figure 3 shows a concrete example of Chinese-Medical LLM performance on medical question-answering. We find that Cus-Prun can produce smaller expert models that maintain their expert capabilities, as demonstrated by its performance score of 2.9/5.0 compared to 3.2/5.0 for the dense model.

5.2 One Dimension Specific Expert Model

Experiment Settings For evaluating the pruning method under a one-dimensional expert model setting, we focus on language-specific pruning, showing how to transform a dense model into

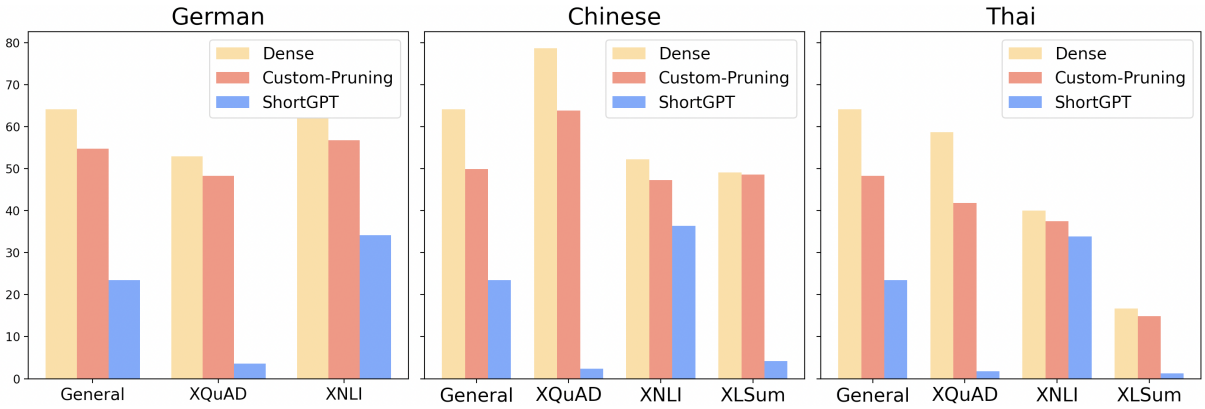

Figure 4: Performance of `Cus-Prun` in obtaining language-specific models.

language-specific variants. We consider three linguistically diverse languages: German, Chinese, and Thai. We conduct experiments based on the Llama3-8b model. To identify language-specific (while domain- and task-agnostic) neurons, we employ a diverse range of corpora, including Wikipedia, MGSM, and M3Exam, ensuring coverage of various domains and tasks. The effectiveness of our pruning technique is then evaluated using three held-out multilingual datasets including XQuAD (Artetxe et al., 2020), XNLI (Conneau et al., 2018), and XSum (Narayan et al., 2018).

Main Results Figure 4 illustrates the performance of language-specific models using `Cus-Prun`. By pruning 25% of the neurons from the original model, `Cus-Prun` not only retains general performance but also preserves language-specific capabilities. For instance, the German-specific model scores 54.7 in general capabilities, 48.3 on XQuAD, and 56.8 on XNLI, compared to the dense model's scores of 64.1, 52.9, and 62.0, respectively. This trend is consistent for Chinese and Thai models as well. In contrast, ShortGPT struggles to maintain the model's capabilities, particularly in XQuAD and XLSUm, which require generative abilities.

6 Related Work

LLM Compression Given the high costs of training, inferencing, and tuning LLMs, many studies focus on model compression methods, including compression (Zhu et al., 2023), quantization (Xu et al., 2023; Dettmers et al., 2024; Lin et al., 2024; Li et al., 2024), and pruning (Wang et al., 2019). Sparsity-based structural pruning enhances GPU efficiency with sparse structures but doesn't always reduce parameter counts (Li et al., 2022, 2023c; Kurz et al., 2024; Zhao et al., 2024a; Huang et al., 2024). Unstructured pruning reduces parameters while maintaining performance, using post-training techniques (Ma et al., 2023; Xia et al., 2024; Muralidharan et al., 2024) or coarse methods like parameter approximation (Zhao et al., 2024a), layer removal (Men et al., 2024), or structural elimination (Zhang et al., 2024). However, these often fail to preserve domain- or task-specific capabilities, limiting their utility for specialized scenarios.

