Data-Efficient LLM Fine-Tuning by Noise Resistance

Anonymous ACL submission

Abstract

Fine-tuning large language models (LLMs) with limited data is challenging due to the entanglement of task features and sample-specific noise. We introduce **Noise Flushing**, a paradigm shift that prioritizes removing noise rather than solely amplifying the weak task signal. Like gold panning, where water washes away sand, Noise Flushing uses abundant irrelevant data – sampled from the LLM itself – to "flush away" noise during LoRA fine-tuning. This constrains the LoRA adapter to suppress noise and focus on task-relevant features. Theoretically, we show that Noise Flushing can achieve performance comparable to vanilla fine-tuning with drastically fewer task samples. Empirically, with remarkably few task examples, Noise Flushing achieves significant improvements over strong fine-tuning baselines on translation, structured text generation, even special token understanding with fewer than 100 samples. Noise Flushing transforms LLMs into statistical "gold panners", learning what to ignore to efficiently learn from sparse data.

1 Introduction

Large Language Models (LLMs) have revolutionized numerous applications, yet effectively adapting them to specialized domains often hinges on fine-tuning. Instruction tuning, a prevalent fine-tuning paradigm that trains models on instruction-response pairs (Zhao et al., 2024; Zhou et al., 2023), encounters significant hurdles when task-specific data is severely limited. In these scenarios, the model's learning process becomes highly inefficient, demanding substantial computational resources and often yielding unsatisfactory results.

The fundamental challenge in extreme low-data finetuning, we posit, stems from the inherent entanglement of **task-specific features with sample-specific stochastic noise** within the sparse data. With only a handful of examples, LLMs are prone to overfitting to noise – fleeting patterns unique to the few samples – rather than grasping the underlying task semantics. While traditional approaches like data augmentation and synthetic data generation (Li et al., 2023; Zhao et al., 2024) attempt to bolster the weak task signal, they often fall short. These methods struggle when data is not only scarce but also specialized or noisy, making it difficult to discern true task features from spurious correlations even with augmented datasets. The core problem remains: how to effectively learn when the signal is faint and buried within noise, and when even discerning signal from noise is a challenge given data scarcity.

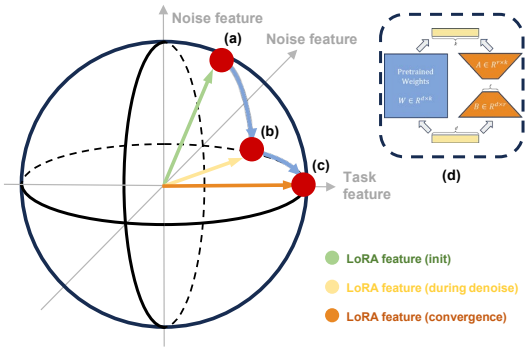

Figure 1: **Overview of Noise Flushing:** Irrelevant data guides LoRA-learned features towards task features. (a) Initial LoRA features (from task data) contain a mix of task features and noise. (b) Loss minimization on self-sampled irrelevant data forces noise suppression. (c) Convergence results in noise suppression, retaining primarily salient task features. (d) The low-rank bottleneck in LoRA's structure inherently promotes noise rejection while preserving task features. See 4.3 for validation of precise task feature capture.

Inspired by the intuition that effective learning involves not just amplifying the signal but also actively mitigating noise, we introduce **Noise Flushing**, a novel framework for data-efficient fine-tuning of LLMs. This approach represents a paradigm shift: instead of solely focusing on strengthening the weak task signal in low-data

regimes, Noise Flushing prioritizes the removal of pervasive sample-specific noise. To visualize this, consider the analogy of gold panning: just as water washes away sand to reveal gold, Noise Flushing uses abundant self-sampled irrelevant data ("water") to selectively "flush away" the noise ("sand") that obscures the underlying task-relevant features within limited task data.

This noise removal process allows the fine-tuned model to concentrate its learning capacity on the sparse yet crucial task-relevant features ("gold"). By strategically leveraging the LLM's own generated data as "noise flushing" agents, Noise Flushing guides the LoRA adapter to effectively reject noise and focus on learning genuine task-relevant features. Theoretically, we demonstrate that this approach leads to a significant reduction in the *absolute quantity* of task data required for effective fine-tuning. While the fundamental complexity related to achieving a target error is maintained, our analysis suggests that mixing a small number of task samples with irrelevant data can achieve performance comparable to traditional methods, but with potentially orders of magnitude fewer task-specific examples needed.

Our empirical evaluation showcases the substantial advantages of Noise Flushing over standard instruction tuning and other robust baselines, especially in extremely low-data regimes. Specifically, Noise Flushing achieves markedly higher accuracy in formatted text generation and improved BLEURT scores in translation, all while maintaining robust semantic consistency for special tokens – a critical aspect often considered extremely hard in low-data finetuning.

The contributions of our work are:

- We reframe data-inefficient instruction tuning as a noise disentanglement problem, emphasizing the entanglement of task and noise features as the primary bottleneck in low-data learning.
- We introduce Noise Flushing, a novel approach that, both theoretically and empirically, demonstrates the efficacy of mixing sparse task data with abundant, self-sampled irrelevant data to enhance task learning by actively suppressing noise.
- We show that Noise Flushing achieves robust semantic consistency for new special tokens, addressing a key limitation of current knowledge injection techniques in LLMs.

2 Related Work

2.1 Challenges in Data-Efficient Instruction Tuning

Instruction tuning, while effective, faces significant challenges in data-scarce settings. Sclar et al. (2023) demonstrated the sensitivity of instruction-tuned models to prompt variations, highlighting robustness issues. Sun and Dredze (2024) observed overfitting to task formats, limiting generalization. Wang et al. (2022b) argued that models become overly specialized, failing to learn underlying semantics, which aligns with our perspective on the entanglement of task and noise features. These works collectively point to the data inefficiency of vanilla instruction tuning, especially when task samples are limited.

2.2 Data-Efficient Fine-Tuning Approaches

Researchers have explored various approaches to improve data efficiency in fine-tuning. Some focus on data augmentation techniques to expand the effective size of task datasets. Others investigate meta-learning approaches to learn how to learn from few samples. Li et al. (2023) and Zhao et al. (2024) proposed iterative self-improvement frameworks to enhance training data quality. Vernikos et al. (2020) modified training objectives to mitigate overfitting. Architectural modifications, like using PEFT modules (Wang et al., 2022a) or embedding noise injection (Jain et al., 2023), have also been explored for better generalization.

