Bayesian Optimization / Scientific Discovery

A Language-Guided Bayesian Optimization for Efficient LoRA Hyperparameter Search

Authors: Baek Seong-Eun, Lee Jung-Mok, Kim Sung-Bin, Tae-Hyun Oh

Affiliation: POSTECH, KAIST

Publication: ICML (accepted)

Year: 2026

Link: https://arxiv.org/pdf/2602.11171

---

1. Summary

Problem

- What problem does the paper address?
    - The paper addresses the challenge of efficiently finding optimal hyperparameters for LoRA, which is highly sensitive to configurations like rank, scaling factor, and learning rate.
    - Exhaustive search is computationally prohibitive because every evaluation requires expensive fine-tuning of LLMs.
- **Why is it important?**
    - While LoRA is a resource-efficient alternative to full fine-tuning, its effectiveness depends on complex, entangled hyperparameter choices that vary by dataset and model.
    - Standard Bayesian Optimization (BO) often fails here because it cannot naturally incorporate domain-specific knowledge and struggles with the discrete, structured nature of the LoRA hyperparameter space

Method

- **Core idea:**
    - The framework utilizes a frozen pre-trained LLM (specifically Qwen2-7B) as a discrete-to-continuous mapping module.
    - It converts hyperparameter configurations and domain-aware prompts into continuous embeddings, enabling the performance of Deep Kernel Learning (DKL) for BO.

- **Novel technique(s):**
    - **Domain-aware Prompting:** Incorporating textual explanations of hyperparameter roles and interactions (e.g., how scaling factor relates to rank) to inject expert inductive bias.
    - **Learnable Token & Projection Layer:** Introducing a learnable token (ψ) and a projection layer (P) with ELU activation to transform raw LLM embeddings into a smooth, discriminative space tailored for BO.
    - **Proxy Training Evaluation:** Exploiting the strong correlation between subset and full-dataset training to evaluate configurations on a randomly selected 10% subset, reducing computational costs by up to 10x.

Main Contributions

1. Proposes the first framework combining an LLM with BO specialized for LoRA HPO, allowing for domain-aware and sample-efficient search.
2. Introduces an efficient optimization design featuring calibrated feature extraction and a proxy evaluation protocol that significantly reduces evaluation time.
3. Demonstrates broad generalizability across LoRA variants (PiSSA, DoRA, rsLoRA), diverse backbone architectures (Mistral, Gemma, Llama2), and even general machine learning tasks in the Bayesmark benchmark.

Experimental Setup

- **Task:**
    - Math reasoning, Code generation, Conversation
- **Dataset(s):**
    - Fine-tuned on MetaMathQA and evaluated on GSM8K and MATH.
    - Fine-tuned on CodeFeedback and evaluated on HumanEval and MBPP.
    - Fine-tuned on WizardLM-Evol-Instruct and evaluated on MT-Bench.
- **Baseline(s):**
    - Random search, Optuna, Standard BO, Latent BO (LBO), NOMAD
- **Evaluation Metric(s):**
    - Accuracy (%) for math
    - Pass@1 for code
    - GPT-Score for conversation

Experimental Results

- **Best result:**

    - Proposed method achieved 21.46% accuracy improvement on GSM8K (from 41.47% to 62.93%) compared to standard LoRA settings within just 30 iterations.
- **Comparison to baselines:**
    - The framework outperformed all HPO baselines (e.g., scoring 62.93% on GSM8K vs. LBO's 59.51% and NOMAD's 52.16%) while being significantly more time-efficient than dedicated methods like NOMAD.
- **Key takeaway:**
    - Injecting domain-specific knowledge through text templates and calibrating embeddings creates a smoother, more discriminative optimization landscape, allowing BO to find superior configurations more efficiently than standard black-box methods.

---

2. Background & Motivation

Problem Statement

- LoRA's performance is highly dependent on a large set of hyperparameters, including rank, scaling, and learning rate
- These factors are entangled in complex ways, making systematic configuration critical yet difficult because the search space contains over 45,000 combinations and full evaluation is extremely costly.

Why is this problem important?

- As LLMs grow in size, full fine-tuning becomes impractical, making parameter-efficient methods like LoRA essential.
- However, "near-optimal" configuration changes based on the model architecture and specific task, meaning a one-size-fits-all approach often leads to sub-optimal results.

Limitations of Previous Work

- Traditional black-box HPO methods are designed for smooth, continuous spaces, failing to handle the discrete and structured nature of LoRA parameters.
- Existing approaches also lack a mechanism to incorporate domain-specific expert know-how, leading to unreliable surrogate models when evaluation data is sparse.

