

# Adaptive LoRA Merge with Parameter Pruning for Low-Resource Generation

**Ryota Miyano**

Grad. Sch. of Information Science and Tech.  
Osaka University  
Japan  
miyano.ryota@ist.osaka-u.ac.jp

**Yuki Arase**

School of Computing  
Institute of Science Tokyo  
Japan  
arase@c.titech.ac.jp

## Abstract

This study proposes a simple yet effective LoRA merge method to achieve LLM adaptation for low-resource language generation tasks. The LoRA merge technique, which integrates multiple LoRA modules trained on different tasks, has gained attention as an effective and efficient approach for adapting LLMs to target tasks. However, previous methods are limited in adaptability as they keep the LoRA parameters frozen. Additionally, the low-resource problem has been out of their scope. We propose a LoRA merge method that updates and prunes LoRA parameters through fine-tuning with minimal target task data, which allows finer-grained adjustments of LoRA parameters and enhancement of task adaptability. Extensive experiments have been conducted taking summarization as a benchmark task. Our datasets cover various domains and multiple languages of English and Japanese. The results confirm that the proposed method achieves significant and consistent improvements in task adaptability over the previous methods.

## 1 Introduction

The rapid advancements in Large Language Models (LLMs) have significantly enhanced text generation capabilities and performance across tasks such as translation, summarization, question answering, and code generation (Zhao et al., 2024a; Raiaan et al., 2024; Minaee et al., 2024; Qin et al., 2024). However, LLMs often struggle with low-resource tasks, including those involving languages with scarce linguistic resources, specialized programming languages, or tasks in medical and other specialized domains (Nasution and Onan, 2024; Shen et al., 2024; Cassano et al., 2024; Singhal et al., 2023). This performance degradation arises from the insufficient adaptation of LLMs to target tasks, despite their general knowledge obtained during pretraining. Fine-tuning is a common method to enhance task-specific performance (Minaee et al.,

2024; Han et al., 2024), but its effectiveness is often constrained by limited training data in low-resource problems (Khade et al., 2025; Yang et al., 2024; To et al., 2024).

An alternative approach gaining attention is the integration of multiple models, particularly using LoRA modules (Hu et al., 2022; Mao et al., 2025; Huang et al., 2024). For instance, combining a model with general language capabilities and another specialized in a specific task can improve performance on target tasks. Such LoRA merge technique linearly combines LoRA modules into a single model. Existing studies (Zhao et al., 2024b; Huang et al., 2024; Wu et al., 2024; Wang et al., 2024) typically keep module parameters fixed and only adjust their combination weights, which reduces training costs. However, we assume it limits adaptability to the target task. Furthermore, low-resource tasks have been out of their scope.

To effectively adapt LLMs on low-resource language generation tasks, we propose a novel LoRA merge method that further updates LoRA modules with minimal target task data while pruning ineffective parameters. Previous studies have reported that each decoder layer in LLMs plays a different role in language generation (Wendler et al., 2024). Furthermore, analyses of LoRA modules trained on multiple tasks suggest that these modules learn task-specific representations that vary across layers (Wu et al., 2024). These findings inspired us to hypothesize that LoRA parameters may require finer-grained adjustments at different layers to better adapt to a target task. Based on this hypothesis, our method evaluates the importance of each LoRA parameter at each layer while pruning away ineffective ones and retraining them in order to enhance task adaptability.

We conducted extensive experiments to evaluate and analyze the proposed method taking summarization as a benchmark task. Our datasets cover various domains of news, scientific papers, and ra-

diology reports in multiple languages of English and Japanese. The results confirm that updating LoRA modules during the merge process improves task adaptability. In addition, pruning ineffective parameters further enhances the performance.

The primary contributions of this study are twofold. First, our simple LoRA merge technique achieves effective LLM adaptation to low-resource tasks across various domains and multiple languages with a minimum amount of target-task data. Second, we show that LoRA parameter pruning enhances the task adaptability of LLMs, which is a novel feature of the pruning technique that often degrades the performance in exchange for the reduction of active parameters. The codes are available at [https://github.com/mr0223/adaptive\\_lora\\_merge](https://github.com/mr0223/adaptive_lora_merge).

## 2 Related Work

This section discusses the previous LoRA merge techniques. In addition, we review studies on LLM layer analysis that inspired us to conduct parameter pruning during the LoRA merging process.

**LoRA Merge.** Several studies have investigated methods for combining multiple LoRA modules to facilitate multi-task learning. Early approaches employed static integration strategies, such as averaging module outputs or using fixed, manually designed weights (Sun et al., 2023; Smith et al., 2023). While these methods are computationally efficient, they often lack flexibility and struggle to adapt to tasks that differ significantly from those seen during training. LoRAHub (Huang et al., 2024) addresses this limitation by optimizing integration weights while keeping the original LoRA modules frozen. Task-specific LoRA modules are pre-trained on approximately 200 tasks, and gradient-free optimization is applied to tune the integration weights based on a small number of target task examples. This data-efficient approach allows low-resource task adaptation. However, because LoRAHub relies solely on adjusting integration weights and keeping the LoRA modules frozen, its capacity to handle tasks that are highly distinct from the pre-training tasks is limited.

The proposed method builds on these approaches by overcoming their limitations. Instead of relying solely on weights to combine frozen pre-trained modules, we directly update LoRA modules through target-task training with pruning for finer-grained adjustments of LoRA parameters.

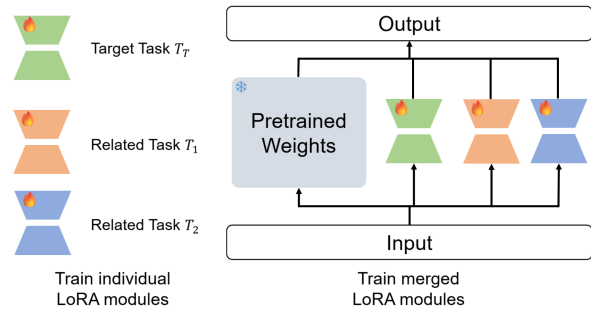


Figure 1: Two-stage training of LoRA modules: individual training on related tasks followed by fine-tuning with parameter pruning on a target task.

**LLM Layer Analysis.** Emergent analyses of LLM layers have shown that different layers of language models play specialized roles in processing input data. Wendler et al. (2024) analyzed the Llama 2 model (GenAI, 2023) and discussed that the layers conduct hierarchical processing to understand input texts. This hierarchical processing indicates that each layer contributes distinctively to tasks such as contextual understanding and language generation. Wu et al. (2024) further investigated layer-specific characteristics in multi-task learning models utilizing LoRA modules. They found that middle layers are more effective for simpler reasoning tasks, while upper layers are better suited to complex reasoning tasks. Based on these observations, they proposed Mixture of LoRA Experts (MoLE) to improve the performance of multi-task learning. MoLE dynamically adjusts the integration of frozen LoRA modules by modifying module weights for each layer, and further, for each input text. MoLE enhances the multi-task learning performance; however, it assumes that abundant training data is available for the target task. These studies inspired us to employ parameter pruning during LoRA merge to achieve finer-grained adjustments of LoRA modules for each LLM layer.

