LoRA Parameter Generation: What Works and What's Next

Anonymous ACL submission

Abstract

LoRA generation, which produces a LoRA adapter directly from a natural language description without per-task training, has recently emerged as a promising path toward adaptation of foundation models. Yet despite a surge of recent methods reporting encouraging results in narrow settings, their reliability across diverse adaptation needs remains poorly understood, leaving a critical blind spot precisely as the community begins to adopt these techniques at scale. In this work, we present a timely analysis of current methods and outline what we see as the key directions forward. Furthermore, we identify using generated LoRA as the initialization for task-specific training as one promising direction, which delivers consistent gains across multiple tasks. Together, our results suggest that LoRA generation is a practical mechanism for fast adaptation, but still requires narrowing the performance gap, stronger condition-following, and better reasoning to approach and outperform task-specific training, establishing LoRA generation as a scalable path toward dynamic adaptation on complex tasks. Our code will be released.

1 Introduction

Large language models (LLMs) are increasingly expected to adapt to new tasks, domains, and user needs. Supervised fine-tuning remains a standard way to obtain such adaptation (Raffel et al., 2020; Ouyang et al., 2022), but it requires task-specific data, gradient-based optimization, and often substantial engineering effort for each new setting. Parameter-efficient fine-tuning methods, such as low-rank adaptation (LoRA) (Hu et al., 2022), reduce this cost by updating only a small set of adapter parameters while keeping the base model fixed. However, they do not remove the underlying per-task optimization loop: each new task still typically requires collecting examples, training an adapter, selecting hyperparameters, and storing the resulting weights. As LLMs are adapted to more tasks and domains, training a new adapter for each case is still costly and slow.

A more recent direction seeks to amortize this adaptation process through LoRA generation. Instead of optimizing a new adapter from scratch, a generator predicts LoRA weights conditioned on task descriptions, data samples, or input contexts (Liang et al., 2026; Charakorn et al., 2025; Liu et al., 2026). If effective, this approach could turn LLM adaptation from repeated gradient-based training into a fast conditional generation problem, enabling cheaper specialization, rapid deployment, and potentially dynamic adaptation at inference time. This makes a systematic analysis of LoRA generation both timely and necessary. We analyze recent LoRA generation methods around one central question: *where does current LoRA generation already work, and what should come next?*

To answer this question, we design an analysis covering both task-level and context-level LoRA generation. Task-level LoRA generation produces adapters from a description of the task, such as a task prompt or data samples. Context-level LoRA generation produces adapters that internalize a given context into LoRA weights, so the model can answer questions about the context without seeing it in the prompt. We first test what current LoRA generation can already achieve by comparing generated adapters with appropriate baselines, including SFT, base models, and In-Context inference. We then analyze the remaining gaps from three angles: (1) how far generated adapters remain from strong baselines, (2) whether generated adapters follow fine-grained task or context conditions, and (3) whether generated LoRA can reliably support the reasoning needed for complex tasks. Combining the advantages of both LoRA generation and SFT, we propose using generated LoRA as the initialization for subsequent SFT, with consistent gains on multiple tasks.

Across task-level and context-level settings, current LoRA generation provides fast adaptation and sometimes approaches strong baselines, but is not yet reliable for all adaptation needs. The gap becomes clear when LoRA generation needs to follow the conditioning input more precisely, reach stronger adaptation performance, or support reasoning for complex tasks.

Our contributions are fourfold. First, we organize recent LoRA generation into task-level and context-level settings and analyze both with appropriate baselines. Second, we show that current LoRA generation can already support fast adaptation, sometimes approaching strong baselines. Third, we identify three concrete progress points: narrowing the performance gap, strengthening semantic discrimination, and improving reasoning capability. Fourth, we propose using generated LoRA as the initialization for task-specific training, with consistent gains across multiple tasks. The setup, metrics, and additional results are provided in Appendix A.1, A.2, A.3, respectively.

