Test-Time Adaptation for LLMs via Hypernetwork-Generated LoRA Weights

Andrews George Varghese & Dhruv Kapur & Raj Maheshwari
10-423/623 Generative AI Course Project

December 11, 2025

Abstract

We propose a novel approach for test-time adaptation of large language models (LLMs) using hypernetworks to predict Low-Rank Adaptation (LoRA) weights on-the-fly during inference. Unlike traditional fine-tuning methods that require separate adapters for each task, we propose a hypernetwork that generates specialized LoRA parameters for individual inputs in a single forward pass enabling dynamic, instance-level model specialization. Building on recent advances in text-to-LoRA generation which utilize a per-task label to generate the corresponding LoRA matrices, we aim to extend this to true per-input adaptation, potentially improving model performance across diverse benchmarks while maintaining computational efficiency. This would thus eliminate the need to create labels or summary prompts per task, signifying a shift from static model specialization per dataset to a more dynamic, prompt-conditional parameter generation.

1 Introduction

Large language models have achieved remarkable success across numerous natural language processing tasks, yet adapting these models to specialized domains or tasks typically requires expensive fine-tuning procedures. Recent work on Low-Rank Adaptation (LoRA) [Hu et al., 2022] has dramatically reduced the computational burden by constraining parameter updates to low-rank decompositions, reducing trainable parameters by factors as high as 10,000x while matching full fine-tuning performance.

However, even LoRA-based approaches face a fundamental limitation: each task requires a separate adapter that must be trained in advance. This creates significant engineering overhead when supporting multiple tasks and prevents dynamic adaptation to individual inputs. Recent advances in hypernetworks [Ha et al., 2017]—neural networks that generate weights for other networks—offer a compelling solution. Sakana AI's Text-to-LoRA work [Charakorn et al., 2025] demonstrates that hypernetworks can generate task-specific LoRA adapters from natural language descriptions, achieving 98% of specialized adapter performance without any task-specific training time.

We extend this paradigm by investigating **instance-level adaptation**: using hypernetworks to generate unique LoRA parameters for each input during inference, rather than uniform task-level adapters in order to maximize performance on every single input. This approach aims to combine the efficiency of hypernetwork-based weight generation with the fine-grained customization potential of per-input adaptation, potentially improving performance on complex tasks that exhibit high intra-task variance, while also studying the behavior of the generated LoRA weights on out of distribution tasks.

2 Dataset / Task

We will evaluate our approach on a diverse set of established benchmarks to assess both the quality and generalization capabilities of our hypernetwork-generated LoRA adapters:

Reasoning Tasks: ARC-Challenge Clark et al. [2018] (2,590 grade-school science questions requiring complex reasoning beyond fact retrieval), GSM8K Cobbe et al. [2021] (8,500 grade-school math word problems requiring 2–8 step solutions), and WinoGrande Sakaguchi et al. [2020] (44,000 common-sense reasoning problems via pronoun resolution).

Code Generation: HumanEval Chen et al. [2021] (164 hand-written Python programming problems), and MBPP Austin et al. [2021] (974 basic Python tasks with assert-style test cases).

Knowledge & Comprehension: BoolQ Clark et al. [2019] (15,942 yes/no questions requiring inference from context), OpenBookQA Mihaylov et al. [2018] (5,957 science questions requiring multi-hop reasoning), and HellaSwag Zellers et al. [2019] (70,000 commonsense inference problems about physical scenarios).

We use standard evaluation metrics for each benchmark: accuracy for multiple-choice tasks (ARC, BoolQ, OpenBookQA, HellaSwag, WinoGrande), exact-match accuracy for GSM8K, and pass@1 (functional correctness) for code generation tasks.

As for model choice, we will begin experimenting with a pretrained model such as Pythia 70M/160M [Biderman et al., 2023] or SmolLM 135M [Allal et al., 2024], due their small size, and if time and compute budgets allow, we are also keen on experimenting with instruction-tuned models like SmolLM 135M Instruct or Gemma3 270M [Gemma, 2025].

3 Related Work

Low-Rank Adaptation. LoRA [Hu et al., 2022] established the currently most prevalent method of parameter-efficient fine-tuning by representing weight updates as $\Delta W = BA$ where $B \in \mathbb{R}^{d \times r}$ and $A \in \mathbb{R}^{r \times k}$ with rank $r \ll \min(d, k)$. This reduces trainable parameters by 10,000x for GPT-3 175B while matching full fine-tuning performance.