## 3 Adaptive LoRA Merge with Pruning

The proposed method achieves effective adaptation to a low-resource target task through training and pruning of LoRA parameters. Figure 1 illustrates the overview of the training procedure in the proposed method. The proposed method applies multiple LoRA modules trained on related tasks to a frozen LLM and further trains them on a target task (Section 3.1). During this process, the importance of LoRA parameters is evaluated at each decoder layer, and the parameters with lower impor-

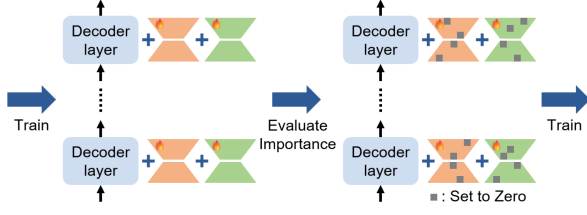


Figure 2: Pruning of LoRA parameters.

tance are pruned and retrained (Section 3.2). We remark that the proposed method does not explicitly ‘merge’ LoRA parameters; rather, our merging process is implicit through updates and pruning of all the original LoRA parameters.

### 3.1 Fine-Tuning of LoRA Modules

First of all, individual LoRA modules are trained independently to learn related tasks on a frozen LLM. Then the proposed method adaptively merges these LoRA modules with further training.

LoRA decomposes the weight update matrix of LLM,  $\Delta W$ , into two low-rank matrices,  $A$  and  $B$ , such that  $\Delta W = BA$ . We denote a LoRA module trained on a small set of target task data as  $B^T A^T$ , while we denote other  $N$  LoRA modules trained on  $N$  related tasks as  $B_1 A_1, B_2 A_2, \dots, B_N A_N$ . These modules are merged and then applied to the LLM parameters  $W_0$ , forming a new model parameterized as  $W_0 + B^T A^T + B_1 A_1 + \dots + B_N A_N$ . This model is fine-tuned using the target task data, with the LLM parameters frozen. The final parameters become  $W_0 + \hat{B}^T \hat{A}^T + \hat{B}_1 \hat{A}_1 + \dots + \hat{B}_N \hat{A}_N$ , where  $\hat{B}^T$ ,  $\hat{A}^T$ ,  $\hat{B}_i$  and  $\hat{A}_i$  ( $i = 1, 2, \dots, N$ ) are the fine-tuned LoRA parameters on the target task.

Note that the proposed method does not necessarily require  $B^T$  and  $A^T$ . It can instead rely on  $N$  LoRA modules trained on other tasks. The effect of the target task LoRA is examined in our experiments.

### 3.2 Pruning of Ineffective LoRA Parameters

Figure 2 illustrates our pruning process. During the training of merged LoRA modules, the importance of LoRA parameters is evaluated at each decoder layer, and ineffective parameters are pruned away at each training step. Algorithm 1 shows a pseudocode of this process. After gradient calculation and parameter updates, parameters are evaluated for their importance. Ineffective parameters are pruned and then retrained at the next step.

**Parameter Importance** Following (Sun et al., 2024; Dettmers et al., 2022), we evaluate the impor-

---

#### Algorithm 1 Adaptive LoRA merge with pruning

---

**Input:** Training and validation sets of target task  $\mathcal{D}_t$  and  $\mathcal{D}_v$ , LLM  $\mathcal{M}$  with frozen parameters  $W_0$  and pre-trained LoRA modules  $\mathcal{R}^{(0)}$

**Output:** LoRA modules with target task adaptation and pruning:  $\hat{\mathcal{R}}^{(n)}$

**repeat**

    Sample mini-batch  $b_i$  from  $\mathcal{D}_t$  for step  $i$

$\mathcal{L} \leftarrow \mathcal{M}(b_i)$  ▷ Compute loss

    Compute gradients, backward loss  $\mathcal{L}$

$\mathcal{R}^{(i)} \leftarrow \text{update}(\hat{\mathcal{R}}^{(i-1)})$  ▷ Update LoRA

$\mathcal{E}^{(i)} \leftarrow \text{eval}(\mathcal{R}^{(i)}, \mathcal{D}_v)$  ▷ Eval. importance

$\hat{\mathcal{R}}^{(i)} \leftarrow \text{prune}(\mathcal{R}^{(i)}, \mathcal{E}^{(i)})$  ▷ Pruning

$\mathcal{M} \leftarrow W_0, \hat{\mathcal{R}}^{(i)}$  ▷ Apply pruned LoRA

**until** converge

---

tance of LoRA parameters based on the magnitude of parameter weights and inputs as illustrated in Figure 3. Sun et al. (2024) empirically showed that not only the magnitude of parameters but also that of input activations should be considered because the scale of input features can significantly differ in LLMs. The importance is defined as the product of the absolute value of a parameter weight  $W_{ij}$  and the  $L_2$  norm of the corresponding input features:

$$I(W_{ij}) = |W_{ij}| \cdot \|X_j\|_2$$

where  $|\cdot|$  computes the absolute value and  $\|X_j\|_2$  is the  $L_2$  norm of the associated input feature  $X_j$ . The proposed method uses a validation set to compute the input features.

**Pruning Strategy** Low-importance parameters are pruned using a **zeroing** strategy; the weights of these parameters are set to zero and trained again in the next training step. This approach allows resetting parameters negatively affecting the target task performance and tuning them again, expecting they to learn better weights in the next step.

We conduct pruning at the parameter level, i.e., evaluating each parameter weight in a LoRA module individually and zeroing out low-importance ones. This approach is suitable when weight importance varies significantly within a LoRA module, as reported in (Dettmers et al., 2022). Sun et al. (2024) showed that parameter-wise pruning allows for retaining useful components while removing unnecessary sub-parameters. This can mitigate performance degradation due to excessive pruning by processing an entire module as a whole.

Weights are pruned based on a predefined ratio

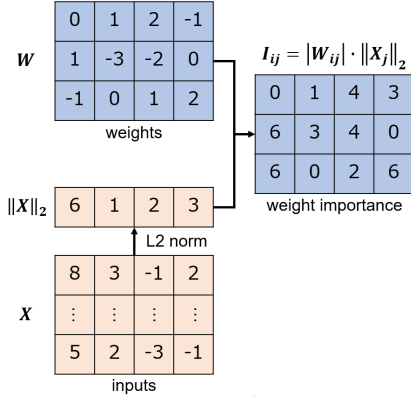


Figure 3: Importance calculation of LoRA parameters

$s\%$ ; the lowest  $s\%$  parameters in terms of importance are zero-out. As each LoRA module has been individually trained on different tasks, the distributions of parameter weights can vary across modules. Therefore, we compare the importance of parameters per each module rather than across modules. The pruning ratio is treated as a hyperparameter and optimized using validation data.