Customizing Model Customizing LLMs is essential for addressing language-specific challenges, domain-specific needs (e.g., healthcare, software development), and task-specific applications (Cui et al., 2023; Yang et al., 2024b; Li et al., 2023a; Roziere et al., 2023; Li et al., 2023b; Azerbayev et al., 2024; Alves et al., 2024). However, customization often demands extensive fine-tuning with curated data, making it resource-intensive. This highlights the need for efficient methods to quickly create robust, expert models tailored to diverse industries without compromising quality.

7 Conclusion

LLMs deliver impressive capabilities but incur high computational costs. Efficient pruning of redundant parameters is vital for conserving resources and improving inference speed, especially for specialized models. Our method, `Cus-Prun`, generates smaller expert models without post-training by pruning irrelevant neurons across "language", "domain", and "task" dimensions. This finer-grained approach outperforms existing techniques on three-dimensional models and can adapt to realistic scenarios, such as language-domain or language-specific models.

Limitation

Despite the promising results of Cus-Prun, several limitations should be noted. First, while our method leverages three dimensions (language, domain, and task) for pruning, certain crucial restrictions cannot be fully captured within this framework, such as variations in query format or input structure. Second, whether pruned base models can effectively undergo post-training remains an open question that requires further investigation. This uncertainty about post-training capabilities could limit the model's adaptability to new scenarios or requirements after pruning. These limitations suggest important directions for future research, including exploring additional dimensions for more comprehensive pruning strategies and investigating the relationship between pruning and post-training effectiveness.

References

Asma Ben Abacha and Dina Demner-Fushman. 2019. On the summarization of consumer health questions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2228–2234.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Duarte M Alves, José Pombal, Nuno M Guerreiro, Pedro H Martins, João Alves, Amin Farajian, Ben Peters, Ricardo Rei, Patrick Fernandes, Sweta Agrawal, et al. 2024. Tower: An open multilingual large language model for translation-related tasks. *arXiv preprint arXiv:2402.17733*.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637.

Saleh Ashkboos, Maximilian L Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. 2024. Slicegpt: Compress large language models by deleting rows and columns. In *The Twelfth International Conference on Learning Representations*.

Mohammed Attia, Younes Samih, Ali Elkahky, and Laura Kallmeyer. 2018. Multilingual multi-class sentiment classification using convolutional neural networks. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen Marcus McAleer, Albert Q Jiang, Jia Deng, Stella Biderman, and Sean Welleck. 2024. Llemma: An open language model for mathematics. In *The Twelfth International Conference on Learning Representations*.

Rickard Brüel-Gabrielsson, Jiacheng Zhu, Onkar Bhardwaj, Leshem Choshen, Kristjan Greenewald, Mikhail Yurochkin, and Justin Solomon. 2024. Compress then serve: Serving thousands of lora adapters with little overhead. *Preprint*, arXiv:2407.00066.

Wenhu Chen, Ming-Wei Chang, Eva Schlinger, William Yang Wang, and William W Cohen. 2020. Open question answering over tables and text. In *International Conference on Learning Representations*.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan R Routledge, et al. 2021. Finqa: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3697–3711.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485.

Yiming Cui, Ziqing Yang, and Xin Yao. 2023. Efficient and effective text encoding for chinese llama and alpaca. *arXiv preprint arXiv:2304.08177*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Iker García-Ferrero, Rodrigo Agerri, Aitziber Atutxa, Elena Cabrio, Iker de la Iglesia, Alberto Lavelli, Bernardo Magnini, Benjamin Molinet, Johana Ramirez-Romero, German Rigau, et al. 2024. Medical mt5: An open-source multilingual text-to-text llm

for the medical domain. In *LREC-COLING 2024-2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation*.

Josh A Goldstein, Girish Sastry, Micah Musser, Renee DiResta, Matthew Gentzel, and Katerina Sedova. 2023. Generative language models and automated influence operations: Emerging threats and potential mitigations. *arXiv preprint arXiv:2301.04246*.