2 Related Work

2.1 Parameter-Efficient Fine-Tuning

Parameter-efficient fine-tuning (PEFT) (Li and Liang, 2021; Lester et al., 2021; Houlsby et al., 2019) updates only a small set of adapter parameters instead of the full model. LoRA (Hu et al., 2022) is a representative PEFT method that stores adaptation in low-rank weights, effectively reducing training and memory cost. Many variants (Dettmers et al., 2023; Zhang et al., 2023) build on LoRA to further improve efficiency or downstream quality. Although these methods make adaptation cheaper, they still require collecting task-specific data and running a full optimization cycle for each new task. This per-task cost has motivated recent work to explore generating LoRA weights directly without training.

2.2 LoRA Parameter Generation

LoRA parameter generation produces LoRA weights directly from a condition input, without training a new adapter on a task-specific dataset. Recent work falls into two main settings. Task-level LoRA generation generates adapters from a task description, task prompts, or data samples, including T2L (Charakorn et al., 2025), DnD (Liang et al., 2026), ORAL (Shahroz et al., 2025), and ICM-LoRA (Shao et al., 2025). Context-level LoRA generation generates adapters from input context or documents, so the model can answer questions using information stored in the LoRA weights, including Generative Adapter (Chen et al., 2025), SHINE (Liu et al., 2026), and Doc-to-LoRA (Charakorn et al., 2026). These methods use different designs, but they share one goal: turn a natural language input into an LLM adapter.

3 What Works: LoRA Generation Enables Fast Adaptation

We first ask whether LoRA generation can serve as a practical mechanism for fast LLM adaptation. Existing methods suggest the answer is yes: a trained generator produces a task- or context-conditioned adapter in a single forward pass, with downstream performance that can approach strong baselines on some tasks. The key shift is that adaptation cost is paid once during generator training instead of repeatedly for every new task or context. This amortization is what makes LoRA generation more than just a parameter-efficient method.

	Method	Single-hop	Multi-hop	SQuAD-N
M-7B	In-Context	**62.24**	**18.75**	**55.86**
	SFT	18.98	13.72	16.35
	Gen Adapter	55.56	15.88	40.57
Q3-8B	In-Context	**89.43**	**32.44**	**81.74**
	SFT	24.95	10.20	27.32
	SHINE	60.32	26.69	39.73

Table 1: **Context-level LoRA generation on QA benchmarks.** We report average answer F1 for context-level adaptation. *Generated adapters fall short of In-Context, but successfully encode context as a LoRA and outperform SFT.*

Table 1 evaluates context-level LoRA generation against two reference points: per-context SFT, which fine-tunes a separate LoRA for each context, and In-Context inference, which keeps the original context in the prompt and serves as a strong upper bound. Across both Mistral-7B (Jiang et al., 2023) and Qwen3-8B (Yang et al., 2025a), LoRA generation substantially outperforms per-context SFT on every QA group. Generative Adapter raises single-hop F1 from 18.98% to 55.56% and SQuAD-N from 16.35% to 40.57% on Mistral-7B, while SHINE improves single-hop F1 from 24.95% to 60.32% and multi-hop F1 from 10.20% to 26.69% on Qwen3-8B. LoRA generation trails In-Context inference but consistently outperforms per-context SFT across all QA settings, providing a practical alternative to keeping the context in the prompt.

The efficiency results in Table 2 make this fast-adaptation claim concrete. Per-task SFT takes nearly 10^6 ms to fit a single dataset cycle, while T2L and DnD produce an adapter in 16 ms and 11

ms. The context-level setting shows the same gap, where per-context SFT takes over 10^4 ms per context, while Generative Adapter and SHINE produce an adapter in 55 ms and 127 ms. LoRA generation thus turns adaptation from a repeated optimization loop into a single inference call.

	Method	Peak Mem (GB)	FLOPs ($\times 10^{12}$)	Latency (ms)
T	SFT (Task)	39.61	1,132,276.66	969,815.30
	T2L	2.77	0.11	16.49
	DnD	2.21	0.84	11.16
C	SFT (Context)	30.46	2,522.90	10455.90
	SHINE	22.88	112.48	126.87
	Gen Adapter	16.30	15.93	55.36

Table 2: **Efficiency of LoRA generation for fast adaptation.** We report peak memory, FLOPs, and adapter-generation latency for task-level (**T**) and context-level (**C**) adaptation. *LoRA generation reduces memory, computation, and latency compared with SFT.*

> **Takeaway.** LoRA generation delivers fast LLM adaptation in a single forward pass, outperforming per-context training.