## 4 Experiment Settings

We evaluate the capability of the proposed method for adapting an LLM for low-resourced target tasks. Intensive experiments are conducted using abstractive summarization as a benchmark task employing datasets of various domains of news, scientific papers, and radiology reports in multiple languages of English and Japanese.

### 4.1 Dataset

This section provides an overview of the datasets used in our experiments, categorized into target and related tasks. The experiments cover both English and Japanese tasks. The English tasks are summarization of radiology reports and scientific papers. The Japanese target tasks are summarization of research papers and news articles. Table 1 lists the number of data samples for each dataset. Details on the construction and preprocessing of the target task datasets are provided in Appendix A.

#### 4.1.1 Related Tasks

We employed publicly available multilingual summarization datasets for pretraining LoRA modules of related tasks.

**XLSum** The XLSum dataset (Hasan et al., 2021) is a multilingual news summarization dataset constructed from BBC news articles. Both the

Dataset	Train	Val	Test
Related task			
XLSum (en)	306,522	11,535	11,535
XLSum (ja)	7,113	889	889
WikiLingua (en)	98,999	13,819	28,607
WikiLingua (ja)	8,852	1,264	2,529
Target task			
MIMIC-III (en)	44,342	5,550	10,996
SciTLDR (en)	1,992	619	618
Bloomberg (ja)	9,656	1,207	1,207
NLP Paper (ja)	312	100	100
Medical Paper (ja)	183	100	100

Table 1: Number of sentences in datasets

Japanese and English subsets are used in our experiments. Summaries are extracted from the lead sentences of the articles, which concisely present the main content of reported news.

**WikiLingua** The WikiLingua dataset (Ladhak et al., 2020) is a multilingual resource derived from WikiHow guides. Input documents consist of concatenated step explanations, while output summaries are formed by combining step headings. We use both the Japanese and English subsets.

#### 4.1.2 Target Tasks

For English tasks, we used two publicly available datasets distinct from the XLSum and WikiLingua domains. For Japanese, there is no available dataset for summarization other than XLSum and WikiLingua. Therefore, we created datasets for our experiments.

**MIMIC-III** The MIMIC-III dataset (Johnson et al., 2016) is used for the English radiology report summarization task. Each report consists of three main sections: background, findings, and impressions. The findings section serves as the input, and the impressions section, summarizing key observations, serves as the output.

**SciTLDR** The SciTLDR dataset (Cachola et al., 2020) is used for the English scientific paper summarization task. It contains short summaries (TLDRs) created by authors and reviewers. The input consists of the abstract, introduction, and conclusion (AIC) sections, enabling the generation of highly compressed summaries.

**Bloomberg** We crawled Bloomberg Japanese articles using the URL list provided by the MassiveSumm project (Varab and Schluter, 2021).



Bloomberg articles have bullet-point highlights that summarize the contents. We extracted them as ground-truth summaries combined with article titles. The full article serves as the input document to summarize. Remarkably, our way of dataset construction is different from that of XLSum utilizing lead sentences as summaries, to ensure that all the content in a summary exists in the input document. This difference makes Bloomberg task as distinct from XLSum, although the domain is the same.

**NLP/Medical Paper** Two datasets were created from research papers on natural language processing and medical case reports. The task is generating titles from the corresponding abstracts as short summaries. The NLP paper dataset was built from the LaTeX corpus of the Journal of Natural Language Processing<sup>1</sup>, extracting titles and abstracts. The medical paper dataset was constructed from case reports published on J-STAGE<sup>2</sup>, covering articles with diverse abstract formats.

#### 4.1.3 Evaluation Metrics

The Bloomberg, MIMIC-III, and SciTLDR tasks were evaluated using ROUGE (Lin, 2004)<sup>3</sup>, while the NLP/Medical paper tasks were evaluated using BLEU (Papineni et al., 2002)<sup>4</sup> due to their shorter summaries. For Japanese tasks, we employed the Mecab (Kudo et al., 2004) for word segmentation. Additionally, statistical significance was assessed using approximate randomization testing (Riezler and Maxwell, 2005).

## 4.2 Baselines

We used the following baselines for comparison:

1. **Zero-shot**: Summarization using an LLM without additional training.
2. **LoRA (XS) / LoRA (WL)**: Summarization directly using LoRA modules trained on the related tasks of XLSum and WikiLingua, respectively.
3. **LoRA (TGT)**: Summarization directly using LoRA modules trained on the target tasks.

Additionally, we compare to LoRAHub, a strong baseline for LoRA merging. LoRAHub involves

merging LoRA modules from related tasks (denoted as “**LoRAHub (XS+WL)**”) and further merging with the target task module (denoted as “**LoRAHub (XS+WL+TGT)**”). We reproduced LoRAHub based on its official Codes<sup>5</sup>, making modifications to support Llama-3.

## 4.3 Implementation

We evaluate variations of the proposed method to investigate the effects of LoRA fine-tuning on target tasks and parameter pruning of the proposed method:

1. **Ours Merge**: Conducts only fine-tuning of LoRA modules on target tasks.
2. **Ours Merge+Del**: Conducts both LoRA fine-tuning and parameter pruning.

In Ours Merge+Del, the deletion ratio was treated as a hyperparameter and optimized based on the evaluation metrics measured on the validation data using grid-search.

For all the methods compared, we employed Llama-3-8B-Instruct (Team, 2024)<sup>6</sup> as the base model for its strong performance on various language tasks. The same prompt design was used for both LoRA module training and output generation. We designed simple yet effective prompts tailored to each task to enhance learning and improve output quality. The prompt details are provided in Appendix B.

## 4.4 Training and Inference

For training on the target tasks, 50 instances were randomly subsampled for both training and validation sets, respectively, to replicate the low-resource scenario. These small subsets were used for training and validating all the methods compared. LoRA modules for the related tasks were trained using all available training sets. The training was stopped early based on the validation loss measured at each epoch. The model with the lowest validation loss was saved as the final model. Details on LoRA module training parameters are in Appendix B.

For testing, all the test set samples were used. At inference time, a summary was generated employing greedy decoding.

<sup>1</sup>[https://www.anlp.jp/resource/journal\\_latex/](https://www.anlp.jp/resource/journal_latex/)

<sup>2</sup><https://www.jstage.jst.go.jp/>

<sup>3</sup><https://github.com/google-research/google-research/tree/master/rouge>