2.3 Leveraging Irrelevant or Diverse Data

The idea of using diverse or seemingly irrelevant data to improve model performance has been explored in different contexts. In domain adaptation, diverse source domains are used to improve generalization to a target domain. In contrastive learning, diverse negative samples are crucial for learning robust representations. Recent work has explored using synthetic data for instruction tuning (Liu et al., 2023; Dong et al., 2024; Mecklenburg et al., 2024).

However, the specific mechanism of how irrelevant data can aid in noise suppression and improve data efficiency in instruction tuning, particularly within a PAC-Learning framework, remains relatively unexplored, which is the focus of our work.

Algorithm 1 Noise Flushing for Data-Efficient Fine-Tuning

Input: Pre-trained LLM, Task dataset D_{task}, Irrelevant queries Q_{irr}
Parameter: Mixing ratio r (ratio of irrelevant data to task data per training step)
Output: Finetuned LLM

```
Initialize fine-tuned model with pre-trained LLM and LoRA
D_irr ← Sample QA pairs from LLM with Q_irr
Prepare dataset by mixing D_task and D_irr with ratio 1 : r.
for each epoch do
    for each batch in shuffled combined dataset do
        Train LLM with LoRA on the batch.
        Update LoRA parameters.
    end for
end for
return Fine-tuned LLM
```

Our approach differs from methods relying on carefully crafted synthetic data by exploring the benefits of readily available, diverse, and even irrelevant data for noise flushing and efficient task learning.

3 Method

3.1 Problem Formulation: Noise Entanglement in Instruction Tuning

Instruction tuning in low-data settings suffers from the entanglement of task-specific features and sample-specific stochastic noise. We decompose the input feature space into orthogonal Task Features subspace F and Noise Features subspace G, with projection operators P_F and P_G respectively, such that for any input x, $x = P_F x + P_G x$. The ideal model update $\Delta^*(x)$ should primarily reside in the task feature subspace: $P_G \Delta^*(x) \approx 0$. However, with limited task data, standard fine-tuning struggles to achieve this disentanglement.

We show it as a reasonable assumption in the Appendix B.

3.1.1 LoRA for Efficient Fine-tuning and Noise Control

We utilize Low-Rank Adaptation (LoRA), which updates pre-trained weights W with a low-rank matrix $\Delta W = AB$, where $\text{rank}(AB) \ll \text{rank}(W)$. The model response becomes $r(x) = Wx + BAx$. The goal of LoRA in Noise Flushing is to learn matrices A and B such that the update $\Delta W = AB$ effectively captures task features and suppresses noise features.

3.2 Noise Flushing with Irrelevant Data: The Mechanism

Noise Flushing uses self-sampled irrelevant data D_{irr} alongside task data D_{task} to disentangle features. We hypothesize that task data D_{task} contains both task signals $P_F x$ and noise $P_G x$, while irrelevant data D_{irr} predominantly contains noise features $P_G x'$ with negligible task signals $P_F x' \approx 0$. During training, the mixed dataset aims to minimize a loss function $\mathcal{L}$ that implicitly encourages noise suppression. Specifically, when training on irrelevant data self-sampled D_{irr}, the objective is to minimize the model's response to irrelevant inputs, effectively driving the LoRA update towards satisfying:

$$\|BAx'\|^2 \to 0, \quad \text{for } x' \in D_{\text{irr}}.$$

This minimization process encourages the LoRA adaptation BA to become less sensitive to features present in irrelevant data, which are hypothesized to be primarily noise features G. Conversely, task data drives the learning of task-specific features F. By mixing D_{task} and D_{irr}, Noise Flushing guides LoRA to learn updates that are selective: amplifying task features while suppressing noise. Algorithm 1 outlines the data mixing and LoRA fine-tuning process. Appendix B support its practicability.

3.3 Theoretical Justification for Noise Flushing

Our theoretical analysis, detailed in Appendix A, provides PAC-Learning guarantees. Key theorems are:

3.3.1 Theorem 1: Task-Only Sample Complexity

For task-only fine-tuning, achieving a task error ϵ_{task} requires $n_{\text{task}} = O\left(1/\epsilon_{\text{task}}^2\right)$ task samples.

3.3.2 Theorem 2: Mixed-Data Sample Complexity and Convergence

Noise Flushing, by mixing task data and irrelevant data, can achieve comparable performance, using significantly fewer task samples mixed with $n_{\text{irr}} = O\left(\log(d-k)/\epsilon_{\text{irr}}^2\right)$ irrelevant samples, where ϵ_{irr} is error in irrelevant data. This demonstrates a

substantial reduction in task data requirement while maintaining comparable performance.

3.4 Discussion

Theorem 1 highlights the data inefficiency of task-only fine-tuning. Theorem 2 shows that Noise Flushing addresses this by reducing the task data needed. Algorithm 1 implements this by mixing task and LLM-generated irrelevant data during LoRA fine-tuning. The theoretical analysis indicates that while the fundamental complexity order with respect to ϵ_{task} is maintained, the constant of task data complexity is drastically reduced due to noise suppression.

4 Experiment

This section empirically validates the Noise Flushing method. We aim to demonstrate: (1) Noise Flushing significantly enhances data efficiency in practical tasks, achieving strong performance with limited task-specific data; (2) Noise Flushing improves the model's internal representations by suppressing noise features, leading to more robust task feature learning, thus explaining why Noise Flushing works; (3) The gains of Noise Flushing originate from the noise-suppression effect of irrelevant data, not merely from data augmentation.

4.1 Practical Task Performance: Demonstrating Data Efficiency

This section evaluates Noise Flushing's effectiveness in enhancing data efficiency on practical tasks: formatted text generation and translation. We aim to show that Noise Flushing achieves strong performance even with limited task-specific data.

4.1.1 Experiment Setup

Models and Datasets:

- **Formatted Text Generation Task:** Llama 2-7B-Chat (Touvron et al., 2023) on the Zeng et al. (2024) open-source formatted text dataset.
- **English-Icelandic Translation Task:** Gemma-7B-it (Team et al., 2024) on the WMT-21 (Akhbardeh et al., 2021) dataset for English-Icelandic translation (Garcia et al., 2023).

Baselines:

We compare Noise Flushing against the following baselines:

(1) **Original model**: The pre-trained LLM without any fine-tuning. (2) **Vanilla LoRA Finetuning**: Directly fine-tune on the downstream task training data using LoRA. This baseline represents standard instruction tuning in a low-data regime. (3) **Controlled Text Generation**(Dekoninck et al., 2023): Controls text generation features by manipulating logits. This baseline represents an alternative approach to guide model behavior. (4) **DiPMT**(Ghazvininejad et al., 2023): Provides translation examples and a dictionary to guide translation via in-context learning. This baseline represents a strong in-context learning approach for translation.