Hypernetworks. The fundamental paradigm of using one network to generate weights for another was established by Ha et al. [2017], demonstrating end-to-end training via backpropagation for generating non-shared LSTM weights. This concept was extended to transformers by HyperFormer Mahabadi et al. [2021], which generates adapter parameters for all layers by conditioning on task embeddings, layer position, and module type, achieving strong multi-task performance while adding only 0.29% parameters per task.

Meta-Learning for Fast Adaptation. Model-Agnostic Meta-Learning (MAML) [Finn et al., 2017] introduced gradient-based meta-learning through bi-level optimization, enabling rapid task adaptation with few gradient steps. While MAML requires gradient updates during adaptation, hypernetwork approaches can generate task-specific parameters in a single forward pass, representing complementary trade-offs between adaptation quality and computational efficiency.

Text-to-LoRA and Test-Time Adaptation. Most relevant to our work, Charakorn et al. Charakorn et al. [2025] showed that hypernetworks can generate complete LoRA adapters directly from natural-language task descriptions, compressing hundreds of task-specific adapters into a single model. In parallel, HyperDecoders Ivison and Peters [2022] proposed a sequence-to-sequence architecture that decodes serialized parameters (e.g., adapters or classification heads) conditioned on natural-language instructions. Unlike fixed-format hypernetworks, HyperDecoders treat parameter generation as an autoregressive text-generation problem, which are then injected into a base model for task execution.

While Text-to-LoRA operate ats *task-level granularity* producing one set of parameters per task description, HyperDecoder approaches only NLP subproblems instead of language modeling tasks.

Our work extends these ideas to *instance-level adaptation*, generating unique LoRA parameters for each inference input to maximize per-example performance on heterogeneous data in the language modeling paradigm.

4 Approach

Our approach builds upon the Text-to-LoRA architecture while introducing instance-level conditioning mechanisms:

Architecture. We will implement a transformer-based hypernetwork that takes as input: (1) embeddings of the input text using a frozen encoder (e.g., `gte-large-en-v1.5` [Li et al., 2023], `ModernBERT-base` [Warner et al., 2024]), `NeoBERT` [Breton et al., 2025], or alternatively a frozen copy of the target language model itself (2) learned module-specific embeddings identifying target layers (query and value projections), and (3) layer-specific positional information. (2) and (3) will build on work by Sakana [Charakorn et al. [2025]]. The hypernetwork processes these concatenated embeddings and generates LoRA weight matrices (B and A) via linear projection heads.

Training Strategy. We propose training the hypernetwork end-to-end on downstream tasks, back-propagating through the (frozen) language model and attached predicted adapter weights to the hypernetwork. This updates the hypernetwork's parameters to learn meaningful mappings from input semantics to valid LoRA weight structures that maximize the performance of the language model on training examples. An alternate idea is to train the hypernetwork using a reconstruction loss, i.e. per-task LoRA adapters are learned using PEFT and saved to create a supervised dataset that maps inputs to their corresponding LoRA adapters. The hypernetwork is then trained to reconstruct LoRA weights from the input questions from the dataset. However, in additon to requiring hundreds LoRA adapters as training examples, [Charakorn et al., 2025] also shows that this method fails to generalize .

Instance-Level Conditioning. Unlike Text-to-LoRA which conditions only on task descriptions, we will condition the hypernetwork on each individual input sequence. This enables the generation of input-specific LoRA weights that can adapt to intra-task variations, edge cases, and specific input characteristics, and crucially, doesn't require additional task labeling/description generation for the hypernetwork input.

Baseline Method. Our primary baseline is standard LoRA fine-tuning on each benchmark dataset, which represents the current state-of-the-art for parameter-efficient adaptation, as well as testing for performance against the base model on out-of-distribution benchmarks.

Key Contribution. Our primary novel contribution is extending hypernetwork-based LoRA generation from task-level to *instance-level* adaptation, validating the feasibility of such test-time adaptation for LLMs, and identifying the feasibility of such a method for handling out-of-distribution inputs. This requires developing effective conditioning mechanisms that can extract input-specific features predictive of optimal LoRA adapter configurations, while tackling computational efficiency problems such as handling different LoRA weights for each element in a batch during training, as well as training issues such as avoiding overfitting to individual examples.

5 Expected Outcomes

We hypothesize that instance-level LoRA adaptation should have a reasonable performance in comparison to task-specfic LoRA adapters, with the potentional to outperform the baseline on benchmarks with high within-task diversity. Specifically:

Performance Targets: We expect to achieve 85-95% of standard LoRA performance (which itself matches or exceeds full fine-tuning) across benchmarks, with the potential to exceed task-level approaches on complex reasoning tasks like Arc-Challenge and GSM8K where individual problems may benefit from specialized adaptations.