3. Related Work

Existing Approaches

- **Black-box Optimization:** Methods like random search, Optuna, and standard BO.
- **LoRA-Specific Strategies:** Tools like NOMAD and other efficient search strategies specifically for LoRA.
- **LLM-enhanced BO:** Recent studies have suggested using LLMs to enhance BO through natural language priors, embeddings, or agent-based mechanisms.
- **Variants:** Efforts to improve LoRA's stability and performance through architectural variants like PiSSA, DoRA, or stabilize-scaling factors.

Differences from Prior Work

- This work is the first to integrate BO with LLMs specifically for LoRA HPO.
- Unlike prior work that uses simple prompting, this method uses domain-aware text templates and a trainable projection layer to calibrate the embedding space, ensuring it is specifically optimized for the LoRA performance landscape.

4. Method

Overview

- The framework repurposes an LLM as a discrete-to-continuous mapping module.
- It uses structured language to guide a BO process, leveraging both the LLM's pre-trained knowledge and learned calibrations to identify high-performing hyperparameters efficiently.

Pipeline

1. **Proxy Evaluation:** Fine-tune LoRA on a 10% data subset to obtain a proxy performance score (y_n).
2. **Feature Extraction:** Convert the configuration into a domain-aware text template (t_n), combine it with a learnable token (ψ), and pass it through the frozen LLM and projection layer to create an embedding (z_n).
3. **Surrogate Update:** Update a Gaussian Process (GP) surrogate model by jointly optimizing its kernel parameters, the projection layer, and the learnable token via marginal log-likelihood.
4. **Configuration Selection:** Use an acquisition function (Expected Improvement) to suggest the next configuration from the candidate pool.

Model Architecture

- The architecture is formulated as LLM-based DKL, where the GP kernel is defined over transformed LLM hidden states.
- It uses a frozen Qwen2-7B for feature extraction (3584-dimensional hidden states), followed by a projection layer consisting of a linear layer, dropout, ELU activation.
- The surrogate utilizes a Matérn 5/2 kernel to model the performance distribution over the calibrated embedding space.

Key Components

Component	Purpose
Domain-aware Prompting	Explicitly injects LoRA expert knowledge (e.g., rank/alpha relationships) into the embedding space via structured text templates.
Automatic Prompting	Uses a surrogate model and GPT to extract hyperparameter structures from optimization feedback when manual expert knowledge is unavailable.
Learnable Token (ψ)	Captures residual domain information not easily expressed in natural language, helping the embedding implicitly internalize LoRA-specific knowledge.
Projection Layer (P)	Calibrates high-dimensional LLM hidden states into a smooth, discriminative feature space specifically tailored for BO tasks.
Matérn 5/2 Kernel	Measures covariance between configurations in the embedding space to approximate the complex black-box performance function.
Expected Improvement (EI)	Acts as the acquisition function to balance the exploration of new configuration regions and the exploitation of known high-performing areas.

Loss Functions

- The kernel parameters (ω), projection layer parameters (θ), and the learnable token (ψ) are jointly optimized by maximizing the marginal log-likelihood.
- This optimization is performed using gradient-based methods by applying the chain rule to the log-likelihood objective.

Training Strategy

- While the mapping LLM remains frozen, the projection layer and learnable token are dynamically updated at every BO iteration based on all currently collected performance data.
- This ensures the embedding space becomes increasingly refined and representative of the true performance landscape as the search progresses.

5. Experiments

Datasets

- **Math Reasoning:** Fine-tuned on MetaMathQA (100K samples) and evaluated on GSM8K and MATH.
- **Code Generation:** Fine-tuned on CodeFeedback (100K samples) and evaluated on HumanEval and MBPP.
- **Conversation:** Fine-tuned on WizardLM-Evol-Instruct (100K samples) and evaluated on MT-Bench.
- A 10K subset (10%) of each training dataset is used for the proxy training evaluation.

Experimental Settings

- The search space includes five key hyperparameters:
    - Rank (r from 1 to 256)
    - Scaling Factor (α relative to r)
    - Batch Size (2 to 256)
    - Learning Rate (1e-6 to 5e-3)
    - Dropout Rate (0.0 to 0.3).
- The optimization budget is typically set to 30 iterations.
- The LLM (ϕ) remains frozen during optimization, while the projection layer and learnable token are jointly trained with the GP kernel parameters.

Evaluation Metrics

- Accuracy (%) for math reasoning tasks (GSM8K and MATH).
- Pass@1 for code generation tasks (HumanEval and MBPP).
- GPT-Score for conversation tasks (MT-Bench).

Baseline Methods

- Random search, Optuna, Standard BO, Latent BO (LBO), NOMAD, LLAMBO

6. Results & Analysis

Main Results

- The method achieved a 21.46% accuracy improvement on GSM8K and demonstrated consistent gains across variants like DoRA, rsLoRA, and PiSSA.
- It revealed that optimal configurations often utilize higher ranks than standard baselines and discovered effective settings where the scaling factor (α) is 16 or 32 times larger than the rank (r).
- The framework generalizes to architectures like Mistral, Gemma, and Llama2, and even to non-LoRA tasks in the Bayesmark benchmark for Decision Trees and Random Forests.