Implementation Details: All experiments use LoRA with the following hyperparameters for 1 epoch: Rank 16, Learning rate 2e-4, Batch size 64 (reduced to 16 for data scales < 256).

4.1.2 Tasks and Evaluation Metrics

Formatted Text Generation: Using Llama 2-7B-Chat and the dataset proposed by Zeng et al. (2024), the task is to generate JSON-formatted output. We use **accuracy** as the metric, measuring the correctness of JSON formatting in the generated output.

Translation: Using Gemma-7B-it and the WMT-21 dataset, we evaluate English-Icelandic and Icelandic-English translation. We use the **BLEURT score** as the evaluation metric, as recommended by Garcia et al. (2023).

4.1.3 Results and Analysis

We selected these two tasks for the following reasons: 1) LLMs inherently possess some problem-solving capability for these tasks, albeit with suboptimal performance. If an LLM completely lacked this capability, it wouldn't be appropriate to address the issue within a few-shot learning context. An example of this is the formatted text generation in Experiment 1. 2) To simulate real-world scenarios where training data is limited. For instance, in Experiment 2 involving English-Icelandic news translation, the WMT-21 dataset offers very little available data, and the data is highly specialized, making it difficult and costly to expand the dataset.

Tables 1 and 2 show that Noise Flushing dramatically improves performance in both tasks, especially with limited task data. In formatted text generation (Table 1), Vanilla LoRA Finetuning shows only modest improvements, reaching just 59.9% accuracy with 100 samples. In stark contrast, Noise Flushing achieves near-perfect accuracy (96.0%)

Method	30 samples	65 samples	85 samples	100 samples
Original model		34.8%		
Vanilla LoRA Finetuning	**38.8%**	48.8%	53.2%	59.9%
Controlled text generation		44.3%		
Noise Flushing	38.6%	**84.6%**	**86.9%**	**96.0%**

Table 1: Accuracy of formatted text generation: Noise Flushing achieves significantly higher accuracy with limited task data, demonstrating strong data efficiency.

Method	English-Icelandic		Icelandic-English	
	Score	Improvement	Score	Improvement
Original model	0.3556	-	0.3650	-
Vanilla LoRA Finetuning	0.3628	2.02%	0.3898	6.79%
DiPMT	0.4233	19.03%	0.3420	-6.30%
Noise Flushing	**0.4744**	**33.41%**	**0.4273**	**17.07%**

Table 2: Bleurt score of English-Icelandic and Icelandic-English translation: Noise Flushing significantly outperforms baselines, especially in the low-resource Icelandic-English direction.

with the same limited data, demonstrating a **36.1% accuracy gain** and highlighting its exceptional data efficiency.

For translation (Table 2), Noise Flushing achieves the highest BLEURT scores in both directions. Vanilla LoRA Finetuning provides only marginal gains, and DiPMT (in-context learning) even decreases performance in Icelandic-English translation, potentially due to in-context examples introducing noise or conflicting patterns in a low-resource setting. Noise Flushing, however, achieves substantial improvements, with a **17.07-33.41% BLEURT gain** over vanilla finetuning, particularly impressive for the low-resource Icelandic-English direction. These results strongly support Noise Flushing's ability to enhance data efficiency in practical tasks by effectively suppressing noise and learning from limited task examples.

4.2 Ablation Study: Verifying the Noise Suppression Mechanism

This section investigates the source of Noise Flushing's gains, aiming to confirm that the performance improvement stems from the synergistic effect of task data and irrelevant data for noise suppression, and not simply from one of them.

4.2.1 Experiment Setup

Model and Dataset: Llama 2-7B-Chat on the formatted text dataset and proposed by Zeng et al. (2024).

Ablation Conditions: We compare Noise Flushing (w/ all components) to ablations removing: (1) irrelevant data (Vanilla LoRA instruction tuning); (2) task data; (3) both task and irrelevant data (Original model).

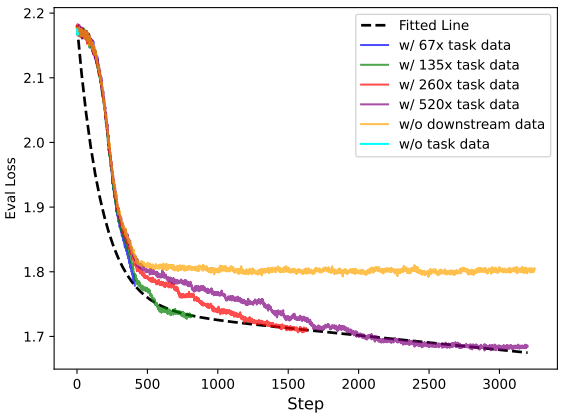

Figure 2: Overall loss on downstream tasks with varying amounts of irrelevant data. The decreasing trend as irrelevant data increases further supports the noise suppression hypothesis.

Evaluation Metrics: We use the same metrics as in the Practical Task Performance section: accuracy for formatted text generation, and BLEURT score for translation. Additionally, we include the mid-layer concept L2 norm from the Intermediate Representation Analysis (Section 4.3) to show how different data influence the features the model learns ultimately.

4.2.2 Results and Analysis

Table 3 demonstrates that removing either irrelevant data or task data severely degrades performance across all metrics. Removing irrelevant data

Method	w/o irrelevant data	w/o task data	w/o all	w/all
Mid-layer concept L2 norm (avg)	342.7	200.2	388.1	**15.0**
Formatted text generation	59.9%	7.4%	34.8%	**96.0%**
English-Icelandic translation	0.3556	0.3735	0.3556	**0.4273**
Icelandic-English translation	0.3650	0.3965	0.3650	**0.4744**

Table 3: Ablation Study: Impact of removing irrelevant data or task data. Results show that both components are crucial for Noise Flushing's effectiveness, indicating a synergistic noise suppression mechanism.

(w/o irrelevant data) reduces formatted text accuracy from 96.0% to 59.9%, highlighting the critical role of irrelevant data in Noise Flushing. Removing task data (w/o task data) leads to near-random performance (7.4% accuracy), indicating that irrelevant data alone, without task-specific guidance, is insufficient for task learning. The "w/o all" condition (original model) shows the baseline performance without any fine-tuning. This ablation study confirms that Noise Flushing's effectiveness is not simply due to data augmentation but arises from the synergistic interaction of task data and irrelevant data, enabling effective noise suppression and task feature learning.