<sup>4</sup><https://github.com/mjpost/sacrebleu>

<sup>5</sup><https://github.com/sail-sg/lorahub>

<sup>6</sup><https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct>

	MIMIC-III		SciTLDR		Bloomberg		NLP Paper		Medical Paper	
	RL	Del%	RL	Del%	RL	Del%	BLEU	Del%	BLEU	Del%
Zero-shot	16.64	-	29.58	-	0.91	-	2.73	-	5.26	-
LoRA (XS)	18.95	-	24.76	-	21.39	-	12.26	-	16.92	-
LoRA (WL)	16.23	-	33.23	-	26.77	-	18.89	-	23.71	-
LoRA (TGT)	27.97	-	35.02	-	25.64	-	21.09	-	30.95	-
LoRAHub (XS+WL)	18.83	-	33.92	-	27.11	-	18.54	-	23.66	-
LoRAHub (XS+WL+TGT)	27.90	-	<b>35.63</b> <sup>†</sup>	-	28.13 <sup>†</sup>	-	21.00	-	26.93	-
Ours <sub>Merge</sub> (XS+WL)	<b>28.92</b> <sup>†</sup>	-	<b>35.95</b> <sup>†</sup>	-	31.94 <sup>†</sup>	-	22.37 <sup>†</sup>	-	32.36 <sup>†</sup>	-
Ours <sub>Merge</sub> (XS+WL+TGT)	<b>29.13</b> <sup>†</sup>	-	35.43	-	31.79 <sup>†</sup>	-	22.46 <sup>†</sup>	-	30.86	-
Ours <sub>Merge+Del</sub> (XS+WL)	<b>28.75</b> <sup>†</sup>	30	<b>35.91</b> <sup>†</sup>	30	<b>32.91</b> <sup>†</sup>	40	<b>23.28</b> <sup>†</sup>	50	32.57 <sup>†</sup>	20
Ours <sub>Merge+Del</sub> (XS+WL+TGT)	<b>28.96</b> <sup>†</sup>	60	<b>35.99</b> <sup>†</sup>	60	<b>33.12</b> <sup>†</sup>	30	<b>23.04</b> <sup>†</sup>	30	<b>34.04</b> <sup>†</sup>	30

Table 2: Results on five summarization tasks of various domains and multiple languages. The best scores (scores with no significant difference from the highest ones) are marked by bold fonts, and <sup>†</sup> indicates a significant difference against LoRA (TGT).

#### 4.5 Ablation Study

We conducted an ablation study to investigate the effectiveness of our design of (a) parameter importance estimation, (b) pruning unit, and (c) pruning value. For (a), we compare our importance calculation method to the one proposed by Zhang et al. (2022), which is based on magnitudes of parameter weights and gradients. For (b), we compare parameter-wise pruning to module-wise deletion and reinitialization. For (c), we examine a method that resets the parameters of pruned modules to their initial values. Further details on these variations are provided in Appendix C.

### 5 Experiment Results

Experiments were conducted independently with three different random seeds, and the results are reported as the average across these runs.

#### 5.1 Main Results

Table 2 shows the results of the proposed method and baselines for the 5 summarization tasks in English and Japanese.<sup>7</sup> Remarkably, our method consistently outperforms LoRA and LoRAHub in most tasks across domains and languages. Comparing Ours<sub>Merge</sub> and Ours<sub>Merge+Del</sub>, Ours<sub>Merge+Del</sub> achieves higher performance in 4 tasks and comparable results in MIMIC-III. These results clearly confirm the effectiveness of the adaptive LoRA merge that further trains LoRA parameters during merging while pruning ineffective parameters. It is noteworthy that the performance gain

<sup>7</sup>BERTScore (Zhang et al., 2020) results, which show the consistent trends with ROUGE/BLEU scores, are also reported in Appendix D.

over LoRAHub is more pronounced on Japanese tasks (Bloomberg, NLP Paper, and Medical Paper), which is another advantage of the proposed method.

On Ours<sub>Merge+Del</sub>, merging both modules of related and target tasks showed marginal improvements over merging only LoRA modules of related tasks for most datasets. We suspect this is because the LoRA modules of related tasks can adapt to the target task through the training during merging. The LoRA module of the target task was significantly effective on the Medical Paper dataset, which may imply domain differences matter. Further investigation constitutes our future work.

Table 3 shows the generated summaries along with a reference. The proposed methods explicitly mention the key innovation, “**community-based autoencoders**”. While Ours<sub>Merge</sub> captures this concept, its description remains vague. Ours<sub>Merge+Del</sub>, however, provides a clearer and more informative summary. In contrast, LoRA and LoRAHub generated an overly generalized description of “*inspired by the way humans learn to communicate,*” which shifts the meaning of “*Motivated by theories of language and communication.*” In addition, they failed to describe the technological novelty, resulting in less sensible summaries for the input paper.

#### 5.2 Ablation Study Results

This section presents the ablation study results on different pruning strategies with the Japanese tasks. Table 4 summarizes the model performance measured on the test sets under various pruning configurations: parameter importance calculation method (**Grad**: magnitudes of parameter weights and gradients; **Input**: magnitudes of parameter weights and

Abstract	Good representations facilitate transfer learning and few-shot learning. <i>Motivated by theories of language and communication that explain why communities with large number of speakers have, on average, simpler languages with more regularity, [...] Generalizing from there, we introduce <b>community-based autoencoders</b> in which <b>multiple encoders and decoders collectively learn representations</b> by being randomly paired up on successive training iterations. Our experiments show that [...]</i>
Reference	<i>Motivated by theories of language and communication, we introduce <b>community-based autoencoders</b>, in which <b>multiple encoders and decoders collectively learn structured and reusable representations</b>.</i>
Ours <sub>Merge+Del</sub> (XS+WL+TGT)	We introduce <b>community-based autoencoders</b> , a framework in which <b>multiple encoders and decoders collectively learn representations</b> by being randomly paired up on successive training iterations.
Ours <sub>Merge</sub> (XS+WL+TGT)	<b>Community-based autoencoders learn more reusable and structured representations.</b>
LoRAHub (XS+WL+TGT)	We introduce a new framework for learning representations that is <i>inspired by the way humans learn to communicate</i> .
LoRA (TGT)	We introduce a new framework for learning representations that is <i>inspired by the way humans communicate</i> and learn from each other.

Table 3: Case study of the predicted output of different models (SciTLDR).

		Bloomberg			NLP Paper			Medical Paper		
		RL	Thresh	Del%	BLEU	Thresh	Del%	BLEU	Thresh	Del%
Ours <sub>Merge</sub> (XS+WL+TGT)		31.79	–	–	22.46	–	–	30.86	–	–
Input	Zero	32.01	10e-3	39.06	<b>23.24</b>	6e-3	33.33	31.40	4e-3	33.33
	Init	31.43	8e-3	33.33	<b>23.05</b>	6e-3	33.33	<b>33.59</b>	4e-3	33.33
Grad	Zero	31.78	2e-13	25.52	22.74	5e-13	25.52	33.25	2e-13	35.94
	Init	32.21	7e-13	58.33	22.52	4e-13	17.71	<b>33.87</b>	3e-13	42.19
Input	Zero	<b>33.12</b>	–	30.00	<b>23.04</b>	–	30.00	<b>34.04</b>	–	30.00
	Init	<b>33.25</b>	–	40.00	<b>23.16</b>	–	40.00	<b>33.96</b>	–	60.00
Grad	Zero	32.49	–	10.00	22.19	–	10.00	32.60	–	20.00
	Init	32.42	–	30.00	<b>22.87</b>	–	60.00	32.73	–	50.00

Table 4: Performance difference of Ours<sub>Merge+Del</sub> (XS+WL+TGT) under pruning strategy variations measured on test sets of Japanese Tasks. The best scores (scores with no significant difference from the highest ones) are marked by bold fonts.

inputs), pruning unit (**Module**: module-level pruning; **Parameter**: parameter-level pruning), and pruning values (**Init**: initialization; **Zero**: zeroing out). A baseline without pruning (Ours<sub>Merge</sub> (XS+WL+TGT)) is also included. The pruning threshold (“Thresh” column) represents the importance score threshold used for module-level pruning. Module-level pruning prunes modules whose average parameter importance score is below the threshold. All parameters in a pruned module were reset. This threshold was treated as a hyperparameter and optimized using validation data. In contrast, parameter-level pruning prunes  $s\%$  parameters of lowest importance scores as shown in the “Del %”

column.