4 What Comes Next: Beyond Surface-Level LoRA Generation

We now ask what LoRA generation needs beyond fast adaptation. The next step is to move past surface-level generation, where LoRA generation should narrow the performance gap to strong baselines (§4.1), become sensitive to the natural language condition that guides it (§4.2), and internalize reasoning capability into the generated adapter (§4.3). LoRA generation can also serve as the initialization for subsequent task-specific training, providing consistent gains on multiple tasks (§4.4).

Method	ARC-e *Acc. (%)*	BoolQ *Acc. (%)*	GSM8K *Acc. (%)*	MATH *Acc. (%)*	HumanEval *pass@1 (%)*
Base	34.26	47.22	**64.67**	**29.48**	15.61
SFT	**65.70**	**62.17**	31.77	16.32	**17.26**
DnD	40.96	17.32	1.99	3.87	14.59
Δ	-24.74	-44.85	-29.78	-12.45	-2.67

Table 3: **Task-level LoRA generation via DnD across five datasets.** Δ denotes the performance difference between DnD and SFT. *DnD-generated adapters remain behind SFT across all five tasks.*

4.1 Usable Adaptation with a Remaining Gap

For *task-level adaptation*, current LoRA generation remains useful but clearly behind task-specific SFT. T2L falls behind SFT by 32.52% on GSM8K and 25.61% on HumanEval (Table 4). DnD shows even larger gaps, trailing SFT by 44.85% on BoolQ and 29.78% on GSM8K (Table 3), and underperforms SFT on all five evaluated tasks. For *context-level adaptation*, the key gap is no longer against per-context SFT, but against In-Context inference. Generative Adapter is close on Mistral-7B single-hop QA, reaching 55.56% compared with 62.24% for In-Context. The gap widens sharply on longer contexts: on Qwen3-8B SQuAD-N, SHINE reaches 39.73% while In-Context reaches 81.74%.

These results reinforce that LoRA generation needs to preserve dense information from the input condition. This would allow LoRA generation to produce discriminative adapters that match the specific problem, rather than adapters that only provide broadly useful behavior.

> **Takeaway.** LoRA generation needs to preserve input information and turn it into adapters that support the intended adaptation.

4.2 Weak Discrimination Across Conditions

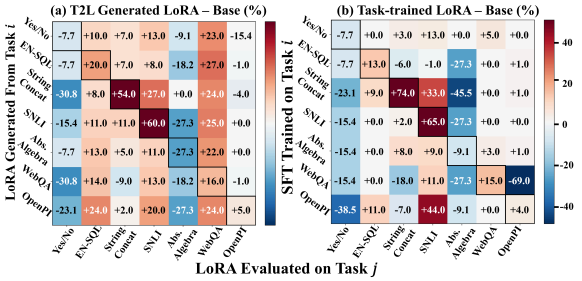

Figure 1: **Discriminative control of task-level LoRA generation via T2L.** Each cell shows the accuracy change over the base model when an adapter from source task i is evaluated on target task j. *T2L shows weaker task control than SFT, with adapters often affecting off-diagonal tasks.*

LoRA generation should produce different adapters for different conditions, yet Figure 1 shows this does not hold for T2L. The string-concat adapter improves its matched target by 54.0%, but also improves SNLI by 27.0% and WebQ QA by 24.0%. The MMLU abstract algebra adapter reduces its matched target by 27.3%, yet still improves text-to-SQL by 13.0%, SNLI by 11.0%, and WebQ QA by 22.0%. The generated adapter does not match the requested task. DnD partially mitigates this by using data samples and a reconstruction objective, instead of the task description and SFT loss used by T2L. Data samples provide a more robust condition signal, which lets the generated adapters discriminate across datasets more reliably. On ARC-c, HellaSwag, PIQA, and WinoGrande, DnD recovers matched-dataset lifts of 13.0%, 13.8%, 25.1%, and 48.1%, and its off-diagonal pattern also closely tracks that of direct SFT (Figure 4).