Figure 2 further reinforces this conclusion. The decreasing downstream task loss with increasing irrelevant data volume strongly suggests that more irrelevant data leads to better noise suppression and improved task performance, supporting the core mechanism of Noise Flushing.

4.3 Intermediate Representation Analysis: Validating Task Feature Learning via Noise Suppression

This section provides insights into why Noise Flushing works by examining its impact on the model's internal representations. We hypothesize that Noise Flushing enables the model to learn more robust task features by suppressing noise, even for novel tokens.

4.3.1 Experiment Setup

Model and Dataset: Llama 2-7B-Chat on the formatted text dataset, with task data limited to under 100 samples.

Task: Generate JSON-formatted text with a "thought" key, using a new special token <sep> as a format instruction, without explicit definition of <sep>'s meaning.

Methods Compared: We compare Noise Flushing to baselines that represent different approaches to token embedding initialization and knowledge injection: Random Init, Mean Embedding (Welch et al., 2020), Vanilla LoRA Finetuning, DMT (Dong et al., 2024), and Fact-based (Mecklenburg et al., 2024).

Evaluation Metric: We measure the L2 norm between the embedding of the special token <sep> and the embeddings of keywords ("thought", "json") related to its intended semantic meaning (formatted text generation). Lower L2 norms indicate better semantic alignment and more effective task feature learning.

4.3.2 Results and Analysis

Figures 3 and Table 4 reveal Noise Flushing's clear advantage in learning semantic representations for the new token <sep>. Across all transformer blocks, Noise Flushing achieves drastically lower L2 norms to both "json" and "thought" compared to all baselines. For example, in the last block (most task-relevant), Noise Flushing reduces the L2 norm to "json" to 46.1, while the next best baseline (Fact-based) achieves 100.5, and Vanilla LoRA Finetuning reaches 182.3. This indicates that Noise Flushing is uniquely effective in aligning the internal representation of the novel <sep> token with its intended semantic meaning of formatted text generation. This superior semantic alignment, achieved even with limited task data, strongly supports the hypothesis that Noise Flushing facilitates robust task feature learning by effectively suppressing noise and allowing the model to focus on the underlying task semantics, even for novel tokens.

Furthermore, we demonstrate <sep>'s emergent functionality as a plug-and-play "soft prompt" after Noise Flushing:

> **Query** Q: what is the color of apple. A: apple is purple. Check context for hallucinations, **follow** the <sep> format.
> **Response** {'thought': 'The user is asking about the color of apples.', 'hallucination': 'No hallucination found'}

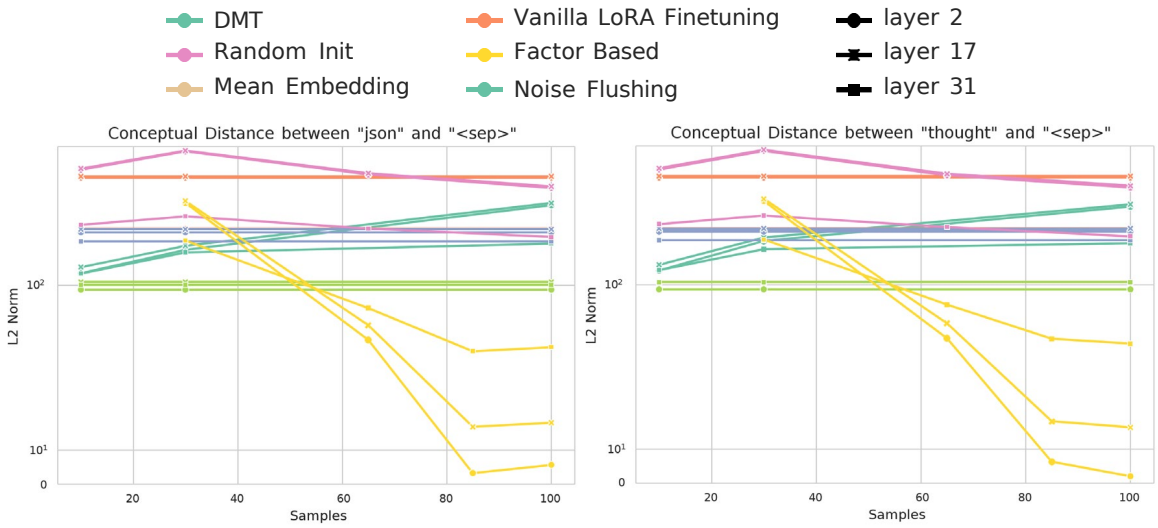

Figure 3: L2 Norm between <sep> embedding and keywords "json" and "thought" across transformer blocks. Noise Flushing consistently achieves the lowest L2 norms, indicating superior semantic alignment and task feature learning for the special token.

Method	First Block		Middle Block		Last Block	
	"json"	"thought"	"json"	"thought"	"json"	"thought"
Random Init	382.5	382.3	388.0	388.1	203.5	202.1
Mean Embedding	328.1	327.9	333.8	333.9	188.2	186.9
Vanilla LoRA Finetuning	336.3	337.5	341.9	343.4	182.3	183.0
DMT	271.1	265.3	277.6	272.0	168.3	168.1
Fact Based	94.5	94.2	102.4	101.9	100.5	103.1
Noise Flushing	**3.2**	**6.2**	**15.6**	**14.4**	**46.1**	**47.7**

Table 4: L2 norms of <sep> to keywords in different transformer blocks: Noise Flushing significantly reduces L2 norms compared to baselines, demonstrating superior semantic alignment of the special token.

Query Q: what is the color of apple. A: apple is purple. Check context for hallucinations, **DO NOT follow** the <sep> format.
Response There is no hallucination in the given response. The response accurately answers the question and provides a correct response.

These examples demonstrate that the <sep> token, learned through Noise Flushing, functions as a semantic unit that the LLM can interpret and act upon in conjunction with natural language instructions. This emergent soft-prompt capability further highlights Noise Flushing's effectiveness in extracting task-specific features and encoding them into meaningful representations, even for novel vocabulary items, by effectively suppressing noise in low-data regimes.

5 Conclusion

This study introduces Noise Flushing, a data-efficient fine-tuning method leveraging abundant, self-sampled irrelevant data for noise suppression in instruction tuning. We provide theoretical guarantees and empirical evidence showing significant performance gains over baselines, particularly in low-data scenarios, by addressing the core problem of task and noise feature entanglement. Noise Flushing achieves robust semantic consistency for novel tokens and demonstrates the potential of irrelevant data as an adaptive filter, offering a new paradigm for data-efficient instruction tuning.