Tahmid Hasan, Abhik Bhattacharjee, Md Saiful Islam, Kazi Mubasshir, Yuan-Fang Li, Yong-Bin Kang, M Sohel Rahman, and Rifat Shahriyar. 2021. Xl-sum: Large-scale multilingual abstractive summarization for 44 languages. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4693–4703.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Weiyu Huang, Guohao Jian, Yuezhou Hu, Jun Zhu, and Jianfei Chen. 2024. Pruning large language models with semi-structural adaptive sparse training. *arXiv preprint arXiv:2407.20584*.

Wonseok Hwang, Dongjun Lee, Kyoungyeon Cho, Hanuhl Lee, and Minjoon Seo. 2022. A multi-task benchmark for korean legal language understanding and judgement prediction. *Advances in Neural Information Processing Systems*, 35:32537–32551.

Phillip Keung, Yichao Lu, György Szarvas, and Noah A. Smith. 2020. The multilingual amazon reviews corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Vishal A Kharde and SS Sonawane. 2016. Sentiment analysis of twitter data: A survey of techniques. *International Journal of Computer Applications*, 975:8887.

Simon Kurz, Zhixue Zhao, Jian-Jia Chen, and Lucie Flek. 2024. Language-specific calibration for pruning multilingual language models. *arXiv preprint arXiv:2408.14398*.

Jianquan Li, Xidong Wang, Xiangbo Wu, Zhiyi Zhang, Xiaolong Xu, Jie Fu, Prayag Tiwari, Xiang Wan, and Benyou Wang. 2023a. Huatuo-26m, a large-scale chinese medical qa dataset. *Preprint*, arXiv:2305.01526.

Shiyao Li, Xuefei Ning, Luning Wang, Tengxuan Liu, Xiangsheng Shi, Shengen Yan, Guohao Dai, Huazhong Yang, and Yu Wang. 2024. Evaluating quantized large language models. *arXiv preprint arXiv:2402.18158*.

Yinheng Li, Shaofei Wang, Han Ding, and Hang Chen. 2023b. Large language models in finance: A survey. In *Proceedings of the fourth ACM international conference on AI in finance*, pages 374–382.

Yixiao Li, Yifan Yu, Qingru Zhang, Chen Liang, Pengcheng He, Weizhu Chen, and Tuo Zhao. 2023c. Losparse: Structured compression of large language models based on low-rank and sparse approximation. In *International Conference on Machine Learning*, pages 20336–20350. PMLR.

Yuchao Li, Fuli Luo, Chuanqi Tan, Mengdi Wang, Songfang Huang, Shen Li, and Junjie Bai. 2022. Parameter-efficient sparsity for large language models fine-tuning. *arXiv preprint arXiv:2205.11005*.

Yunlong Liang, Fandong Meng, Songming Zhang, Yufeng Chen, Jinan Xu, Jie Zhou, et al. 2024. Multilingual knowledge editing with language-agnostic factual neurons. *arXiv preprint arXiv:2406.16416*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100.

Junling Liu, Peilin Zhou, Yining Hua, Dading Chong, Zhongyu Tian, Andrew Liu, Helin Wang, Chenyu You, Zhenhua Guo, Lei Zhu, et al. 2023. Benchmarking large language models on cmexam–a comprehensive chinese medical exam dataset. *arXiv preprint arXiv:2306.03030*.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 36:21702–21720.

Xin Men, Mingyu Xu, Qingyu Zhang, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. 2024. Shortgpt: Layers in large language models are more redundant than you expect. *arXiv preprint arXiv:2403.03853*.

Saurav Muralidharan, Sharath Turuvekere Sreenivas, Raviraj Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. 2024. Compact language models via pruning and knowledge distillation. *arXiv preprint arXiv:2407.14679*.

Micah Musser. 2023. A cost analysis of generative language models and influence operations. *arXiv preprint arXiv:2308.03740*.

Shashi Narayan, Shay Cohen, and Maria Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *2018 Conference on Empirical Methods in Natural Language Processing*.

James O'Neill, Polina Rozenshtein, Ryuichi Kiryo, Motoko Kubota, and Danushka Bollegala. 2021. I wish

i would have loved this one, but i didn't–a multilingual dataset for counterfactual detection in product review. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7092–7108.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. 2022. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Proceedings of the Conference on Health, Inference, and Learning*, volume 174 of *Proceedings of Machine Learning Research*, pages 248–260. PMLR.