> **Takeaway.** Data samples provide stronger guidance than underspecified task descriptions for generating task-specific adapters.

Method	ARC-C *Acc. (%)*	PIQA *Acc. (%)*	GSM8K *Acc. (%)*	HumanEval *Pass@1 (%)*	Grammar *ROUGE-L (%)*	Fallacy *ROUGE-L (%)*	Substitute *ROUGE-L (%)*	Style *ROUGE-L (%)*	Narrative *ROUGE-L (%)*
Base	**75.92**	69.40	**57.16**	**34.15**	52.94	47.06	77.26	8.55	**14.69**
SFT	74.15	**83.40**	41.62	27.44	87.94	**76.47**	**90.76**	83.33	7.75
T2L	72.24	71.00	9.10	1.83	**88.82**	58.82	68.67	**84.62**	11.82
Δ	-1.91	-12.40	-32.52	-25.61	+0.88	-17.65	-22.09	+1.29	+4.07

Table 4: **Task-level LoRA generation via T2L across nine datasets.** We compare the base model, task-specific SFT, and LoRA adapters generated by T2L. Δ denotes the performance difference between T2L and SFT. *LoRA generation is competitive on some generation tasks, but remains far behind SFT on reasoning-heavy tasks.*

4.3 Weak Reasoning in Generated Adapters

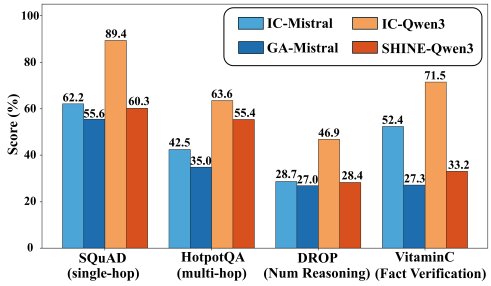

Figure 2: **Reasoning gap in context-level LoRA generation.** *Context compression into LoRA remains weak when answers require reasoning over the context.*

On reasoning-heavy tasks, the generated adapters perform poorly across both task and context levels. T2L reaches 9.10% on GSM8K, while DnD reaches 1.99% on GSM8K and 3.87% on MATH (Tables 4, 3). At the context level, the gap to In-Context inference widens sharply when answers require reasoning over the context rather than direct evidence lookup. Generated adapters reach 28.4% on DROP and 33.2% on claim verification, compared with 46.9% and 71.5% for In-Context inference (Figure 2). LoRA generation can store useful context signals, but loses the evidence and computation needed to reason over them.

> **Takeaway.** Current LoRA generation cannot reliably produce adapters that perform complex reasoning.

4.4 LoRA Generation as Initialization

Beyond direct use, generated LoRA can serve as an initialization for subsequent task-specific training. We start SFT from a T2L-generated LoRA instead of random weights, then continue training on the target task's data. Figure 3 compares the two initialization strategies. T2L initialization yields a more stable training curve with smaller fluctuations across gradient steps, and improves the final performance on 3 of 4 tasks: Fallacy (+17.6%), Macroeconomics (+10.0%), and Style (+3.8%). On Computer Science, the two converge to similar performance. The largest gain on Fallacy is not a special case, but *combines the advantages of LoRA generation and task-specific training on only 136 training examples.* With such limited data, random initialization cannot recover a strong adapter on its own, while T2L initialization provides prior knowledge that the small training set refines.

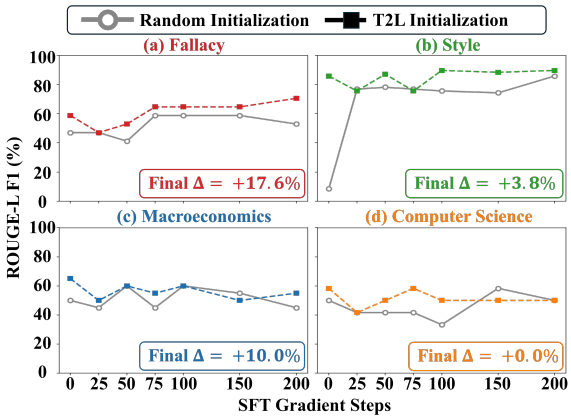

Figure 3: **Generated LoRA as initialization for task-specific training.** *Generated LoRA can serve as a stable initialization for subsequent task-specific training and improve final performance.*

> **Takeaway.** Initializing task-specific training from generated LoRA yields large gains on small training sets.