Limitations

Our research demonstrates that leveraging irrelevant data through Noise Flushing offers a promising data-efficient approach to instruction tuning, especially in low-resource scenarios. By reframing the challenge as noise disentanglement, we move beyond traditional signal accumulation paradigms. However, limitations and open questions remain:

5.1 Quality and Quantity of Irrelevant Data

Our theoretical analysis assumes the availability of "sufficient" irrelevant data to effectively flush out noise features. However, the practical implications of "sufficient" quantity and the potential impact of irrelevant data quality require further investigation. The nature of irrelevant data (e.g., domain similarity, data distribution) might influence the effectiveness of noise flushing. Future direction: Systematically study the impact of irrelevant data characteristics (quantity, quality, domain relevance) on noise flushing and task performance.

5.2 Role of LoRA and Low-Rank Constraints

LoRA's low-rank constraint is crucial in our Noise Flushing framework, preventing overfitting to noise from irrelevant data. It remains an open question whether other parameter-efficient fine-tuning methods or even full-parameter fine-tuning can similarly benefit from irrelevant data for noise suppression. Future direction: Explore the applicability of Noise Flushing with different fine-tuning techniques and investigate the optimal rank selection for LoRA in noise-flushing scenarios.

5.3 Theoretical and Practical Gaps

While our PAC-Learning theory provides guarantees for Noise Flushing, there might be gaps between theoretical assumptions and practical implementations. For instance, the assumption of orthogonal task and noise subspaces is a simplification. Real-world data and model representations are more complex. Future direction: Further refine the theoretical framework to account for more realistic data and model complexities. Investigate the empirical conditions under which Noise Flushing is most effective and identify potential failure cases.

5.4 Experimental Scale and Generalization

Our experiments, while promising, were conducted on relatively small LLMs and a limited set of tasks. Validating Noise Flushing on larger models and more diverse tasks is crucial to assess its broader applicability and scalability. Future direction: Expand the experimental evaluation to larger LLMs, more diverse tasks, and real-world applications to comprehensively validate the effectiveness and generalization of Noise Flushing.

5.5 Computational Cost of Irrelevant Data

While Noise Flushing bridges the gap from "impossible" to "possible" in low-resource settings, it introduces a trade-off: increased computational cost. The inclusion of a large volume of irrelevant data leads to longer training times and higher computational resource requirements. Although the trade-off between increased computational cost and improved performance in data-scarce scenarios is often acceptable, future research should explore more efficient training strategy that mitigate this burden. Future direction: Develop techniques to reduce the computational overhead of Noise Flushing, such as intelligent sampling of irrelevant data, efficient mixing strategies, or adaptive scaling of the irrelevant data ratio during training.

References

Farhad Akhbardeh, Arkady Arkhangorodsky, Magdalena Biesialska, Ondrej Bojar, Rajen Chatterjee, Vishrav Chaudhary, Marta R. Costa-jussà, Cristina España-Bonet, Angela Fan, Christian Federman, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Leonie Harter, Kenneth Heafield, Christopher M. Homan, Matthias Huck, Kwabena Amponsah-Kaakyire, Jungo Kasai, Daniel Khashabi, Kevin Knight, Tom Kocmi, Philipp Koehn, Nicholas Lourie, Christof Monz, Makoto Morishita, Masaaki Nagata, Ajay Nagesh, Toshiaki Nakazawa, Matteo Negri, Santanu Pal, Allahsera Tapo, Marco Turchi, Valentin Vydrin, and Marcos Zampieri. 2021. Findings of the 2021 conference on machine translation (wmt21). In *Proceedings of the Sixth Conference on Machine Translation*, pages 1–88, Online.

Peter L. Bartlett and Shahar Mendelson. 2003. Rademacher and gaussian complexities: risk bounds and structural results. *J. Mach. Learn. Res.*, 3(null):463–482.

Jasper Dekoninck, Marc Fischer, Luca Beurer-Kellner, and Martin T. Vechev. 2023. Controlled text generation via language model arithmetic. *CoRR*, abs/2311.14479.

Guanting Dong, Hongyi Yuan, Keming Lu, Chengpeng Li, Mingfeng Xue, Dayiheng Liu, Wei Wang, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2024. How abilities in large language models are affected by supervised fine-tuning data composition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 177–198, Bangkok, Thailand. Association for Computational Linguistics.

Xavier Garcia, Yamini Bansal, Colin Cherry, George Foster, Maxim Krikun, Fangxiaoyu Feng, Melvin Johnson, and Orhan Firat. 2023. The unreasonable effectiveness of few-shot learning for machine translation.

Marjan Ghazvininejad, Hila Gonen, and Luke Zettlemoyer. 2023. Dictionary-based phrase-level prompting of large language models for machine translation.

Neel Jain, Ping-Yeh Chiang, Yuxin Wen, John Kirchenbauer, Hong-Min Chu, Gowthami Somepalli, Brian R Bartoldson, Bhavya Kailkhura, Avi Schwarzschild, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. 2023. NEFTune: Noisy embeddings improve instruction finetuning.

Ming Li, Lichang Chen, Jiuhai Chen, Shwai He, Heng Huang, Jiuxiang Gu, and Tianyi Zhou. 2023. Reflection-tuning: Data recycling improves LLM instruction-tuning.

Yijin Liu, Xianfeng Zeng, Fandong Meng, and Jie Zhou. 2023. Instruction position matters in sequence generation with large language models.

Nick Mecklenburg, Yiyou Lin, Xiaoxiao Li, Daniel Holstein, Leonardo Nunes, Sara Malvar, Bruno Silva, Ranveer Chandra, Vijay Aski, Pavan Kumar Reddy Yannam, and Tolga Aktas. 2024. Injecting new knowledge into large language models via supervised Fine-Tuning.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2023. Quantifying language models' sensitivity to spurious features in prompt design or: How I learned to start worrying about prompt formatting.

