Low-Rank Adapters Meet Neural Architecture Search for LLM Compression

J. Pablo Muñoz[1], Jinjie Yuan[2], Nilesh Jain[1]

[1] Intel Labs
[2] Intel Corporation
{pablo.munoz, jinjie.yuan, nilesh.jain}@intel.com

Abstract

The rapid expansion of Large Language Models (LLMs) has posed significant challenges regarding the computational resources required for fine-tuning and deployment. Recent advancements in low-rank adapters have demonstrated their efficacy in parameter-efficient fine-tuning (PEFT) of these models. This retrospective paper comprehensively discusses innovative approaches that synergize low-rank representations with Neural Architecture Search (NAS) techniques, particularly weight-sharing super-networks. Robust solutions for compressing and fine-tuning large pre-trained models are developed by integrating these methodologies. Our analysis highlights the potential of these combined strategies to democratize the use of LLMs, making them more accessible for deployment in resource-constrained environments. The resulting models exhibit reduced memory footprints and faster inference times, paving the way for more practical and scalable applications of LLMs. Models and code are available at https://github.com/IntelLabs/Hardware-Aware-Automated-Machine-Learning.

Introduction and Preliminaries

Structured low-rank representations (Kolda and Bader 2009) have played a significant role in the latest successes in Artificial Inteligence (AI). For example, low-rank adaptation (LoRA) (Hu et al. 2022) is a preferred method for parameter-efficient fine-tuning (PEFT) of large foundation models (Bommasani et al. 2021). LoRA expands a linear layer by attaching low-rank adapters, $\boldsymbol{L_1}$ and $\boldsymbol{L_2}$, as demonstrated in the following equations:

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{W}, \tag{1}$$

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{W} + s\boldsymbol{X}\boldsymbol{L}_1\boldsymbol{L}_2, \tag{2}$$

where $\boldsymbol{X}$ is the input to the layer, and $\boldsymbol{W}$ are the layer's weights. s is a scaling factor. $\boldsymbol{W}$ remains frozen, and only the adapters' weights are adapted during fine-tuning, which is significantly more efficient than performing full fine-tuning. Often, the number of parameters in the adapters is a minimal fraction of the total number of parameters in the base model.

Copyright © 2025, AAAI 25 Workshop on Connecting Low-rank Representations in AI (https://april-tools.github.io/colorai/). All rights reserved.

Neural Architecture Search (NAS) techniques attempt to identify a high-performing architectural configuration from a search space of candidate architectures (White et al. 2023). NAS techniques evolved rapidly with the advent of *deep learning*. However, many NAS techniques have become obsolete with the increasing size of large AI models because it is too resource-demanding to evaluate many possible architectures when models have billions of parameters. A particular efficient NAS technique relevant to this paper uses weight-sharing super-networks generated by activating sub-structures of the original neural network (Cai et al. 2020).

We claim that the benefits of cross-pollination between low-rank representations and weight-sharing neural architecture search techniques are bi-directional:

- **NAS techniques enhance low-rank adapters, and,**
- **NAS becomes more efficient by incorporating the guidance of low-rank representations.**

In the following sections, we discuss several solutions that realize these benefits and suggest additional enhancements to be explored in the future.

Elastic LoRA Adapters and Their Applications

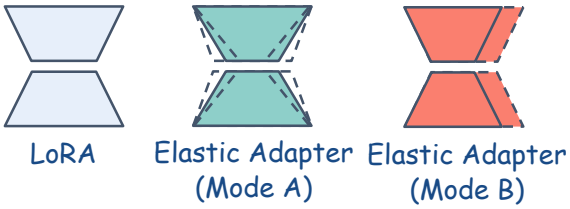

Figure 1: Vanilla LoRA Adapter and two different modes of the elastic adapter. Mode A allows only the LoRA rank to be elastic, while Mode B also enables the input or output channels to be elastic.