Efficiency Gains: Following Text-to-LoRA's results, we anticipate computational efficiency compared to adapter training (just a single forward pass through the hypernetwork), while also potentially achieving better performance on benchmarks than in-context learning.

Generalization: We hope to demonstrate generalization to held-out benchmarks, showing that the hypernetwork learns generalizable mappings from input characteristics to optimal LoRA configurations rather than memorizing task-specific solutions.

Analysis: We will analyze the learned latent representations to understand what characteristics of the inputs drive LoRA parameter selection, visualize the diversity of generated adapters across inputs, and identify which types of problems benefit most from instance-level adaptation through comparison with the baseline.

5.1 Potential Challenges

Key challenges include: (1) achieving stable training where the hypernetwork actually learns to predict valid and usable LoRA wieghts, (2) preventing overfitting to individual training examples, (3) balancing hypernetwork capacity against memory constraints, (4) developing robust regularization strategies and (5) maximizing deliverables given our compute and time constraints.

6 Plan

There are 4 primary tasks that need to be completed for this project:

1. Building a flexible evaluation framework that we will use to evaluate the baseline language model, the task-specific fine-tuned models and our hypernetwork augmented model on the selected benchmarks. Available options like Harness [Gao et al., 2024] and LightEval [Habib et al., 2023] provide reliable evaluation pipelines but are too opaque to be easily used in this project.

2. Creating the PEFT pipeline to train the per-task LoRA adapters.

3. Building and training the hypernetwork.

4. Analysing the generated LoRA adapters.

All team members will participate in hyperparameter tuning, debugging, and results interpretation. We will use pair programming for critical components like the hypernetwork forward pass and hypernetwork output analysis.

Midway Executive Summary Deliverables: By the midway point, we will have: (1) completed preliminary hypernetwork architecture implementation, (2) established evaluation pipeline and used it to evaluate the base model and (3) PEFT pipeline for per-benchmark fine-tuning.

References

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Leandro von Werra, and Thomas Wolf. Smollm - blazingly fast and remarkably powerful, 2024.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, Sharad Ramamurthy, David Bieber, Aleksandar Makelov, Chenjie Xiong, Michihiro Yasunaga, and Qian Lin. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR, 2023.

Lola Le Breton, Quentin Fournier, Mariam El Mezouar, and Sarath Chandar. Neobert: A next-generation bert, 2025. URL `https://arxiv.org/abs/2502.19587`.

Rujikorn Charakorn, Edoardo Cetin, Yujin Tang, and Robert Tjarko Lange. Text-to-LoRA: Instant transformer adaption. In *Forty-second International Conference on Machine Learning*, 2025. URL `https://openreview.net/forum?id=zWskCdu3QA`.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Kate Ray, Raahil Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Ashish Power, Lukasz Kaiser, Mohammad Bavarian, Robert Winter, Phil Tillet, Felipe Such, David Cummings, Matthias Plappert, Feynman Chantzis, Alex Barnes, Ariel Herbert-Voss, Will Guss, Kamal Dwivedi, Trent Millard, Shantanu Ogawa, Sam Thompson, Time Chen, Amos Azaria, Chris McLeavey, Edward Farhi, Josh Achiam, Sandhini Agarwal, Andrew Lohn, Vicki Zhuang, Jack Clark, Dario Amodei, Ingrid Kaplan, Tom McCandlish, and Ilya Sutskever. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Christopher Clark, Kenton Lee, Ming-Wei Chang, and Tom Kwiatkowski. 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*, 2019.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

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

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR, 2017.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. URL `https://zenodo.org/records/12608602`.

Team Gemma. Gemma 3. 2025. URL `https://goo.gle/Gemma3Report`.

David Ha, Andrew M Dai, and Quoc V Le. Hypernetworks. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=rkpACe1lx`.

Nathan Habib, Clémentine Fourrier, Hynek Kydlíček, Thomas Wolf, and Lewis Tunstall. Lighteval: A lightweight framework for llm evaluation, 2023. URL `https://github.com/huggingface/lighteval`.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=nZeVKeeFYf9`.

Hamish Ivison and Matthew E. Peters. Hyperdecoders: Instance-specific decoders for multi-task nlp, 2022. URL `https://arxiv.org/abs/2203.08304`.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*, 2023.

Rabeeh Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. 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 565–576. Association for Computational Linguistics, 2021.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 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*, 2018.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8732–8740, 2020.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, and Iacopo Poli. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference, 2024. URL `https://arxiv.org/abs/2412.13663`.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.