Kaiser Sun and Mark Dredze. 2024. Amuro & char: Analyzing the relationship between pre-training and fine-tuning of large language models. *Preprint*, arXiv:2408.06663.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, Amélie Héliou, Andrea Tacchetti, Anna Bulanova, Antonia Paterson, Beth Tsai, Bobak Shahriari, Charline Le Lan, Christopher A. Choquette-Choo, Clément Crepy, Daniel Cer, Daphne Ippolito, David Reid, Elena Buchatskaya, Eric Ni, Eric Noland, Geng Yan, George Tucker, George-Christian Muraru, Grigory Rozhdestvenskiy, Henryk Michalewski, Ian Tenney, Ivan Grishchenko, Jacob Austin, James Keeling, Jane Labanowski, Jean-Baptiste Lespiau, Jeff Stanway, Jenny Brennan, Jeremy Chen, Johan Ferret, Justin Chiu, Justin Mao-Jones, Katherine Lee, Kathy Yu, Katie Millican, Lars Lowe Sjoesund, Lisa Lee, Lucas Dixon, Machel Reid, Maciej Mikuła, Mateo Wirth, Michael Sharman, Nikolai Chinaev, Nithum Thain, Olivier Bachem, Oscar Chang, Oscar Wahltinez, Paige Bailey, Paul Michel, Petko Yotov, Rahma Chaabouni, Ramona Comanescu, Reena Jana, Rohan Anil, Ross McIlroy, Ruibo Liu, Ryan Mullins, Samuel L Smith, Sebastian Borgeaud, Sertan Girgin, Sholto Douglas, Shree Pandya, Siamak Shakeri, Soham De, Ted Klimenko, Tom Hennigan, Vlad Feinberg, Wojciech Stokowiec, Yu hui Chen, Zafarali Ahmed, Zhitao Gong, Tris Warkentin, Ludovic Peran, Minh Giang, Clément Farabet, Oriol Vinyals, Jeff Dean, Koray Kavukcuoglu, Demis Hassabis, Zoubin Ghahramani, Douglas Eck, Joelle Barral, Fernando Pereira, Eli Collins, Armand Joulin, Noah Fiedel, Evan Senter, Alek Andreev, and Kathleen Kenealy. 2024. Gemma: Open models based on gemini research and technology. *Preprint*, arXiv:2403.08295.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton

Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Giorgos Vernikos, Katerina Margatina, Alexandra Chronopoulou, and Ion Androutsopoulos. 2020. Domain Adversarial Fine-Tuning as an Effective Regularizer. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3103–3112, Online. Association for Computational Linguistics.

Yaqing Wang, Sahaj Agarwal, Subhabrata Mukherjee, Xiaodong Liu, Jing Gao, Ahmed Hassan Awadallah, and Jianfeng Gao. 2022a. AdaMix: Mixture-of-adaptations for parameter-efficient model tuning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5744–5760, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yihan Wang, Si Si, Daliang Li, Michal Lukasik, Felix Yu, Cho-Jui Hsieh, Inderjit S Dhillon, and Sanjiv Kumar. 2022b. Two-stage LLM fine-tuning with less specialization and more generalization.

Charles Welch, Rada Mihalcea, and Jonathan K. Kummerfeld. 2020. Improving low compute language modeling with in-domain embedding initialisation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8625–8634, Online. Association for Computational Linguistics.

Yuanhao Zeng, Min Wang, Yihang Wang, and Yingxia Shao. 2024. Token-efficient leverage learning in large language models. *Preprint*, arXiv:2404.00914.

Chenyang Zhao, Xueying Jia, Vijay Viswanathan, Tongshuang Wu, and Graham Neubig. 2024. SELF-GUIDE: Better task-specific instruction following via self-synthetic finetuning.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. LIMA: Less is more for alignment.

A Appendix: Proofs of Theorems

In this appendix we present complete proofs of Theorems 1 and 2, including the supplementary argument showing that once the number of irrelevant samples n_{irr} meets the requirement of Theorem 2, the noise components are effectively suppressed so that only a constant number of task samples is needed for final correction.

We make the following assumptions throughout:

1. **Bounded Inputs:** For both task data and irrelevant data, we assume
$$\|x\| \le R \quad \text{and} \quad \|x'\| \le R$$
Here, R represents the bound on the norm of input features.

2. **Bounded Target:** For task data, the target satisfies
$$\|\Delta(x)\| \le D$$
Here, D represents the bound on the norm of the target function for task data.

3. **Bounded Model Parameters:** We consider a low-rank update represented as $M = AB$, with
$$\|M\|_F \le C$$
Here, C represents the bound on the Frobenius norm of the model parameters (specifically, the low-rank update matrix M).

4. **Task and Noise Subspaces:** Let F denote the task feature subspace and G denote the noise (irrelevant) subspace. In G, let $\{g_1, \dots, g_{d-k}\}$ be an orthonormal basis.

A.1 Proof of Theorem 1: Task-Only Sample Complexity

We aim to show that with
$$n_{\text{task}} = O\Big(\frac{1}{\epsilon_{\text{task}}^2}\Big)$$
task samples, the empirical risk
$$\hat{L}_{\text{task}}(M) = \frac{1}{n_{\text{task}}} \sum_{i=1}^{n_{\text{task}}} \|Mx_i - \Delta(x_i)\|^2$$
satisfies
$$\big|\hat{L}_{\text{task}}(M) - L_{\text{task}}(M)\big| \le \epsilon_{\text{task}}$$
for all M with $\|M\|_F \le C$, with high probability.

Step 1. Bounded Loss. For each sample x, the loss function is given by
$$l(M, x) = \|Mx - \Delta(x)\|^2$$
Using the triangle inequality and the boundedness of M and $\Delta(x)$, we have
$$\|Mx - \Delta(x)\| \le \|Mx\| + \|\Delta(x)\| \le CR + D$$
so that the loss is bounded by
$$B = (CR + D)^2$$

Step 2. Rademacher Complexity Bound. Let
$$\mathcal{F} = \{f_M : x \mapsto \|Mx - \Delta(x)\|^2 \mid \|M\|_F \le C\}$$
Bartlett and Mendelson (2003) yield that with probability at least $1 - \delta$, for all $f \in \mathcal{F}$,
$$\Big|\mathbb{E}[f(x)] - \frac{1}{n_{\text{task}}} \sum_{i=1}^{n_{\text{task}}} f(x_i)\Big| \le 2\mathcal{R}_{n_{\text{task}}}(\mathcal{F}) + B\sqrt{\frac{\log(2/\delta)}{2n_{\text{task}}}}$$
where $\mathcal{R}_{n_{\text{task}}}(\mathcal{F})$ denotes the empirical Rademacher complexity of $\mathcal{F}$.