The results indicate that Input, which evaluates parameter importance based on magnitudes of parameter weights and inputs, and Parameter, which conducts parameter-level pruning, consistently achieve higher performance than their counterparts. For resetting values on pruning, both methods worked comparably. It is noteworthy that pruning with inferior configurations still improved upon the baseline without pruning, which confirms that pruning is crucial in our method.

To further analyze the effects of pruning configurations, we examine the relationship between pruning hyperparameters and model performance. Fig-

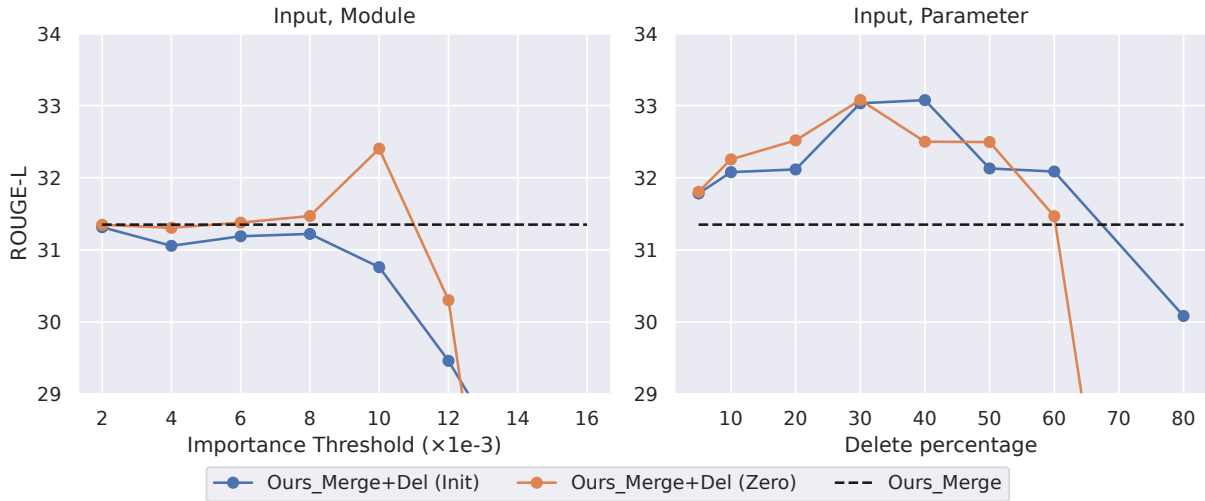


Figure 4: Impact of pruning hyperparameters on model performance (validation set of Bloomberg)

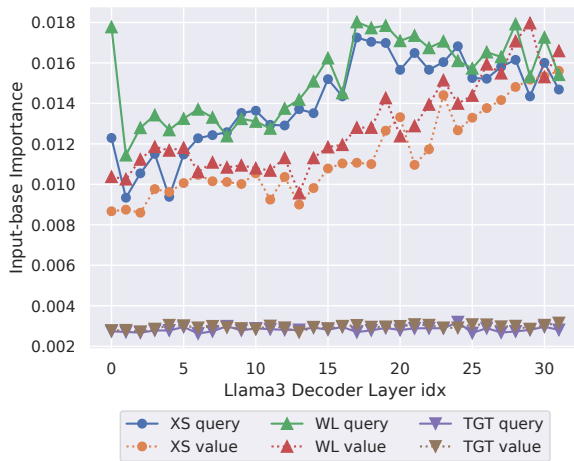


Figure 5: Distribution of Module-wise importance based on Input (Bloomberg, Ours<sub>Merge+Del</sub> (XS+WL+TGT)).

Figure 4 shows the impact of the pruning thresholds on Ours<sub>Merge+Del</sub> (XS+WL+TGT) with module-level (Module) or parameter-level (Parameter) pruning measured on the validation set of Bloomberg. The parameter importance was evaluated based on the magnitudes of parameter weights and inputs (Input). The graph of the parameter-level pruning (right) shows a bell-like shape, i.e., the performance initially improves as ineffective parameters are pruned and then decreases when pruning becomes excessive. In contrast, the graph of module-level pruning (left) exhibits that the performance hardly outperforms the baseline, which indicates that module-level pruning is too coarse-grained and may result in removing effective parameters in these modules. Appendix D shows the graphs on the Grad configuration.

Figure 5 shows the module-wise importance dis-

tribution in different layers of LLM measured on the Bloomberg task, where the importance was calculated based on magnitudes of parameter weights and inputs. The importance scores of LoRA modules vary: LoRA modules of the target task range from 0.002 to 0.004 while those of related tasks range from 0.008 to 0.018. Also, the score range differs across layers, too. This result suggests two things. First, for parameter-level pruning, it is crucial to determine pruning parameters per module based on importance score rankings inside a module rather than the global, across-module ranking. This aligns with the previous study showing that module-wise importance ranking outperforms global or layer-level pruning in LLM parameter pruning (Sun et al., 2024). Second, module-level pruning has a risk of removing target task LoRA modules, which contradicts our expectation that effective parameters should be kept.

### 5.3 Effects of Size of Target Task Data

The previous sections evaluated the performance with a training dataset of 50 instances on the target task to simulate the low-resource scenario. In this section, we investigate the effects of the size of the target training set by varying the size: 5, 50, 100, and 200 instances on Bloomberg. Intuitively, the performance gain by the proposed method should shrink as the training data becomes larger.

The results are presented in Figure 6. As expected, the performance gain by the proposed method shrinks as the training set becomes larger. As the number of training instances increases, LoRA (TGT), trained only on the target task, improves significantly. Yet all the variations of the



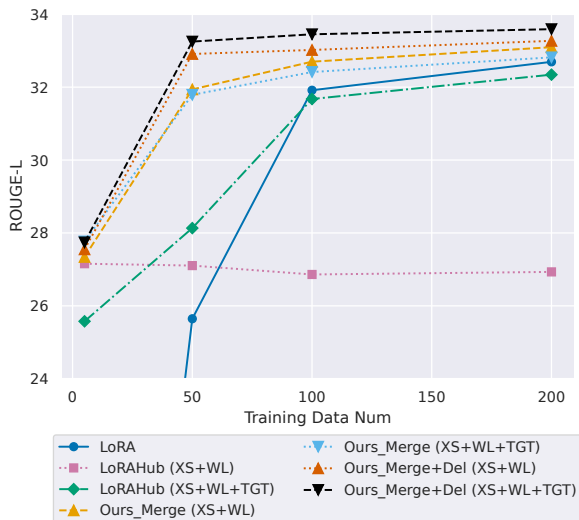


Figure 6: Effect of training data size on model performance (Bloomberg).

proposed method still achieve higher ROUGE-L scores across sizes, even at the largest training set, indicating that incorporating LoRA modules from related tasks is useful. Furthermore, the proposed method with merging and pruning,  $\text{Ours}_{\text{Merge+Del}}(\text{XS+WL})$  and  $\text{Ours}_{\text{Merge+Del}}(\text{XS+WL+TGT})$ , consistently outperformed the merging only methods,  $\text{Ours}_{\text{Merge}}(\text{XS+WL})$  and  $\text{Ours}_{\text{Merge}}(\text{XS+WL+TGT})$ , across all data sizes. This result again confirms the importance of parameter pruning while merging.