In this section, we first introduce the Elastic LoRA Adapter, highlighting its capability to dynamically adjust adapter configurations. This adaptability, coupled with an extensive sub-adapter search space, facilitates its application across various scenarios, enhancing model compression and fine-tuning efficiency and effectiveness. Next, let

us delve step by step into the story of the combination of LoRA adapters and NAS techniques.

Elastic Adapter In weight-sharing NAS, an *elastic* layer, as opposed to a traditional *static* layer, has variable values for its properties. For instance, the weights, $\boldsymbol{W} \in \mathbb{R}^{m \times n}$, of a linear layer might be masked or sliced to activate a smaller structure, $\boldsymbol{W} \in \mathbb{R}^{m \times k}$ where $k < n$. By allowing the activation of variable configurations of a layer during the forward and backward passes, one is effectively training a super-network in which the smaller structures share their weights with their bigger counterparts. Recent advancements in NAS weight-sharing techniques have been utilized in conjunction with low-rank representations. As illustrated in Figure 1, the Elastic LoRA Adapter primarily operates in two modes:

i) Mode A: In the LoRA adapter, matrices $\boldsymbol{L}_1 \in \mathbb{R}^{m \times r}$ and $\boldsymbol{L}_2 \in \mathbb{R}^{r \times n}$ can be rendered elastic by adopting smaller rank values. Specifically, $\boldsymbol{L}_{\delta 1} \in \mathbb{R}^{m \times \{r_0, r_1, \ldots, r\}}$ and $\boldsymbol{L}_{\delta 2} \in \mathbb{R}^{\{r_0, r_1, \ldots, r\} \times n}$, where $r_i \leq r$, $r \ll m$ and $r \ll n$ (Muñoz, Yuan, and Jain 2024).

ii) Mode B: Alternatively, $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$ can achieve elasticity by allowing the activation of substructures with reduced channel widths. This is represented as $\boldsymbol{L}_{\delta 1} \in \mathbb{R}^{\{m_0, m_1, \ldots, m\} \times \{r_0, r_1, \ldots, r\}}$ and $\boldsymbol{L}_{\delta 2} \in \mathbb{R}^{\{r_0, r_1, \ldots, r\} \times \{n_0, n_1, \ldots, n\}}$, where $m_i \leq m$ and $n_i \leq n$ (Muñoz et al. 2024).

To this end, we will describe several methodologies in which low-rank structures and weight-sharing super-networks techniques can benefit each other.

Efficient Neural Architecture Search with the Guidance of Low-Rank Adapters

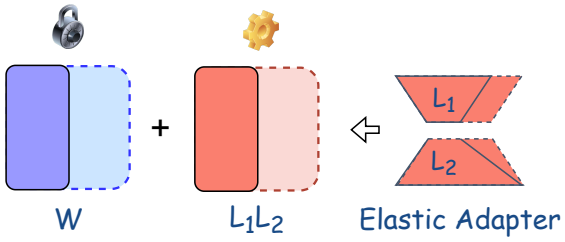

Figure 2: Elastic adapters guide the removal of elements in the frozen model weights, resulting in smaller, high-performing models. This process exemplifies the application of Mode B as depicted in Figure 1.