P Rajpurkar. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Pratyusha Sharma, Jordan T Ash, and Dipendra Misra. 2024. The truth is in there: Improving reasoning in language models with layer-selective rank reduction. In *The Twelfth International Conference on Learning Representations*.

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, et al. 2023. Language models are multilingual chain-of-thought reasoners. In *The Eleventh International Conference on Learning Representations*.

Jiwon Song, Kyungseok Oh, Taesu Kim, Hyungjun Kim, Yulhwa Kim, et al. 2024. Sleb: Streamlining llms through redundancy verification and elimination of transformer blocks. In *Forty-first International Conference on Machine Learning*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158.

Tianyi Tang, Wenyang Luo, Haoyang Huang, Dongdong Zhang, Xiaolei Wang, Xin Zhao, Furu Wei, and Ji-Rong Wen. 2024. Language-specific neurons: The key to multilingual capabilities in large language models. *arXiv preprint arXiv:2402.16438*.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

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

Pingjie Wang, Ziqing Fan, Shengchao Hu, Zhe Chen, Yanfeng Wang, and Yu Wang. 2024. Reconstruct the pruned model without any retraining. *arXiv preprint arXiv:2407.13331*.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Gary Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Maitreya Patel, Kuntal Kumar Pal, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Shailaja Keyur Sampat, Savan Doshi, Siddhartha Mishra, Sujan Reddy, Sumanta Patro, Tanay Dixit, Xudong Shen, Chitta Baral, Yejin Choi, Noah A. Smith, Hannaneh Hajishirzi, and Daniel Khashabi. 2022. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. *Preprint*, arXiv:2204.07705.

Ziheng Wang, Jeremy Wohlwend, and Tao Lei. 2019. Structured pruning of large language models. *arXiv preprint arXiv:1910.04732*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. 2024. Sheared llama: Accelerating language model pre-training via structured pruning. In *The Twelfth International Conference on Learning Representations*.

Yuhui Xu, Lingxi Xie, Xiaotao Gu, Xin Chen, Heng Chang, Hengheng Zhang, Zhengsu Chen, Xiaopeng Zhang, and Qi Tian. 2023. Qa-lora: Quantization-aware low-rank adaptation of large language models. *arXiv preprint arXiv:2309.14717*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024a. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Yifei Yang, Zouying Cao, and Hai Zhao. 2024b. Laco: Large language model pruning via layer collapse. *arXiv preprint arXiv:2402.11187*.

Wenxuan Zhang, Mahani Aljunied, Chang Gao, Yew Ken Chia, and Lidong Bing. 2023. M3exam: A multilingual, multimodal, multilevel benchmark for examining large language models. *Advances in Neural Information Processing Systems*, 36:5484–5505.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Yang Zhang, Yawei Li, Xinpeng Wang, Qianli Shen, Barbara Plank, Bernd Bischl, Mina Rezaei, and Kenji Kawaguchi. 2024. Finercut: Finer-grained interpretable layer pruning for large language models. *arXiv preprint arXiv:2405.18218*.

Pengxiang Zhao, Hanyu Hu, Ping Li, Yi Zheng, Zhefeng Wang, and Xiaoming Yuan. 2024a. A convex-optimization-based layer-wise post-training pruner for large language models. *arXiv preprint arXiv:2408.03728*.

Yiran Zhao, Wenxuan Zhang, Guizhen Chen, Kenji Kawaguchi, and Lidong Bing. 2024b. How do large language models handle multilingualism? *arXiv preprint arXiv:2402.18815*.

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*.

A Appendix

A.1 GPT-4o Prompts

Task	Prompt
Generation	Generate a text document in {language}/{domain}/{task}. Make sure the documents is not fixed to one {language}/{domain}/{task} or {language}/{domain}/{task}. Ensure the content is clear, concise, and appropriate for the specified request. Use professional and domain-specific terminology where necessary.
Evaluation	Evaluate the quality of the given answer to the question. Provide a score from 0 to 5, where 0 represents very low quality and 5 represents very high quality. Question: {question} Answer: {answer}.

Table 4: GPT-4o prompts for generating documents and evaluating answer quality.

A.2 Detailed Results for Multilingual

Detailed results for multilingual settings can be found in Table 5, Table 6 and Table 7 for German, Chinese and Thai correspondingly.