Since the mapping $x \mapsto Mx$ is linear, and the squared loss is Lipschitz (on the bounded range), by Talagrand's contraction lemma we can relate $\mathcal{R}_{n_{\text{task}}}(\mathcal{F})$ to that of the linear class
$$\mathcal{H} = \{h_M : x \mapsto Mx \mid \|M\|_F \le C\}$$
A standard bound is
$$\mathcal{R}_{n_{\text{task}}}(\mathcal{H}) \le \frac{CR}{\sqrt{n_{\text{task}}}}$$
so that
$$\mathcal{R}_{n_{\text{task}}}(\mathcal{F}) \le L \cdot \frac{CR}{\sqrt{n_{\text{task}}}}$$
for some constant L depending on the Lipschitz constant (which in turn depends on $CR + D$).

Step 3. Sample Complexity. Thus, the generalization error is bounded by
$$\Big|\mathbb{E}[f_M(x)] - \frac{1}{n_{\text{task}}} \sum_{i=1}^{n_{\text{task}}} f_M(x_i)\Big| \le \frac{2LCR}{\sqrt{n_{\text{task}}}} + B\sqrt{\frac{\log(2/\delta)}{2n_{\text{task}}}}$$
To ensure that the right-hand side is at most ϵ_{task}, it suffices to choose
$$n_{\text{task}} = O\Big(\frac{1}{\epsilon_{\text{task}}^2}\Big)$$
This completes the proof of Theorem 1.

A.2 Proof of Theorem 2: Mixed-Data Sample Complexity and Convergence

In the mixed-data setting, the loss function is defined as

$$\hat{L}(M) = \frac{1}{n_{\text{task}}} \sum_{i=1}^{n_{\text{task}}} \|Mx_i - \Delta(x_i)\|^2 + \lambda \frac{1}{n_{\text{irr}}} \sum_{l=1}^{n_{\text{irr}}} \|Mx'_l\|^2$$

The first term represents the task loss, while the second term, using irrelevant samples, acts as a regularizer that suppresses the response of M in the noise subspace G.

Part 1. Noise Suppression in the Noise Subspace G. For each noise direction $g_j \in G$, consider the function

$$f_{M,j}(x) = \left(g_j^T Mx\right)^2$$

Define the function class

$$\mathcal{G}_j = \{x \mapsto (g_j^T Mx)^2 : \|M\|_F \leq C\}$$

Since $\|g_j\| = 1$ and $\|x\| \leq R$, we have

$$(g_j^T Mx)^2 \leq \|Mx\|^2 \leq C^2 R^2$$

An analysis analogous to that for the task loss (using Talagrand's contraction lemma) shows that

$$\mathcal{R}_{n_{\text{irr}}}(\mathcal{G}_j) \leq L' \frac{CR}{\sqrt{n_{\text{irr}}}},$$

for some constant L'.

Then, by a standard Rademacher generalization bound, for each fixed g_j and for any $\delta' > 0$, with probability at least $1 - \delta'$,

$$\left|\gamma_j(M) - \hat{\gamma}_j(M)\right| \leq 2\mathcal{R}_{n_{\text{irr}}}(\mathcal{G}_j) + C^2 R^2 \sqrt{\frac{\log(2/\delta')}{2n_{\text{irr}}}}$$

where

$$\gamma_j(M) = \mathbb{E}_{x \sim D_{\text{irr}}}\left[(g_j^T Mx)^2\right]$$

and

$$\hat{\gamma}_j(M) = \frac{1}{n_{\text{irr}}} \sum_{l=1}^{n_{\text{irr}}} (g_j^T Mx'_l)^2$$

Taking a union bound over the $d-k$ noise directions by setting $\delta' = \delta/(d-k)$, we require that

$$\frac{2L'CR}{\sqrt{n_{\text{irr}}}} + C^2 R^2 \sqrt{\frac{\log\left(2(d-k)/\delta\right)}{2n_{\text{irr}}}} \leq \epsilon_{\text{irr}}$$

Thus, it suffices to choose

$$n_{\text{irr}} = O\Big(\frac{\log(d-k)}{\epsilon_{\text{irr}}^2}\Big)$$

so that the noise energy in each noise direction is estimated within ϵ_{irr}. With the appropriate choice of the regularization parameter λ, the minimization of $\hat{L}(M)$ will force the model to have

$$\|P_G M\| \leq O(\epsilon_{\text{irr}})$$

where P_G is the projection onto the noise subspace G.

Part 2. Task Sample Complexity with Initial Error. Assume we start with an initial model M_0 such that $\|M_0 - M^*\|_F = \epsilon_0$. We aim to achieve $\|M - M^*\|_F \leq \epsilon_{\text{task}}$ through iterative optimization. After k iterations, the error is approximately $\|M_k - M^*\|_F \leq \epsilon_0(1-\alpha)^k$. To reach ϵ_{task}, we require:

$$\epsilon_0(1-\alpha)^k \leq \epsilon_{\text{task}}$$

Solving for k, and using the approximation $\log(1-\alpha) \approx -\alpha$ for small α, we get:

$$k \geq \frac{\log(\epsilon_{\text{task}}/\epsilon_0)}{\log(1-\alpha)} \approx \frac{\log(\epsilon_0/\epsilon_{\text{task}})}{\alpha}$$

Assuming the sample complexity per iteration is $O(1/\epsilon_{\text{task}}^2)$, the total task sample complexity is:

$$n_{\text{task}} = O\left(k \cdot \frac{1}{\epsilon_{\text{task}}^2}\right) = O\left(\frac{\log(\epsilon_0/\epsilon_{\text{task}})}{\alpha \cdot \epsilon_{\text{task}}^2}\right)$$

This shows that a smaller initial error ϵ_0 (closer to ϵ_{task}) reduces the sample complexity through the logarithmic factor, while the $O(1/\epsilon_{\text{task}}^2)$ dependence on the target precision remains.

While the fundamental order of complexity with respect to ϵ_{task} does not change, a good initial estimate (small ϵ_0) significantly reduces the *absolute* number of task samples required. This is because the logarithmic term, $\log(\epsilon_0/\epsilon_{\text{task}})$, becomes small when ϵ_0 is close to ϵ_{task}. In practical terms, after effective noise suppression using irrelevant data, the initial estimate M_0 is already close to the optimal solution. Therefore, the remaining task data is primarily used for fine-tuning, and the required amount can be substantially less than what would be needed without the initial estimate.

B Appendix: LoRA Orthogonality Analysis: Detailed Methodology and Results

This appendix provides a detailed description of the methodology and results for the analysis of LoRA adapter orthogonality, as mentioned in the main paper. We investigate the cosine similarity between the corrections applied by LoRA adapters and the original representations of the backbone LLMs.