LoNAS During fine-tuning, the sub-adapters activated can be used to guide the activation of substructures in the base model, as illustrated in Figure 2. This approach corresponds to Mode B in Figure 1, which is characterized by its ability to reduce the overall number of parameters in the model compared to Mode A. In this scenario, the adapters are elastic, and the frozen weights of the base model, denoted as $\boldsymbol{W} \in \mathbb{R}^{m \times n}$, are transformed into $\boldsymbol{W}_\delta \in \mathbb{R}^{m \times \{n_0, n_1, \ldots, n\}}$ or $\boldsymbol{W} \in \mathbb{R}^{m \times n}$ into $\boldsymbol{W}_\delta \in \mathbb{R}^{\{m_0, m_1, \ldots, m\} \times n}$.The search space of possible low-rank adapter configurations is generated by allowing several configurations in the width and rank of the adapters. This alignment, as proposed by LoNAS (Muñoz et al. 2024), results in models with fewer parameters than the base model while maintaining a minimal drop in accuracy and achieving immediate improvements in inference speedup. LoNAS is analogous to traditional NAS, but with the key distinction that only the adapter parameters are trained. The expectation is that this training will guide and adapt the model's search and pruning processes. Due to the high cost of searching LLM with adapters, LoNAS also proposes heuristic sub-networks (i.e., middle point of the search space) to quickly evaluate the quality of the trained super-network. Users can then decide whether further search is necessary based on specific needs. This heuristic strategy has also been applied in subsequent works such as Shears (Muñoz, Yuan, and Jain 2024) and SQFT (Muñoz, Yuan, and Jain 2024), which are discussed later.

Empirical results demonstrate that this approach yields an inference speedup of up to 1.4x and can reduce the model parameters by approximately 80% compared to the original model. For more detailed information, refer to Table 1 and the subsequent section.

LoNAS enhancements have recently been proposed (Sukthanker et al. 2024) by applying elastic LoRA adapters to all the weight matrices of the transformer and allowing the removal of entire transformer blocks. Initially, LoNAS focused solely on the Self-Attention and MLP layers. While LoNAS and its extensions have proven effective, they still face challenges due to the size of state-of-the-art pre-trained models, which have driven the development of more efficient solutions, which we will discuss in the following sections.

Restricting the Elasticity to the Adapter Rank and Exploiting Model Sparsity and Low Numerical Precision

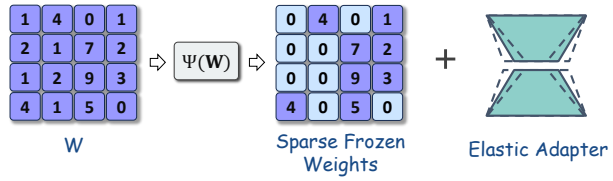

Figure 3: Elastic low-Rank adapters for fine-tuning sparse efficient models. This style exemplifies the application of Mode A as depicted in Figure 1.

Shears Building on LoNAS, Shears (Muñoz, Yuan, and Jain 2024) proposes several modifications to enhance the efficiency of the fine-tuning stage. This approach constrains the application of elasticity exclusively to the low-rank structures, leaving the significantly demanding weights of the base model intact. This strategy is termed **Neural Low-Rank Adapter Search (NLS)**. Additionally, as illustrated in Figure 3, the base model can be sparsified using an arbitrary metric, Ψ, to determine the importance of the pre-trained weights. A popular weight importance metric is Wanda (Sun et al. 2023), where a few feature input activa-

tions, $\boldsymbol{X}$, are used to assess the importance of the weights, i.e., $\Psi(\boldsymbol{W}) = |\boldsymbol{W}| \cdot \|\boldsymbol{X}\|_2$. This score, combined with a desired sparsity level, s, is used to obtain the sparse weights, $\boldsymbol{W}^p$, with a sparsity pattern $S\{\boldsymbol{W}^p\} = \{(i,j) \mid \boldsymbol{W}^p_{i,j} \neq 0, 1 \leq i \leq m, 1 \leq j \leq n\}$, such that $|S\{\boldsymbol{W}^p\}| \leq |S\{\boldsymbol{W}\}|$.

Overall, Shears introduces the concept of Mode A elastic adapters, which allow the rank values to be flexible, thereby enabling the exploration of more potential sub-adapters. This approach has been demonstrated to outperform traditional LoRA (with fixed rank values) and alleviates the challenge of setting the hyperparameter rank value when using LoRA. Most importantly, Shears found that the NLS algorithm is particularly well-suited for sparse models. When pre-trained weights are sparsified, there is a natural and significant drop in accuracy compared to dense models. On this basis, using NLS for fine-tuning can maximally recover or adapt the model's performance to a specific downstream task.