A.3 Experiments Detailed Settings

A.3.1 Multilingual Settings

Experiment Details For multilingual setting, we can obtain two corpora: $C_{L_{\text{Exp}}}$ and $C_{(D,T)_{\text{Exp}}}$. The first corpus contains samples in a specific language across various domains and tasks, while the second corpus contains samples from a specific domain-task combination in other languages, i.e., the target dataset in other languages. Specifically, for $C_{L_{\text{Exp}}}$ we employ Wikipedia[4] to construct language-specific corpus covering various domains and tasks. For $C_{(D,T)_{\text{Exp}}}$, we employ the corresponding datasets in English, including GSM8K (Cobbe et al., 2021) for MGSM, the English split of M3Exam[5] for M3Exam, SQuAD (Rajpurkar, 2016) for XQuAD, and XSum (Narayan et al., 2018) for XLSum.

Hyperparameters, including the sizes of $C_{L_{\text{Exp}}}$ and $C_{(D,T)_{\text{Exp}}}$, are determined using the validation set of the XLSum dataset and then applied to testsets in other multilingual datasets. Furthermore, accuracy is the metric used for ARC-c, GSM8K, MMLU, MGSM, M3Exam, and XQuAD, while Rouge-L (Lin, 2004) is used for XLSum.

A.3.2 Multidomain Settings

Settings For multidomain setting, we can obtain two corpora: $C_{D_{\text{Exp}}} = \{(L, D_{\text{Exp}}, T) | L \in \mathbb{L}, T \in \mathbb{T}\}$ and $C_{(L,T)_{\text{Exp}}} = \{(D, (L,T)_{\text{Exp}}) | D \in \mathbb{D}\}$. The first corpus contains samples in a specific domain across various languages and tasks, while the second corpus contains samples from a specific language-task combination across different domains, i.e., the target dataset in other domains. Specifically, for $C_{D_{\text{Exp}}}$ we employ specific domain corpus, including English split of medical corpus (García-Ferrero et al., 2024) for medical domain, general finance corpus for finance domain[6], general Twitter corpus (Kharde and Sonawane, 2016), and English split of Amazon corpus (Keung et al., 2020). For $C_{(L,T)_{\text{Exp}}}$, we employ the corresponding datasets in general domains, including CommonsenseQA (Talmor et al., 2019) for MedMCQ, open table question-answering OTT-QA (Chen et al., 2020) for FinTQA, general sentiment analysis (Attia et al., 2018) for TSA and

[4] https://huggingface.co/datasets/wikimedia/wikipedia

[5] M3Exam is language-specific and does not utilize a translated parallel corpus.

[6] https://huggingface.co/datasets/gbharti/finance-alpaca

Table 5: Main Results of `Cus-Prun` on Germany with a pruning ratio of 25%, where "general capability" is tested in English and averaged across several expert models, while "specific capability" is averaged across languages. Results are expressed in Rouge-L in XLSum and in accuracy (%) for other datasets.

Model	Method	General Capability				Expert Capability				
		ARC-c	GSM8K	MMLU	Avg.	MGSM	M3Exam	XQuAD	XLSum	Avg.
Llama3-8B	Dense	70.7	58.3	63.1	64.1	44.8	-	52.9	-	48.8
	LLMPrun.	26.3	2.5	24.2	17.7	0.0	-	11.0	-	5.5
	SliceGPT	41.5	0.0	24.2	21.9	0.0	-	9.8	-	4.9
	ShortGPT	38.3	0.0	28.6	22.3	0.0	-	0.0	-	0.0
	`Cus-Prun`	61.4	38.9	54.5	**51.6**	32.8	-	49.6	-	**41.2**
Mistral-12B	Dense	82.6	68.5	50.4	59.3	56.8	-	41.2	-	49.0
	LLMPrun.	22.5	2.7	30.7	18.6	2.4	-	13.4	-	7.9
	SliceGPT	49.4	1.9	32.1	27.8	0.8	-	15.5	-	8.2
	ShortGPT	37.8	0.0	33.9	23.9	3.6	-	20.3	-	12.0
	`Cus-Prun`	64.6	39.7	43.2	**49.2**	31.6	-	35.9	-	**33.8**
Llama2-13B	Dense	50.3	31.4	53.4	45.1	24.4	-	40.3	-	32.3
	LLMPrun.	22.4	2.1	23.6	16.0	2.0	-	5.7	-	3.9
	SliceGPT	45.9	2.4	48.7	32.3	3.6	-	18.1	-	10.9
	ShortGPT	39.5	3.8	37.2	26.8	2.8	-	27.2	-	15.0
	`Cus-Prun`	47.6	19.8	49.9	**39.1**	18.4	-	31.7	-	**25.0**
Llama3-70B	Dense	84.1	82.7	78.8	81.9	74.8	-	58.2	-	66.5
	LLMPrun.	69.1	26.0	53.2	49.4	18.0	-	27.3	-	22.7
	SliceGPT	65.7	0.0	54.2	40.0	0.0	-	17.3	-	8.7
	ShortGPT	59.4	5.6	75.5	46.8	9.6	-	31.5	-	20.6
	`Cus-Prun`	66.8	59.3	69.1	**65.1**	48.2	-	53.9	-	**51.1**