Search Method	Accuracy (%)		Pass@1	
	GSM8K	MATH	HumanEval	MBPP
Random	59.14	10.51	23.17	36.77
Optuna	54.13	10.50	27.44	38.62
BO	57.32	11.42	20.12	35.19
LBO	59.51	11.88	26.83	37.83
Ours	**62.93**	**12.88**	**30.49**	**42.59**

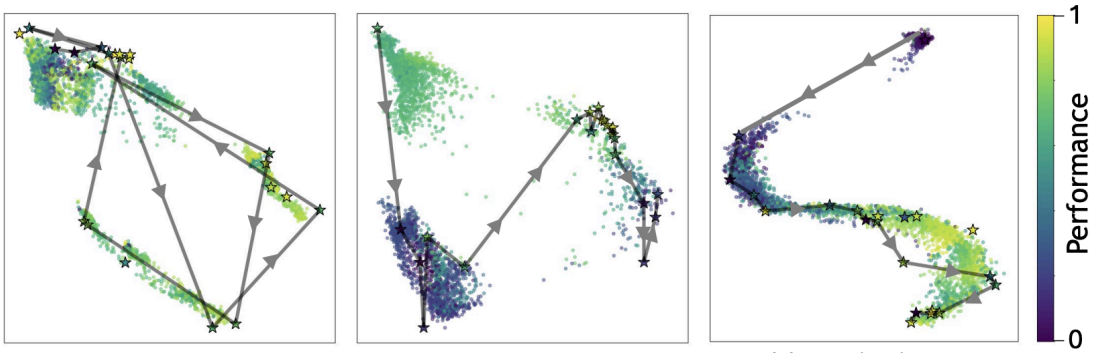
Ablation Studies

- Ablation results show that domain-aware prompting is the most crucial component, providing the largest individual performance gain by injecting explicit inductive bias.
- Removing the projection layer or learnable token leads to an entangled embedding space where high- and low-performing configurations are poorly separated, causing BO trajectories to lack clear direction.

Projection Layer	Domain-aware Prompting	Learnable Token	GSM8K	MATH
✗	✗	✗	47.76	8.72
✓	✗	✗	53.98	9.16
✓	✓	✗	61.41	12.46
✓	✓	✓	**62.93**	**12.88**

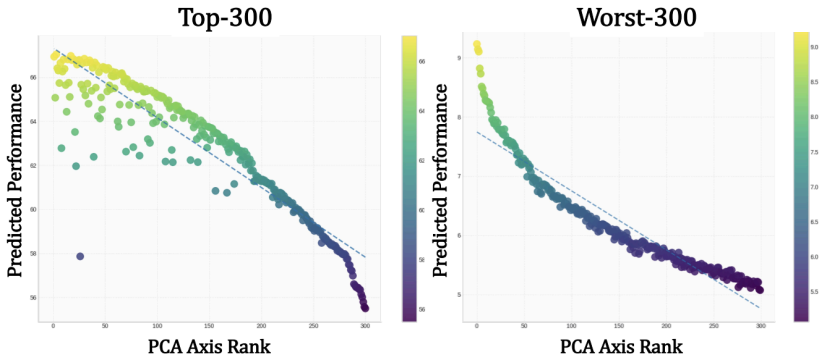
Qualitative Analysis

- Optimization trajectories demonstrate that calibrated embeddings guide the search consistently toward high-performing regions, whereas frozen embeddings results in an unstable and poorly directed search.



(a) $\phi(t)$ (b) $P(\phi(t);\theta)$ (c) $P(\phi(t,\psi);\theta)$

- Principal Component Analysis (PCA) of the learned embeddings reveals a strong monotonic relationship (Spearman's ρ=0.8779) between the dominant axis of the embedding space and predicted performance.
- In other words, statistical analysis confirms that the primary direction of the calibrated embedding space effectively sorts hyperparameter configurations from worst to best with a near-perfect correlation of 0.8779 to predicted performance.



Failure Cases / Error Analysis

- Hyperparameter configurations discovered by the framework are highly architecture-dependent.
- Transferring configurations between different model series (e.g., from Qwen2.5 to Llama2) results in substantial performance degradation, proving that the "near-optimal" settings found for one model family do not generalize to others.