SQFT Shears is extended by SQFT (Muñoz, Yuan, and Jain 2024) to manipulate sparse models on low numerical precision. SQFT is inspired by QLoRA (Dettmers et al. 2023), which was proposed to improve fine-tuning efficiency when using low-rank adapters. SQFT enables three different pipelines that account for the varying characteristics of the base models, such as whether they possess sparsity or have been quantized to low numerical precision.

Empirical results demonstrate that by combining elastic LoRA adapters into the sparse or quantized base model, Shears, and SQFT enable effective fine-tuning of compressed models to adapt to specific downstream tasks. This approach produces compressed models that either improve or exhibit only minor drops in accuracy. The enhanced sparsity and precision can lead to significant speedups when utilizing runtimes optimized for these patterns.

However, a significant challenge in Shears and SQFT when dealing with compressed models and dense adapters is the potential limitations encountered when attempting to merge the low-rank adapters with the based model after fine-tuning. For instance, if the model is sparse but the adapters are dense, the sparsity in the model will be lost when merging. A similar limitation arises when the based model has a different numerical precision than the low-rank adapters used for fine-tuning. In the next section, we describe how these limitations are addressed to ensure the integrity and performance of the fine-tuned models.

Addressing the Challenges of Merging Adapters with Low-precision Sparse Models

Within SQFT, two strategies, *SparsePEFT* and *QA-SparsePEFT*, are proposed to address the limitations described in the previous section. These limitations arise when attempting to merge the low-rank adapters with base models with differing sparsity patterns or numerical precision. The following sections provide a detailed discussion of these strategies.

SparsePEFT This strategy ensures that the sparsity in low-rank adapters is aligned with their corresponding base model's weights during fine-tuning. SQFT achieves this by generating a binary mask $\boldsymbol{M}$ for each weight matrix $\boldsymbol{W}$ in the base model. The mask $\boldsymbol{M}$ is $\forall i \forall j (W_{i,j} \neq 0 \Rightarrow \boldsymbol{M}_{i,j} = 1)$. Utilizing $\boldsymbol{M}$, SQFT sparsifies the adapters' matrix $(\boldsymbol{L}_1\boldsymbol{L}_2)$ to obtain $\boldsymbol{L}^p$, i.e., $\boldsymbol{L}^p = (\boldsymbol{L}_1\boldsymbol{L}_2) \odot \boldsymbol{M}$. This approach ensures sparsity awareness during fine-tuning, allowing the merging of the base model's weights and adapter's weights without losing the sparsity induced before fine-tuning.

QA-SparsePEFT This strategy is employed by SQFT when the model has been quantized to lower numerical precision, and low-rank adapters with higher numerical precision are applied for fine-tuning. Quantization-aware SparsePEFT (QA-SparsePEFT) leverages the frozen zeros $\boldsymbol{z}$ and scales $\boldsymbol{s}$ resulting from the pre-fine-tuning stage in which each weight matrix, $\boldsymbol{W}$, was asymmetrically quantized. By utilizing $\boldsymbol{z}$, $\boldsymbol{s}$, the numerical target range $[0, 2^{n-1}]$ for quantization (where n represents the target bit-width), and the pre-quantized sparse weights, $\boldsymbol{W}^p$, QA-SparsePEFT achieves quantization-aware fine-tuning with low-rank adapters on $\widehat{\boldsymbol{W}}^p_m$, i.e., the sparse quantized (merged) weights. This process is formalized as,

$$\widehat{\boldsymbol{W}}^p_m = \text{clamp}\left(\text{round}\left(\frac{\boldsymbol{W^p} + \boldsymbol{L^p}}{\boldsymbol{s}}\right) + \boldsymbol{z}, 0, 2^n - 1\right), \quad (3)$$