Table 6: Main Results of `Cus-Prun` on Chinese with a pruning ratio of 25%, where "general capability" is tested in English and averaged across several expert models, while "specific capability" is averaged across languages. Results are expressed in Rouge-L in XLSum and in accuracy (%) for other datasets.

Model	Method	General Capability				Specific Capability				
		ARC-c	GSM8K	MMLU	Avg.	MGSM	M3Exam	XQuAD	XLSum	Avg.
Llama3-8B	Dense	70.7	58.3	63.1	64.1	43.6	55.1	78.7	49.1	56.6
	LLMPrun.	26.3	2.5	24.2	17.7	2.4	23.6	21.3	32.8	20.0
	SliceGPT	41.5	0.0	24.2	21.9	0.0	17.4	23.5	8.3	12.3
	ShortGPT	38.3	0.0	28.6	22.3	0.0	28.3	0.0	3.1	7.9
	`Cus-Prun`	60.5	25.7	49.4	**45.2**	36.0	44.7	65.6	46.3	**48.2**
Mistral-12B	Dense	82.6	68.5	50.4	59.3	53.2	47.8	62.2	33.0	49.1
	LLMPrun.	22.5	2.7	30.7	18.6	2.8	30.7	31.8	32.6	24.5
	SliceGPT	49.4	1.9	32.1	27.8	1.6	26.4	28.3	10.8	16.8
	ShortGPT	37.8	0.0	33.9	23.9	4.4	28.2	29.1	7.2	17.2
	`Cus-Prun`	68.3	43.2	39.5	**50.3**	38.4	40.7	50.6	30.3	**40.0**
Llama2-13B	Dense	50.3	31.4	53.4	45.1	21.6	36.5	59.8	35.3	38.3
	LLMPrun.	22.4	2.1	23.6	16.0	1.2	23.3	3.8	25.1	13.4
	SliceGPT	45.9	2.4	48.7	32.3	4.8	24.5	28.4	11.2	17.2
	ShortGPT	39.5	3.8	37.2	26.8	4.4	22.9	24.6	13.7	16.4
	`Cus-Prun`	48.6	20.7	51.9	**40.4**	14.8	28.2	47.3	34.4	**31.2**
Llama3-70B	Dense	84.1	82.7	78.8	81.9	68.4	76.1	81.3	55.3	70.3
	LLMPrun.	69.1	26.0	53.2	49.4	16.8	47.5	56.1	41.3	40.4
	SliceGPT	65.7	0.0	54.2	40.0	6.4	48.3	42.2	29.3	31.6
	ShortGPT	59.4	5.6	75.5	46.8	12.4	45.5	44.6	36.1	34.7
	`Cus-Prun`	72.3	48.5	65.2	**62.0**	40.8	61.7	66.9	51.6	**55.3**

Table 7: Main Results of `Cus-Prun` on Thai with a pruning ratio of 25%, where "general capability" is tested in English and averaged across several expert models, while "specific capability" is averaged across languages. Results are expressed in Rouge-L in XLSum and in accuracy (%) for other datasets.