## 6 Conclusion

We proposed the adaptive merging method for multiple LoRA modules to improve LLMs in low-resource tasks. Experiments on the five English and Japanese summarization tasks show that our method significantly outperforms existing LoRA merging techniques across domains and languages.

Future work includes the application of the proposed method to broader tasks and cross-lingual settings. Additionally, we plan to evaluate its effectiveness across various LLMs of different sizes. Exploring the merging of more diverse and numerous LoRA modules is another important direction. Currently, the proposed method requires tuning the pruning threshold for each task. Automating this process would enhance the practicality of our method.

## Limitations

Our method conducts LoRA training twice: once to pre-train them for related tasks and another to

merge, leading to increased training time. Although the merging step on the target task is efficient, as we assume the low-resource scenario (in our experiments, we used just 50 instances), the overall cost remains a concern. This could be mitigated by leveraging publicly available pre-trained LoRA adapters.

We experimented with summarization tasks in English and Japanese, but summarization itself was monolingual. It is worth investigating the applicability of the proposed method to cross-lingual tasks.

Another limitation is that the proposed method requires tuning the hyperparameter of the pruning ratio, which should be adjusted depending on the datasets. Future work should explore automatic methods to determine this hyperparameter.

## Acknowledgments

We sincerely appreciate the anonymous reviewers for their insightful comments and suggestions to improve the paper. This work was supported by the Cross-ministerial Strategic Innovation Promotion Program (SIP) on “Integrated Health Care System” Grant Number JPJ012425.

## References

- Isabel Cachola, Kyle Lo, Arman Cohan, and Daniel Weld. 2020. [TLDR: Extreme Summarization of Scientific Documents](#). In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4766–4777.
- Federico Cassano, John Gouwar, Francesca Lucchetti, Claire Schlesinger, Anders Freeman, Carolyn Jane Anderson, Molly Q Feldman, Michael Greenberg, Abhinav Jangda, and Arjun Guha. 2024. [Knowledge transfer from high-resource to low-resource programming languages for code llms](#). *Proceedings of the ACM on Programming Languages*.
- Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. [LLM.int8\(\): 8-bit matrix multiplication for transformers at scale](#). In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*.
- Meta GenAI. 2023. [Llama 2: Open foundation and fine-tuned chat models](#). *arXiv preprint arXiv:2307.09288*.
- Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. 2024. [Parameter-efficient fine-tuning for large models: A comprehensive survey](#). *arXiv preprint arXiv:2403.14608*.
- Tahmid Hasan, Abhik Bhattacharjee, Md Saiful Islam, Kazi Samin, 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*, pages 4693–4703.
- Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. **LoRA: Low-Rank Adaptation of Large Language Models**. In *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. 2024. **LoraHub: Efficient Cross-Task Generalization via Dynamic LoRA Composition**. In *Proceedings of the Conference on Language Modeling (COLM)*.
- Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. **Mimic-iii, a freely accessible critical care database**. *Scientific data*, 3(1):1–9.
- Omkar Khade, Shruti Jagdale, Abhishek Phaltankar, Gauri Takalikar, and Raviraj Joshi. 2025. **Challenges in adapting multilingual llms to low-resource languages using lora peft tuning**. In *Proceedings of the First Workshop on Challenges in Processing South Asian Languages (CHiPSAL)*, pages 217–222.
- Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. 2004. **Applying conditional random fields to Japanese morphological analysis**. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 230–237.
- Faisal Ladhak, Esin Durmus, Claire Cardie, and Kathleen McKeown. 2020. **WikiLingua: A New Benchmark Dataset for Cross-Lingual Abstractive Summarization**. In *Findings of the Association for Computational Linguistics: EMNLP*, pages 4034–4048.
- Chen Liang, Simiao Zuo, Minshuo Chen, Haoming Jiang, Xiaodong Liu, Pengcheng He, Tuo Zhao, and Weizhu Chen. 2021. **Super Tickets in Pre-Trained Language Models: From Model Compression to Improving Generalization**. In *Proceedings of the Joint Conference of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 6524–6538.
- Chin-Yew Lin. 2004. **ROUGE: A Package for Automatic Evaluation of Summaries**. In *Text Summarization Branches Out*, pages 74–81.
- Yuren Mao, Yuhang Ge, Yijiang Fan, Wenyi Xu, Yu Mi, Zhonghao Hu, and Yunjun Gao. 2025. **A survey on lora of large language models**. *Frontiers of Computer Science*, 19(7).
- Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. **Large language models: A survey**. *arXiv preprint arXiv:2402.06196*.
- Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. 2019. **Importance Estimation for Neural Network Pruning**. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Arbi Haza Nasution and Aytuğ Onan. 2024. **Chatgpt label: Comparing the quality of human-generated and llm-generated annotations in low-resource language nlp tasks**. *IEEE Access*, 12:71876–71900.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. **Bleu: a method for automatic evaluation of machine translation**. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318.
- Libo Qin, Qiguang Chen, Xiachong Feng, Yang Wu, Yongheng Zhang, Yinghui Li, Min Li, Wanxiang Che, and Philip S Yu. 2024. **Large language models meet nlp: A survey**. *arXiv preprint arXiv:2405.12819*.
- Mohaimenul Azam Khan Raiaan, Md. Saddam Hossain Mukta, Kaniz Fatema, Nur Mohammad Fahad, Sadman Sakib, Most Marufatul Jannat Mim, Jubaer Ahmad, Mohammed Eunus Ali, and Sami Azam. 2024. **A review on large language models: Architectures, applications, taxonomies, open issues and challenges**. *IEEE Access*, 12:26839–26874.
- Stefan Riezler and John T. Maxwell. 2005. **On Some Pitfalls in Automatic Evaluation and Significance Testing for MT**. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 57–64.
- Lingfeng Shen, Weiting Tan, Sihao Chen, Yunmo Chen, Jingyu Zhang, Haoran Xu, Boyuan Zheng, Philipp Koehn, and Daniel Khashabi. 2024. **The language barrier: Dissecting safety challenges of llms in multilingual contexts**. In *Findings of the Association for Computational Linguistics: ACL*, pages 2668–2680.
- Karan Singhal, Shekoofeh Azizi, Tao Tu, S. Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, Perry Payne, Martin Seneviratne, Paul Gamble, Chris Kelly, Nathaneal Scharli, Aakanksha Chowdhery, Philip Mansfield, Blaise Aguerre y Arcas, Dale Webster, Greg S. Corrado, Yossi Matias, Katherine Chou, Juraj Gottweis, Nenad Tomasev, Yun Liu, Alvin Rajkomar, Joelle Barral, Christopher Semturs, Alan Karthikesalingam, and Vivek Natarajan. 2023. **Large language models encode clinical knowledge**. *Nature*, 620(7972):172–180.
- James Seale Smith, Paola Cascante-Bonilla, Assaf Arbelle, Donghyun Kim, Rameswar Panda, David Cox, Diyi Yang, Zsolt Kira, Rogerio Feris, and Leonid Karlinsky. 2023. **Construct-vl: Data-free continual structured vl concepts learning\***. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14994–15004.

- Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2024. [A Simple and Effective Pruning Approach for Large Language Models](#). In *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Yuhan Sun, Mukai Li, Yixin Cao, Kun Wang, Wenxiao Wang, Xingyu Zeng, and Rui Zhao. 2023. [To be or not to be? an exploration of continuously controllable prompt engineering](#). *arXiv preprint arXiv:2311.09773*.
- Llama Team. 2024. [The Llama 3 Herd of Models](#). *arXiv preprint arXiv:2407.21783*.
- Huy Quoc To, Ming Liu, and Guangyan Huang. 2024. [DeakinNLP at BioLaySumm: Evaluating Fine-tuning Longformer and GPT-4 Prompting for Biomedical Lay Summarization](#). In *Proceedings of the Workshop on Biomedical Natural Language Processing (BioNLP)*, pages 748–754.
- Dave Van Veen, Cara Van Uden, Maayane Attias, Anuj Pareek, Christian Bluethgen, Malgorzata Polacin, Wah Chiu, Jean-Benoit Delbrouck, Juan Zambrano Chaves, Curtis Langlotz, Akshay Chaudhari, and John Pauly. 2023. [RadAdapt: Radiology Report Summarization via Lightweight Domain Adaptation of Large Language Models](#). In *The Workshop on Biomedical Natural Language Processing and BioNLP Shared Tasks*, pages 449–460.
- Daniel Varab and Natalie Schluter. 2021. [MassiveSumm: a very large-scale, very multilingual, news summarisation dataset](#). In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 10150–10161.
- Hanqing Wang, Bowen Ping, Shuo Wang, Xu Han, Yun Chen, Zhiyuan Liu, and Maosong Sun. 2024. [Lora-flow: Dynamic lora fusion for large language models in generative tasks](#). In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 12871–12882.
- Chris Wendler, Veniamin Veselovsky, Giovanni Monea, and Robert West. 2024. [Do Llamas Work in English? On the Latent Language of Multilingual Transformers](#). In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 15366–15394.
- Xun Wu, Shaohan Huang, and Furu Wei. 2024. [Mixture of LoRA Experts](#). In *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Qimin Yang, Rongsheng Wang, Jiexin Chen, Runqi Su, and Tao Tan. 2024. [Fine-tuning medical language models for enhanced long-contextual understanding and domain expertise](#). *arXiv preprint arXiv:2407.11536*.
- Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023. [Adaptive Budget Allocation for Parameter-Efficient Fine-Tuning](#). In *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Qingru Zhang, Simiao Zuo, Chen Liang, Alexander Bukharin, Pengcheng He, Weizhu Chen, and Tuo Zhao. 2022. [PLATON: Pruning Large Transformer Models with Upper Confidence Bound of Weight Importance](#). In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 26809–26823.
- Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. [BERTScore: Evaluating text generation with bert](#). In *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2024a. [A survey of large language models](#). *arXiv preprint arXiv:2303.18223*.
- Ziyu Zhao, Leilei Gan, Guoyin Wang, Wangchunshu Zhou, Hongxia Yang, Kun Kuang, and Fei Wu. 2024b. [LoraRetriever: Input-Aware LoRA Retrieval and Composition for Mixed Tasks in the Wild](#). In *Findings of the Association for Computational Linguistics: ACL*, pages 4447–4462.

## A Details of Dataset Construction

This section details the construction processes for the Bloomberg, NLP/Medical Paper, and MIMIC-III datasets.

### A.1 Bloomberg Dataset

The Bloomberg dataset was constructed from Japanese news articles published on Bloomberg’s online platform. The primary goal was to create a task structurally distinctive from the XLSum task by utilizing article highlights as summaries instead of lead sentences. The dataset was constructed through the following steps:

1. **Article Collection:** We referred to the URL list provided by the MassiveSumm project (Varab and Schluter, 2021), which includes links to Bloomberg articles. Articles containing bullet-point highlights were identified and extracted for further processing.
2. **Highlight Extraction:** The bullet-point highlights, a key feature of Bloomberg articles, were automatically extracted using an HTML

parser. These highlights summarize the essential points of the article and were used as the basis for the output summaries.

3. **Title Combination:** To enhance coverage, the extracted highlights were combined with the article title. This combination ensures that the summary captures the main content more comprehensively, as the highlights alone may sometimes lack sufficient detail.
4. **Input Document Construction:** The full text of each article was extracted and used as the input document. This includes all relevant content except for metadata or sections not related to the main article text.

This construction approach differs from that used in other datasets, such as MassiveSumm and XLSum. While MassiveSumm extracts summaries from lead sentences, they may contain extraneous information not found in the main article. Our method leverages bullet-point highlights that are closely tied to the core content. This ensures a more accurate representation of the article and introduces structural variety between the target and related tasks.

## A.2 NLP/Medical Paper Dataset

We constructed two datasets for research paper summarization: one using NLP research papers and the other using medical case reports.

### A.2.1 NLP Paper Dataset

The NLP Paper dataset was created from the LaTeX corpus of the Journal of Natural Language Processing. The construction process involved the following steps:

1. **Document Extraction:** We extracted LaTeX source files from the corpus, selecting only papers written in Japanese.
2. **Title and Abstract Extraction:** The title was extracted from either the ‘jtitle’ or ‘title’ field, while the abstract was extracted from either the ‘jabstract’ or ‘abstract’ field.
3. **Preprocessing:** LaTeX-specific commands such as ‘\cite’ and ‘\vspace’ were removed.

Parameter	Value
LoRA Rank	8
LoRA Alpha	32
LoRA Dropout	0.05
Target Modules	Query, Value
Learning Rate	0.0001
Optimizer	AdamW
Batch Size	16
Epoch Num	40

Table 5: Parameters used for LoRA module training.

### A.2.2 Medical Paper Dataset

The Medical Paper dataset was constructed from case reports published on J-STAGE. The dataset construction involved:

1. **Document Collection:** Case reports from multiple journals were collected to cover diverse topics.
2. **Title and Abstract Extraction:** Titles and abstracts were extracted automatically from the structured metadata of each report.