To obtain the dequantized weights,$\tilde{\boldsymbol{W}}^p_m$, the inverse process is followed,

$$\tilde{\boldsymbol{W}}^p_m = \boldsymbol{s}\left(\widehat{\boldsymbol{W}}^p_m - \boldsymbol{z}\right). \quad (4)$$

In summary, the integration of Elastic LoRA Adapters with NAS techniques offers a promising approach to model compression and fine-tuning. By leveraging the flexibility of elastic adapters and the efficiency of NAS, methods like LoNAS, Shears, and SQFT demonstrate significant improvements in parameter reduction and inference speedup without sacrificing accuracy. Additionally, strategies such as SparsePEFT and QA-SparsePEFT address the challenges of merging (elastic or static) LoRA adapters with low-precision sparse models, ensuring robust performance and maintaining model integrity. These advancements highlight the potential of combining low-rank adapters with NAS to optimize large language models.

Performance Summary and Additional Considerations

We summarize the performance, accuracy, and compression efficiency of LoNAS (Table 1), SQFT (Table 2) and Shears (Tables 2, and 3) from their respective papers. The reader can find additional details and an exhaustive list of experiments in each solutions' source. From these tables, we can observe that LoNAS can obtain competitive results compared to vanilla LoRA. However, LoNAS application is costlier than the other two discussed solutions due to the elasticity enabled in the model's weights, in addition to the inserted elastic adapters, which makes the fine-tuning stage more expensive. Shears and SQFT, on the other hand, are more

Table 1: The performance of LoNAS using elastic adapters mode B, including accuracy score and model compression efficiency when fine-tuning LLaMA-7B on 15k unified commonsense reasoning dataset from LLM-Adapters (Hu et al. 2023). The average score represents the results across eight commonsense tasks. These results are reproduced from Muñoz et al. (2024)

Method	Average Score	Relative Score	Total Params.	TFLOPs	Inference Speedup
LoRA	65.8	100.0%	6.7B	1.7	1.00×
LoNAS (Heuristic Subnet)	65.2	99.1%	5.6B	1.4	1.23×
LoNAS (Search Subnet-1)	67.1	102.0%	5.6B	1.4	1.28×
LoNAS (Search Subnet-2)	65.6	99.7%	5.1B	1.3	1.41×

Table 2: The performance of Shears and SQFT from Muñoz et al. (2024), when fine-tuning Mistral-7B-v0.3 on GSM8K using elastic adapters mode A.

Method	Accuracy	Relative Acc.	Sparsity	Precision (Base + Adapter / Base)
w/o tune	36.0	-	-	FP16
LoRA	44.1	100.0%	50%	FP16
Shears	45.1	102.3%	50%	FP16 + FP16
SQFT + SparsePEFT	50.1	113.6%	50%	FP16
SQFT	44.5	100.9%	50%	INT4 + FP16
SQFT + QA-SparsePEFT	44.0	99.8%	50%	INT4

fine-tuning efficient since their manipulation is only at the adapters' level.

Table 3: The performance of Shears for LLaMA-13B on a 10k unified math reasoning dataset from LLM-Adapters (Hu et al. 2023) using elastic adapters mode A. These results are reproduced from Muñoz et al. (2024), and the average score represents the results across four math tasks (GSM8K (Cobbe et al. 2021), AQUA (Ling et al. 2017), MAWPS (Lan et al. 2022) and SVAMP (Patel, Bhattamishra, and Goyal 2021)).

Method	Average Score	Relative Score	Sparsity	Non-zero Params.
LoRA	51.1	100.0%	-	13.0B
Shears	52.0	101.8%	40%	8.0B
Shears	50.9	99.6%	50%	6.7B

The larger research community can further improve the solutions discussed here. For instance, the additional stage to discover a high-performing adapter configuration from the search space of possible configurations presents several opportunities for improvement. As illustrated in Figure 4, evolutionary algorithms, e.g., the Non-Dominated Sorting Genetic Algorithm II (NSGA-II) (Deb et al. 2002), might be used to discover Pareto-optimal elastic low-rank adapter configurations. In this example, a multi-objective search is performed on multiply-accumulate (MAC) operations and validation accuracy. This search can be expensive, presenting opportunities for more efficient alternatives.