Model	Settings	GSM8K	MATH
LLaMA2-7B	LLaMA2-7B	62.93	12.88
	Qwen2.5-3B	39.5	5.2
	Qwen2.5-7B	32.68	4.7
	Qwen2.5-14B	52.46	8.06
Qwen2.5-7B	Qwen2.5-7B	83.93	48.08
	LLaMA2-7B	81.12	41.06
	LLaMA2-13B	80.06	40.18

7. Discussion

Strengths

- **Sample and Computational Efficiency:** The framework identifies near-optimal configurations within only 30 iterations by utilizing a proxy evaluation protocol on 10% of the training data, which reduces total optimization time by up to 10x.
- **Domain-Awareness:** It effectively bridges the gap between discrete hyperparameter spaces and continuous optimization by using structured text templates that inject explicit LoRA expert knowledge into the surrogate model.
- **Scale-Invariant Transferability:** Hyperparameters discovered on smaller models (e.g., Qwen2.5-3B) remain highly effective when transferred to larger versions (e.g., Qwen2.5-14B) within the same model family, allowing for significant cost savings during the tuning process.
- **Broad Generalizability:** The method operates as a plug-and-play module compatible with various LoRA variants (PiSSA, DoRA, rsLoRA), diverse architectures (Mistral, Gemma, Llama2), and even general machine learning tasks like optimizing Decision Trees and Random Forests.

Limitations

- **Architectural Sensitivity:** While transferable across scales, the "near-optimal" settings are highly dependent on model architecture; applying configurations found for one model series (e.g., Qwen2.5) to another (e.g., Llama2) leads to substantial performance degradation.
- **Resource Requirements:** Despite the 10x efficiency gain, the process still requires repeated fine-tuning cycles (one per iteration), which may be prohibitive for users with extremely limited computational budgets or for the very largest model scales.
- **Security and Accuracy Risks:** As a tool for fine-tuning LLMs, there is a potential risk of misuse in security-sensitive areas or specialized fields (like healthcare) where small errors in hyperparameter-driven performance could lead to severe consequences.

Future Work

- **Expansion of Tuning Strategies:** The authors expect this framework to serve as a practical baseline for optimizing broader fine-tuning strategies beyond LoRA and other parameter-efficient fine-tuning (PEFT) methods.
- **Embedding Model Optimization:** Future research could systematically investigate how different embedding LLM properties, such as model scale, architecture, or

hidden-state dimensionality, influence the quality of the learned hyperparameter representations.

- **High-Knowledge Domains:** There is potential to apply this language-guided optimization to complex fields requiring deep domain-specific knowledge, such as medical diagnostics, manufacturing processes, and catalyst discovery.

8. Personal Insights

Key Takeaways

- **Calibration is Critical:** Raw LLM embeddings are too "messy" for effective optimization; the introduction of a projection layer and learnable token is what successfully organizes the search space into a smooth gradient of performance.
- **The "Rule of Two" is Obsolete:** Traditional guidelines suggest setting the scaling factor (α) to twice the rank (r), but this framework discovered that α values 16 or 32 times larger than r often yield significantly higher accuracy.
- **Architecture Trumps Scale:** Optimal hyperparameters are highly transferable between different sizes of the same model (e.g., from 3B to 14B) but fail when transferred between different architectures (e.g., from Qwen to Llama), meaning tuning is an architecture-specific requirement.
- **Efficiency through Proxies:** Using a fixed 10% random subset for training evaluation provides a correlation of over 0.87 with full-dataset performance, proving that we can find near-optimal settings with 1/10th of the computational cost.

Ideas Relevant to the SaeGyeol AI Project

- **Optimizing Sovereign and Local Models:** SaeGyeol selects between frontier and sovereign models. This framework can be used within the secure control plane to automatically fine-tune local/sovereign models (like Llama or Qwen) for specific institutional tasks. By using the proxy training protocol, SaeGyeol can identify the best hyperparameters for these models on masked or de-identified data subsets (10% of institutional data) before committing high-resource GPU cycles to a full-scale sensitive run.
- **Automated Execution Routing via Knowledge Extraction:** The framework's "Automatic Prompting" capability—which extracts hyperparameter structures from optimization feedback—could be applied to SaeGyeol's execution route selection. The system could analyze the "performance-to-cost" logs of various workflows to

automatically generate textual descriptions of which model/tool configurations are best for specific data sensitivity levels.

- **IP Protection via Domain-Aware Prompting:** Instead of sending raw, sensitive training logs to an external optimization service, SaeGyeol can use the Domain-aware Prompting method. Institutions can describe their specific hardware constraints and research goals in natural language prompts that stay within the Private AI Control Plane, allowing the LLM-guided BO to optimize the workflow without exposing raw research data.
- **Auditability and Scale-Invariant Tuning:** Since the paper proves that hyperparameters found on a 3B model scale effectively to a 14B model of the same family, SaeGyeol can perform its initial optimization on smaller, cheaper local GPU models. Once the optimal configuration is verified and recorded for auditability, it can be confidently deployed to larger research infrastructure, ensuring that the transition from local testing to institutional scale is both high-performing and cost-effective.