5 Conclusion

This paper analyzes recent advances in LoRA generation. We show that LoRA generation is emerging as a fast route for LLM adaptation, with early signs that generated adapters can approach SFT on some tasks. More importantly, our analysis identifies where LoRA generation can move next: narrowing the performance gap to strong baselines, improving condition controllability, and strengthening reasoning capability. We further propose initializing task-specific training from generated LoRA, which significantly improves SFT performance when task data is limited. Together, these findings establish LoRA generation as a practical step toward fast LLM adaptation, while pointing to the key advances needed for its scalability. In addition, we believe LoRA generation also holds strong potential for personalized AI assistants, scalable multi agent systems, and potentially opening the next stage of the AI model acquisition paradigm.

6 Limitations

Although we analyze LoRA generation across performance gap, condition control, and reasoning capability, several directions remain worth further analysis. First, a robustness analysis under noisy or poisoned condition inputs remains to be done. All our experiments use clean task descriptions and clean contexts, but noisy or poisoned inputs may affect both generator training and the quality of adapters at inference time. Second, a controlled analysis is needed to test whether current training paradigms encourage shortcut learning in LoRA generation. Standard objectives, either pure adapter reconstruction or end-to-end task loss, may push generators to capture surface patterns rather than the underlying task knowledge, producing adapters that match the output distribution without internalizing task knowledge. Finally, a scaling analysis of generator training data is needed. Studying how generation quality changes as the generator sees more tasks, more domains, or more training tokens would help establish scaling laws for LoRA generation and clarify whether current generalizability limits are inherent or can be closed by data scale.

References

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, and 1 others. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 632–642.

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. *arXiv preprint arXiv:2407.00066*.

Rujikorn Charakorn, Edoardo Cetin, Yujin Tang, and Robert Tjarko Lange. 2025. Text-to-loRA: Instant transformer adaption. In *Forty-second International Conference on Machine Learning*.

Rujikorn Charakorn, Edoardo Cetin, Shinnosuke Uesaka, and Robert Tjarko Lange. 2026. Doc-to-lora: Learning to instantly internalize contexts. *arXiv preprint arXiv:2602.15902*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, and 1 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Tong Chen, Hao Fang, Patrick Xia, Xiaodong Liu, Benjamin Van Durme, Luke Zettlemoyer, Jianfeng Gao, and Hao Cheng. 2025. Generative adapter: Contextualizing language models in parameters with a single forward pass. In *The Thirteenth International Conference on Learning Representations*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. 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 2924–2936.

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, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. 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 2368–2378.

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

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

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

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 *International Conference on Learning Representations*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pages 3045–3059.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597.

Zhiyuan Liang, Dongwen Tang, Yuhao Zhou, Xuanlei Zhao, Mingjia Shi, Wangbo Zhao, Zekai Li, Peihao Wang, Konstantin Schürholt, Damian Borth, Michael M. Bronstein, Yang You, Zhangyang Wang, and Kai Wang. 2026. Drag-and-drop LLMs: Zero-shot prompt-to-weights. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yewei Liu, Xiyuan Wang, Yansheng Mao, Yoav Gelbery, Haggai Maron, and Muhan Zhang. 2026. Shine: A scalable in-context hypernetwork for mapping context to lora in a single pass. *arXiv preprint arXiv:2602.06358*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2381–2391.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Gray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 2383–2392.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Tal Schuster, Adam Fisch, and Regina Barzilay. 2021. Get your vitamin c! robust fact verification with contrastive evidence. In *Proceedings of the 2021 conference of the North American chapter of the Association for Computational Linguistics: Human language technologies*, pages 624–643.