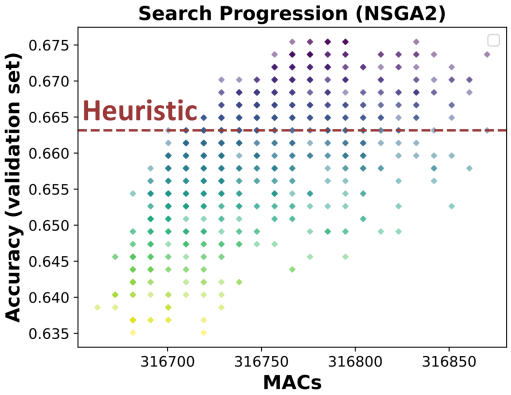

Figure 4: Search progression to discover Pareto-optimal low-rank adapter configurations. The horizontal line represents the zero-shot accuracy of the midpoint heuristic sub-adapter.

Conclusion

This retrospective paper discusses recent work on low-rank representations and the synergy with neural architecture search (NAS). The results from the papers that propose solutions aligned with this synergy confirm the benefits in both directions: (i) Low-rank adapters are enhanced by NAS techniques, i.e., elastic low-rank adapters achieve better results than their vanilla low-rank adapter counterparts, and (ii) NAS becomes more efficient by incorporating the utilization of low-rank representations. This synergy also motivates future work to better understand the interaction between these two domains and propose more sophisticated solutions that expand on existing work.

References

Bommasani, R.; Hudson, D. A.; Adeli, E.; Altman, R.; Arora, S.; von Arx, S.; Bernstein, M. S.; Bohg, J.; Bosselut, A.; Brunskill, E.; Brynjolfsson, E.; Buch, S.; Card, D.; Castellon, R.; Chatterji, N. S.; Chen, A. S.; Creel, K. A.; Davis, J.; Demszky, D.; Donahue, C.; Doumbouya, M.; Durmus, E.; Ermon, S.; Etchemendy, J.; Ethayarajh, K.; Fei-Fei, L.; Finn, C.; Gale, T.; Gillespie, L. E.; Goel, K.; Goodman, N. D.; Grossman, S.; Guha, N.; Hashimoto, T.; Henderson, P.; Hewitt, J.; Ho, D. E.; Hong, J.; Hsu, K.; Huang, J.; Icard, T. F.; Jain, S.; Jurafsky, D.; Kalluri, P.; Karamcheti, S.; Keeling, G.; Khani, F.; Khattab, O.; Koh, P. W.; Krass, M. S.; Krishna, R.; Kuditipudi, R.; Kumar, A.; Ladhak, F.; Lee, M.; Lee, T.; Leskovec, J.; Levent, I.; Li, X. L.; Li, X.; Ma, T.; Malik, A.; Manning, C. D.; Mirchandani, S. P.; Mitchell, E.; Munyikwa, Z.; Nair, S.; Narayan, A.; Narayanan, D.; Newman, B.; Nie, A.; Niebles, J. C.; Nilforoshan, H.; Nyarko, J. F.; Ogut, G.; Orr, L.; Papadimitriou, I.; Park, J. S.; Piech, C.; Portelance, E.; Potts, C.; Raghunathan, A.; Reich, R.; Ren, H.; Rong, F.; Roohani, Y. H.; Ruiz, C.; Ryan, J.; R'e, C.; Sadigh, D.; Sagawa, S.; Santhanam, K.; Shih, A.; Srinivasan, K. P.; Tamkin, A.; Taori, R.; Thomas, A. W.; Tramèr, F.; Wang, R. E.; Wang, W.; Wu, B.; Wu, J.; Wu, Y.; Xie, S. M.; Yasunaga, M.; You, J.; Zaharia, M. A.; Zhang, M.; Zhang, T.; Zhang, X.; Zhang, Y.; Zheng, L.; Zhou, K.; and Liang, P. 2021. On the Opportunities and Risks of Foundation Models. *ArXiv*.