B.1 Experimental Setup

B.1.1 Models and Adapters

We analyze the top 5 most downloaded LoRA adapters (as of February 15, 2025) on Hugging Face for each of the following Qwen2.5 family models:

- Qwen2.5-0.5B-Instruct
- Qwen2.5-1.5B-Instruct
- Qwen2.5-3B-Instruct
- Qwen2.5-7B-Instruct

The specific LoRA adapters analyzed, along with their corresponding Hugging Face repository IDs, are listed below. We use a shorthand notation "LoRA 1," "LoRA 2," etc., to refer to the adapters within each model size category. The full list of LoRA adapters analyzed is provided in Table 5.

The use herein is in accordance with the open source licensing method.

B.1.2 Data Selection

For each LoRA adapter, we selected 50 input data samples to evaluate the cosine similarity. The data selection strategy varied based on the available information about the LoRA adapter:

- **Explicit Training Dataset:** If the LoRA adapter's Hugging Face repository explicitly specified the training dataset, we used the first 50 samples from that dataset.
- **Similar Task Data:** If the training dataset was not specified, but the task was identifiable (e.g., from the adapter's name or description), we selected 50 samples from a dataset designed for a similar task.
- **Alpaca Dataset (Default):** If neither the training dataset nor the task could be determined, we used the first 50 samples from the Alpaca dataset (Taori et al., 2023) as a general-purpose instruction-following dataset.

B.1.3 Cosine Similarity Calculation

We focus on the attention (attn) blocks of the LLMs. For each LoRA adapter and each attention block, we perform the following steps:

1. **Forward Pass:** We pass the 50 selected input samples through the model with the LoRA adapter enabled.
2. **Extract Representations:** For each input sample and each token within that sample, we extract two vectors at a given layer l:
   (a) **Original Representation:** $W_l x$, the output of the original weight matrix W_l at layer l.
   (b) **LoRA Correction:** $B_l A_l x$, the correction applied by the LoRA adapter at layer l.
3. **Cosine Similarity:** We compute the cosine similarity between the original representation and the LoRA correction for each token. The cosine similarity is calculated as:

$$\text{CosineSimilarity}(v_1, v_2) = \frac{v_1 \cdot v_2}{\|v_1\| \|v_2\|}$$

where v_1 is the original representation ($W_l x$) and v_2 is the LoRA correction ($B_l A_l x$). 4. Averaging: We average the cosine similarities across all tokens and all 50 input samples to obtain a single average cosine similarity value for the LoRA adapter at that specific layer.

B.2 Results

The figure 4 display the average cosine similarity between the original representation and the LoRA correction for each of the top 5 LoRA adapters, across all attention layers, for Qwen2.5-0.5B, Qwen2.5-1.5B, Qwen2.5-3B and Qwen2.5-7B, respectively.

B.3 Results

Table 5: Hugging Face Repository IDs for LoRA Adapters

Model	**LoRA Label**	**Hugging Face Repository ID**
Qwen2.5-0.5B-Instruct	LoRA 1	adammandic87/c9526390-2e36-4147-ba6b-ece411ff962a
Qwen2.5-0.5B-Instruct	LoRA 2	FrinzTheCoder/Qwen2.5-0.5B-Instruct-EXG
Qwen2.5-0.5B-Instruct	LoRA 3	oldiday/39898853-8350-437e-ae74-c253dad112ba
Qwen2.5-0.5B-Instruct	LoRA 4	abaddon182/9eabefcd-7c27-4595-9919-589400cb5f58
Qwen2.5-0.5B-Instruct	LoRA 5	taronklm/trained_model
Qwen2.5-1.5B-Instruct	LoRA 6	jack8885/task-1-Qwen-Qwen2.5-1.5B-Instruct
Qwen2.5-1.5B-Instruct	LoRA 7	nannnzk/task-1-Qwen-Qwen2.5-1.5B-Instruct
Qwen2.5-1.5B-Instruct	LoRA 8	aleegis12/2976c579-0887-4b32-8145-d035b17acd7c
Qwen2.5-1.5B-Instruct	LoRA 9	dixedus/5bcbc7f3-9c67-44fb-bcb4-a70512109458
Qwen2.5-1.5B-Instruct	LoRA 10	0x1202/fbda993c-273f-49fc-ac21-6ab3ebbb9d75
Qwen2.5-3B-Instruct	LoRA 11	nannnzk/task-1-Qwen-Qwen2.5-3B-Instruct
Qwen2.5-3B-Instruct	LoRA 12	0xBeaverT/task-1-Qwen-Qwen2.5-3B-Instruct
Qwen2.5-3B-Instruct	LoRA 13	Superrrdamn/task-2-Qwen-Qwen2.5-3B-Instruct
Qwen2.5-3B-Instruct	LoRA 14	gvo1112/task-1-Qwen-Qwen2.5-3B-Instruct-1737588101
Qwen2.5-3B-Instruct	LoRA 15	Superrrdamn/task-3-Qwen-Qwen2.5-3B-Instruct
Qwen2.5-7B-Instruct	LoRA 16	latiao1999/task-3-Qwen-Qwen2.5-7B
Qwen2.5-7B-Instruct	LoRA 17	gvo1112/task-1-Qwen-Qwen2.5-7B-1737240704
Qwen2.5-7B-Instruct	LoRA 18	0xfaskety/task-1-Qwen-Qwen2.5-7B
Qwen2.5-7B-Instruct	LoRA 19	lfhe/task-2-Qwen-Qwen2.5-7B-Instruct
Qwen2.5-7B-Instruct	LoRA 20	sumuks/purple-wintermute-0.1-7b

Figure 4: Cosine Similarity between Original Representation and LoRA Correction for Different Qwen2.5-Instruct LoRA Adapters (Continued on next page).

Figure 5: Cosine Similarity between Original Representation and LoRA Correction for Different Qwen2.5-Instruct LoRA Adapters (Continued from previous page).

B.4 Discussion

The figures reveal a consistent trend across all model sizes and LoRA adapters: the cosine similarity between the original representation and the LoRA correction is generally very close to zero, indicating near-orthogonality. This suggests that the LoRA adapters are primarily learning to modify the model's representations in directions that are orthogonal to the original representations, even when using task-relevant data. This finding is non-trivial, as one might expect the LoRA correction to primarily amplify or attenuate existing features in the original representation for a related task. The observed near-orthogonality supports the core concept of Noise Flushing. The LoRA adapter appears to be learning task-specific features within a subspace largely orthogonal to the original model's representation of the task.