Model	Method	General Capability				Specific Capability				
		ARC-c	GSM8K	MMLU	Avg.	MGSM	M3Exam	XQuAD	XLSum	Avg.
Llama3-8B	Dense	70.7	58.3	63.1	64.1	35.2	43.0	58.7	16.7	38.4
	LLMPrun.	26.3	2.5	24.2	17.7	0.8	24.4	8.4	13.5	11.8
	SliceGPT	41.5	0.0	24.2	21.9	0.0	12.3	16.6	8.7	9.4
	ShortGPT	38.3	0.0	28.6	22.3	0.0	25.4	0.0	2.3	6.9
	`Cus-Prun`	58.9	31.2	52.4	**47.5**	21.6	38.3	42.6	16.8	**29.8**
Mistral-12B	Dense	82.6	68.5	50.4	59.3	45.2	39.9	44.1	17.8	36.8
	LLMPrun.	22.5	2.7	30.7	18.6	1.2	24.8	11.9	13.7	12.9
	SliceGPT	49.4	1.9	32.1	27.8	0.0	23.8	8.4	4.7	12.3
	ShortGPT	39.5	3.8	37.2	26.8	0.8	25.7	4.7	2.8	8.5
	`Cus-Prun`	68.2	35.8	47.6	**50.5**	32.8	37.7	35.6	15.9	**30.5**
Llama2-13B	Dense	50.3	31.4	53.4	45.1	6.4	24.3	28.3	14.5	18.4
	LLMPrun.	22.4	2.1	23.6	16.0	0.0	22.3	1.8	10.2	8.6
	SliceGPT	45.9	2.4	48.7	32.3	0.0	26.2	23.7	8.6	14.6
	ShortGPT	39.5	3.8	37.2	26.8	0.0	23.1	22.3	8.9	13.6
	`Cus-Prun`	47.8	20.9	50.7	**39.8**	4.8	24.2	23.6	13.8	**16.6**
Llama3-70B	Dense	84.1	82.7	78.8	81.9	65.2	66.1	67.8	17.8	54.2
	LLMPrun.	69.1	26.0	53.2	49.4	15.6	39.9	29.8	16.6	25.5
	SliceGPT	65.7	0.0	54.2	40.0	4.8	41.3	39.6	13.2	24.7
	ShortGPT	59.4	5.6	75.5	46.8	13.7	40.7	40.4	11.9	26.7
	`Cus-Prun`	73.3	58.7	68.4	**66.8**	40.4	53.6	58.5	16.9	**42.4**

AMSA.

Experiment Details Hyperparameters, including the sizes of $C_{D_{\text{Exp}}}$ and $C_{(L,T)_{\text{Exp}}}$, are determined using the validation set of the Amazon sentiment analysis dataset and then applied to testsets in other multidomain datasets. Furthermore, accuracy is the metric used for all datasets.

A.3.3 Multitask Settings

Settings For multitask setting, we can obtain two corpora: $C_{T_{\text{Exp}}} = \{(L, D, T_{\text{Exp}}) | L \in \mathbb{L}, D \in \mathbb{D}\}$ and $C_{(L,D)_{\text{Exp}}} = \{(T, (L, S)_{\text{Exp}}) | T \in \mathbb{T}\}$. The first corpus contains samples in a specific task across various languages and domains, while the second corpus contains samples from a specific language-domain combination across different tasks, i.e., the target dataset in other tasks. Specifically, for $C_{T_{\text{Exp}}}$ we employ specific task corpus, including XSum corpus (Abacha and Demner-Fushman, 2019) for summarization task, general conterfact corpus[7] for counterfactual task. For $C_{(L,D)_{\text{Exp}}}$, we employ the corresponding datasets in other tasks, including MedQCQ (Pal et al., 2022) for MedSum, AMSA (Zhang et al., 2015) for AMSum and AMContFact.

[7] https://huggingface.co/datasets/azhx/counterfact-easy

Experiment Details Hyperparameters, including the sizes of $C_{T_{\text{Exp}}}$ and $C_{(L,D)_{\text{Exp}}}$, are determined using the validation set of the Amazon counterfactual dataset and then applied to testsets in other multitask setting datasets. Furthermore, accuracy is the metric used for ARC-c, GSM8K, MMLU, and AMContFact, while Rouge-L (Lin, 2004) is used for MedSum and AMSum.