### A.3 MIMIC-III Dataset Processing

For the MIMIC-III dataset, we extracted and processed radiology reports for the summarization task following the methodology proposed in RadAdapt (Van Veen et al., 2023). The procedure consisted of the following steps:

1. **Section Extraction:** We extracted the *Findings* and *Impressions* sections from raw radiology reports. The *Findings* section serves as the input, while the *Impressions* section, which provides a concise summary of key observations, serves as the output.
2. **Filtering:** To further refine the dataset, we applied an additional filtering step. Specifically, samples where the *Findings* section was shorter than or comparable in length to the *Impressions* section were removed, ensuring that the dataset aligns with the characteristics of a summarization task.

This filtering step improves dataset quality by ensuring that the input text contains more detailed information than the output summary, reinforcing a meaningful document-summarization relationship.



Dataset	Prompt
XLSum	Summarize the following Article in no more than three sentence. Article: {{article}}
WikiLingua	Summary: Summarize the following How-to Guide and write a one-sentence summary for each step: How-to Guide: {{article}}
Bloomberg	Summary: Summarize the following article in three sentences. Article: {{article}}
Title Generation	Read the following Abstract of a scientific paper and create an appropriate title that reflects the content. Please only output the Japanese title. Abstract: {{article}}
MIMIC-III	Title: Summarize the following radiology report. Findings: {{article}}
SciTLDR	Impression: Write a TLDR by summarizing the following scientific paper in one sentence based on its Key Sections (Abstract, Introduction, and Conclusion). Key Sections: {{article}}
	TLDR:

Table 6: Prompt Design

## B Implementation Details

### B.1 LoRA Training Parameters

Table 5 presents the parameters used for LoRA module training.

### B.2 Computation Environment

Experiments were conducted on NVIDIA RTX A6000 GPUs with 48GB of memory. We used 2 GPUs for training LoRA modules and merging them under the proposed method, while 1 GPU was allocated for training baseline methods such as LoRAHub and for inference.

### B.3 Prompt Design

Table 6 presents the prompt design used in both LoRA training and output generation.

## C Pruning Strategies

As the proposed method, we used the importance evaluation metric based on magnitudes of parameter weights and inputs. In the ablation study, we compared it to another metric that considers the magnitudes of parameter weights and gradients.

This metric is defined as follows:

$$I = |W_{ij} \cdot \Delta W_{ij}|$$

where  $\Delta W_{ij}$  represents the gradient of weight  $W_{ij}$ . This formulation estimates the impact of pruning  $W_{ij}$  by approximating the change in loss when setting  $W_{ij}$  to zero (Molchanov et al., 2019; Liang et al., 2021).

To address the variance caused by batch sampling, we apply an uncertainty-aware smoothing technique (Zhang et al., 2022, 2023). The importance at step  $t$ , denoted as  $I^{(t)}$ , is smoothed using an exponential moving average to obtain  $\bar{I}^{(t)}$ . Additionally, the uncertainty measure  $\bar{U}^{(t)}$  quantifies the local fluctuations of  $I^{(t)}$ . The final importance score  $S^{(t)}$  is computed as the product of these two terms:

$$\begin{aligned}\bar{I}^{(t)} &= \beta_1 \bar{I}^{(t-1)} + (1 - \beta_1) I^{(t)} \\ \bar{U}^{(t)} &= \beta_2 \bar{U}^{(t-1)} + (1 - \beta_2) |I^{(t)} - \bar{I}^{(t)}| \\ S^{(t)} &= \bar{I}^{(t)} \cdot \bar{U}^{(t)}\end{aligned}$$

## D Additional Results

Table 7 shows BERTScore results. Figure 7 shows the impact of the pruning thresholds on



	MIMIC-III	SciTLDR	Bloomberg	NLP Paper	Medical Paper
Zero-shot	0.693	0.739	0.605	0.627	0.637
LoRA (XS)	0.729	0.601	0.692	0.754	0.776
LoRA (WL)	0.698	0.756	0.717	0.797	0.812
LoRA (TGT)	0.763	0.778	0.710	0.817	0.843
LoRAHub(XS+WL)	0.717	0.745	0.719	0.798	0.809
LoRAHub (XS+WL+TGT)	0.763	0.780	0.726	0.824	0.827
Ours Merge (XS+WL)	0.768	0.782	0.750	0.824	0.840
Ours Merge (XS+WL+TGT)	0.769	0.780	0.749	0.820	0.843
Ours Merge+Del (XS+WL)	0.766	0.783	0.752	0.838	0.840
Ours Merge+Del (XS+WL+TGT)	0.766	0.783	0.757	0.825	0.857

Table 7: BERTScore results on five summarization tasks of various domains and multiple languages.

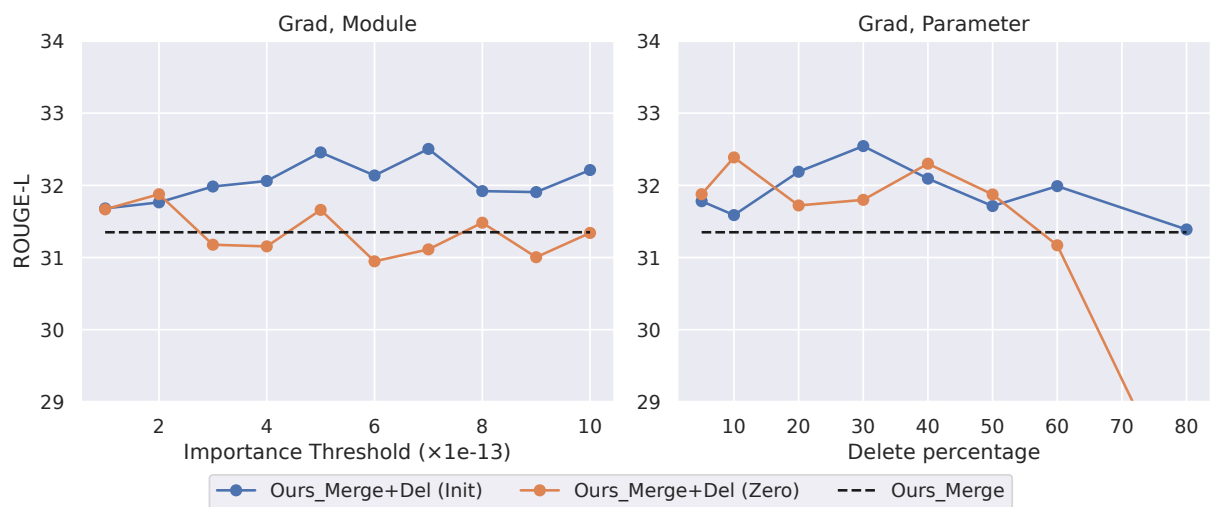


Figure 7: Impact of pruning hyperparameters on model performance (Bloomberg, Ours Merge+Del (XS+WL+TGT), Grad).

Ours Merge+Del (XS+WL+TGT) with Grad and Module or Parameter level pruning configurations.