Cai, H.; Gan, C.; Wang, T.; Zhang, Z.; and Han, S. 2020. Once for All: Train One Network and Specialize it for Efficient Deployment. In *International Conference on Learning Representations*.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; Hesse, C.; and Schulman, J. 2021. Training Verifiers to Solve Math Word Problems. arXiv:2110.14168.

Deb, K.; Pratap, A.; Agarwal, S.; and Meyarivan, T. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2): 182–197.

Dettmers, T.; Pagnoni, A.; Holtzman, A.; and Zettlemoyer, L. 2023. QLoRA: Efficient Finetuning of Quantized LLMs. *arXiv preprint arXiv:2305.14314*.

Hu, E. J.; yelong shen; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*.

Hu, Z.; Lan, Y.; Wang, L.; Xu, W.; Lim, E.-P.; Lee, R. K.-W.; Bing, L.; and Poria, S. 2023. LLM-Adapters: An Adapter Family for Parameter-Efficient Fine-Tuning of Large Language Models. *arXiv preprint arXiv:2304.01933*.

Kolda, T. G.; and Bader, B. W. 2009. Tensor Decompositions and Applications. *SIAM Review*, 51(3): 455–500.

Lan, Y.; Wang, L.; Zhang, Q.; Lan, Y.; Dai, B. T.; Wang, Y.; Zhang, D.; and Lim, E.-P. 2022. Mwptoolkit: an open-source framework for deep learning-based math word problem solvers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 13188–13190.

Ling, W.; Yogatama, D.; Dyer, C.; and Blunsom, P. 2017. Program Induction by Rationale Generation: Learning to Solve and Explain Algebraic Word Problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 158–167. Vancouver, Canada: Association for Computational Linguistics.

Muñoz, J. P.; Yuan, J.; and Jain, N. 2024. Shears: Unstructured Sparsity with Neural Low-rank Adapter Search. In Yang, Y.; Davani, A.; Sil, A.; and Kumar, A., eds., *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)*, 395–405. Mexico City, Mexico: Association for Computational Linguistics.

Muñoz, J. P.; Yuan, J.; Zheng, Y.; and Jain, N. 2024. LoNAS: Elastic Low-Rank Adapters for Efficient Large Language Models. In Calzolari, N.; Kan, M.-Y.; Hoste, V.; Lenci, A.; Sakti, S.; and Xue, N., eds., *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, 10760–10776. Torino, Italia: ELRA and ICCL.

Muñoz, J. P.; Yuan, J.; and Jain, N. 2024. SQFT: Low-cost Model Adaptation in Low-precision Sparse Foundation Models. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Findings of the Association for Computational Linguistics: EMNLP 2024*, 12817–12832. Miami, Florida, USA: Association for Computational Linguistics.

Patel, A.; Bhattamishra, S.; and Goyal, N. 2021. Are NLP Models really able to Solve Simple Math Word Problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2080–2094. Online: Association for Computational Linguistics.

Sukthanker, R. S.; Staffler, B.; Hutter, F.; and Klein, A. 2024. Large Language Model Compression with Neural Architecture Search. In *Workshop on Machine Learning and Compression, NeurIPS 2024*.

Sun, M.; Liu, Z.; Bair, A.; and Kolter, J. Z. 2023. A Simple and Effective Pruning Approach for Large Language Models. *arXiv preprint arXiv:2306.11695*.

White, C.; Safari, M.; Sukthanker, R.; Ru, B.; Elsken, T.; Zela, A.; Dey, D.; and Hutter, F. 2023. Neural Architecture Search: Insights from 1000 Papers.