Rana Shahroz, Dongwen Tang, Pingzhi Li, Kai Wang, and Tianlong Chen. 2025. ORAL: Prompting your large-scale LoRAs via conditional recurrent diffusion. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 1357–1370, Suzhou, China. Association for Computational Linguistics.

Yihua Shao, Minxi Yan, Yang Liu, Siyu Chen, Wenjie Chen, Xinwei Long, Ziyang Yan, Lei Li, Chenyu Zhang, Nicu Sebe, Hao Tang, Yan Wang, Hao Zhao, Mengzhu Wang, and Jingcai Guo. 2025. In-context meta lora generation. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pages 6138–6146. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Niket Tandon, Keisuke Sakaguchi, Bhavana Dalvi, Dheeraj Rajagopal, Peter Clark, Michal Guerquin, Kyle Richardson, and Eduard Hovy. 2020. A dataset for tracking entities in open domain procedural text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6408–6417.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, and 1 others. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. musique: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, and 1 others. 2022. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. In *Proceedings of the 2022 conference on empirical methods in natural language processing*, pages 5085–5109.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 23 others. 2025b. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2369–2380.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 4791–4800.

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 *The Eleventh International Conference on Learning Representations*.

A Experimental Setup

A.1 Dataset and Base Model

We evaluate four LoRA generation methods, two for task-level adaptation and two for context-level adaptation.

DnD (Liang et al., 2026). A method that generates LoRA adapters from task data samples through a reconstruction objective. We use the released DnD checkpoints with Qwen2.5-Instruct (Yang et al., 2025b) as the base model. For the cross-task control analysis (Figure 4), we use Qwen2.5-0.5B-Instruct across seven commonsense tasks: ARC-c, ARC-e (Clark et al., 2018), BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), OpenBookQA (Mihaylov et al., 2018), PIQA (Bisk et al., 2020), and WinoGrande (Sakaguchi et al., 2021). For the five-task performance evaluation (Table 3), we use Qwen2.5-0.5B-Instruct for ARC-e and BoolQ, and Qwen2.5-1.5B-Instruct for GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021b), and HumanEval (Chen et al., 2021).

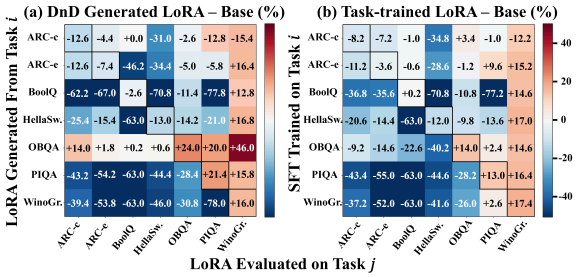

Figure 4: **Cross-dataset discrimination of task-level LoRA generation via DnD.** *DnD adapters align with matched SFT behavior through reconstruction, showing a degree of condition-driven control.*

T2L (Charakorn et al., 2025). A hypernetwork that generates LoRA adapters from natural language task descriptions. We use the SFT-trained variant with Gemma-2-2B-it (Team et al., 2024) as the base model, loaded from the official release. For the nine-task performance evaluation (Table 4), we evaluate on four reasoning and code tasks (ARC-C (Clark et al., 2018), PIQA (Bisk et al., 2020), GSM8K (Cobbe et al., 2021), HumanEval (Chen et al., 2021)) and five generation tasks (Grammar, Fallacy, Substitute, Style, Narrative) drawn from the Super-Natural Instructions (SNI) dataset (Wang et al., 2022; Brüel-Gabrielsson et al., 2024). For the cross-task control analysis (Figure 1), we use seven SNI tasks: Yes/No, English-to-SQL, String Concatenation, SNLI (Bowman et al., 2015), MMLU Abstract Algebra (Hendrycks et al., 2021a), WebQA (Berant et al., 2013), and OpenPI (Tandon et al., 2020). For the warm-start initialization analysis (Figure 3), we evaluate on four SNI tasks: Fallacy, Style, Macroeconomics, and Computer Science. For each task, we compare SFT from random LoRA initialization against SFT initialized from a T2L-generated LoRA on the same target task.

SHINE (Liu et al., 2026). A context-level method that internalizes context into LoRA adapters through a hypernetwork. We use the released SHINE checkpoint with Qwen3-8B-Instruct (Yang et al., 2025a) as the base

model across all SHINE experiments. For the context-level QA benchmarks (Table 1), we evaluate on three QA groups: Single-hop uses SQuAD (Rajpurkar et al., 2016), Multi-hop uses MuSiQue (Trivedi et al., 2022), and SQuAD-N denotes SQuAD with contexts extended to approximately 2K tokens following SHINE's setup (Liu et al., 2026). For the reasoning gap analysis (Figure 2), we further evaluate SHINE on HotpotQA (Yang et al., 2018) (multi-hop), DROP (Dua et al., 2019) (numerical/discrete reasoning), and VitaminC (Schuster et al., 2021) (fact verification), alongside SQuAD as the single-hop reference. For the efficiency benchmark (Table 2), we measure peak memory, FLOPs, and latency on SQuAD.

Generative Adapter (Chen et al., 2025). A context-level method that compresses context into LoRA weights via end-to-end training. We use the released Generative Adapter checkpoint with Mistral-7B-Instruct (Jiang et al., 2023) as the base model across all Generative Adapter experiments. For the context-level QA benchmarks (Table 1), we evaluate on the same three QA groups as SHINE: Single-hop (SQuAD), Multi-hop (MuSiQue), and SQuAD-N. For the reasoning gap analysis (Figure 2), we evaluate Generative Adapter on SQuAD, HotpotQA, DROP, and VitaminC, matching the SHINE evaluation setup. For the efficiency benchmark (Table 2), we measure peak memory, FLOPs, and latency on SQuAD.

A.2 Evaluation Metrics

We use standard metrics for each task category.

Classification and multiple-choice tasks. We report **accuracy** for ARC-c, ARC-e, BoolQ, HellaSwag, OpenBookQA, PIQA, WinoGrande, GSM8K, MATH, SNLI, MMLU, and the SNI classification tasks used in the cross-task control analysis (Yes/No, English-to-SQL, String Concatenation, WebQA).

Code generation. We report **Pass@1** (Chen et al., 2021) for HumanEval, measuring the fraction of generated programs that pass all unit tests on the first attempt.

Text generation. We report **ROUGE-L F1** (Lin, 2004) for the SNI generation tasks (Grammar, Fallacy, Substitute, Style, Narrative) and for the warm-start initialization analysis (Fallacy, Style, Macroeconomics, Computer Science).

Context-level QA. We report **F1** for SQuAD, MuSiQue, SQuAD-N, HotpotQA, and DROP, measuring token-level overlap between predicted and reference answers. For VitaminC, we report **accuracy** for the three-way classification. The reasoning gap figure (Figure 2) plots these as a unified score for visual comparison.

Efficiency. We report peak GPU memory (GB, per GPU), FLOPs ($\times 10^{12}$), and latency (ms). For LoRA generation methods, these measure generating a single adapter; for SFT, they measure a full training cycle on the target task. All experiments use a single node with 4 NVIDIA H100 NVL GPUs (96 GB per GPU).

A.3 Additional Results

Table 5 presents a hint-controlled comparison on Gemma-2-2B-it across seven SNI tasks, where we add an explicit output or task hint to the base model (Base+Hint) and to the T2L adapter.

Task	Δ_{Base}	Δ_{T2L}	T2L–B+H	SFT–B+H
Fallacy	+5.88	-5.88	+5.88	+23.53
Style	+65.81	0.00	+10.26	+8.97
Substitute	+7.88	-6.75	-9.72	+5.62
Grammar	+32.23	-2.39	+4.00	+3.12
Paraphrase	+11.58	-3.30	+6.65	-0.04
BabI-Loc	+7.59	-2.00	+14.00	+24.00
EssTerms	+2.88	+12.39	-22.73	+14.46
Avg.	+19.12	-1.13	+1.19	+11.38

Table 5: **Hint-controlled comparison on Gemma-2-2B-it.** Δ_{Base} and Δ_{T2L} denote the effect of adding an explicit output or task hint to the base model and T2L adapter, respectively. B+H denotes Base+Hint, the base model with hint added to the prompt.