Data-Efficient LLM Fine-Tuning by Noise Resistance

Anonymous ACL submission

Abstract

Fine-tuning large language models (LLMs) remains challenging due to the scarcity of downstream task data and the prevalence of noisy supervision. In this paper, we introduce Noise Flushing (NF), a novel paradigm that prioritizes noise elimination over data augmentation. NF leverages abundant irrelevant data—sampled from the base LLM—to mitigate noise and sharpen focus on task-relevant signals during fine-tuning, thus enabling effective adaptation in extremely low-resource settings. Theoretically, we show that NF can match or even surpass the performance of standard LoRA finetuning settings, despite using substantially fewer task-specific examples. Empirically, NF achieves consistent and significant improvements over strong fine-tuning baselines across various tasks, including machine translation, structured text generation, text-to-SQL, and special token understanding—even with fewer than 100 examples.

1 Introduction

Large Language Models (LLMs) have revolutionized numerous applications, yet effectively adapting them to specialized domains often depends on fine-tuning. Instruction tuning—a widely adopted paradigm that trains models on instruction-response pairs (Zhao et al., 2024; Zhou et al., 2023)—faces substantial challenges when task-specific data is scarce. In such settings, both instructions and responses often embed specific facts or biases. We argue that the presence of noise in these sparse datasets further exacerbates the difficulty of effective adaptation.

With only a handful of examples, LLMs are prone to overfitting to noise rather than grasping the underlying task semantics. Traditional approaches—such as data augmentation and synthetic data generation (Li et al., 2023; Zhao et al., 2024)—aim to amplify the weak task signal, but often fall short when the available data is limited and noisy. The fundamental challenge remains: how can models learn effectively when the signal is weak and entangled with noise?

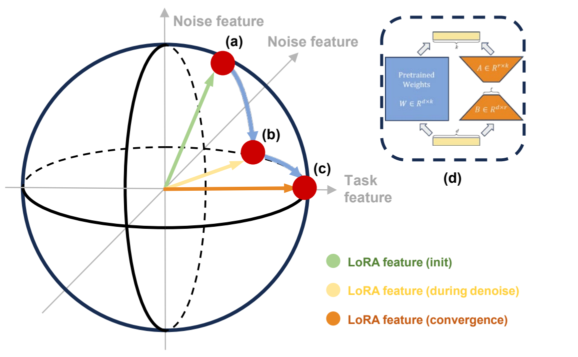

Figure 1: **Overview of Noise Flushing**, where irrelevant data guides LoRA to learn task specific features and mitigate noise. (a) Initial LoRA features (from task data) contain a mix of task features and noise. (b) Loss minimization on self-sampled irrelevant data forces noise suppression. (c) Convergence 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.

Inspired by the intuition that effective learning involves not just amplifying the signal but also actively mitigating noise, we introduce **Noise Flushing (NF)**, a novel framework for data-efficient fine-tuning of LLMs. This approach represents a different paradigm: instead of solely focusing on strengthening the weak task signal in limited data, NF prioritizes the removal of pervasive sample-specific noise. NF uses abundant, irrelevant data sampled from the base LLM to mitigate noise and sharpen focus on task-relevant signals, thus enabling effective adaptation in limited data settings.

Specifically, our analysis and experiments reveal that the LoRA module operates within a subspace orthogonal to the base LLM's foundational representation space. However, noise in the training data can mislead the LLM by activating undesired directions within this foundational space. In NF, the sampled irrelevant data serve as constraints that encourage the LoRA module to suppress responses to irrelevant inputs, ensuring that it becomes selectively active only for task-relevant signals.

Via theoretical analysis, we show that this approach substantially reduces the amount of task-specific data required for effective fine-tuning. Our analysis suggests that by mixing a small number of task-relevant examples with irrelevant data, the model can achieve performance on par with traditional methods, while requiring up to orders of magnitude fewer task samples.

Empirically, we demonstrate the substantial advantages of NF over strong baselines, particularly in extremely low-data regimes. Specifically, NF achieves significantly higher accuracy in formatted text generation and improved BLEURT scores in translation, while preserving robust semantic consistency for special tokens—a notoriously difficult challenge in low-resource fine-tuning.

In summary, the contributions of our work are:

- First, we reframe data-inefficient instruction tuning as a noise-resistance problem, highlighting the entanglement of task semantics and noise as a fundamental bottleneck in low-data learning.
- Second, we propose NF, a novel method that, both theoretically and empirically, demonstrates the effectiveness of combining sparse task data with abundant irrelevant data sampled from the base LLM to enhance task learning by effectively suppressing noise.
- Lastly, we show that NF achieves robust semantic consistency for novel special tokens, overcoming a key limitation of existing knowledge injection techniques in LLMs.

2 Related Work

In this section, we first review existing approaches for fine-tuning LLMs in low-data regimes, followed by a discussion of studies that distinguish between different types of fine-tuning data.

2.1 Fine-tuning under Data Scarcity

In low-data regimes, researchers have explored a variety of approaches to enhance model performance. These include data augmentation to virtually expand training sets (Li et al., 2023; Zhao et al., 2024); meta-learning methods that aim to “learn to learn” from limited examples (Zhu et al., 2024); iterative self-improvement strategies for refining training data quality (Li et al., 2023; Zhao et al., 2024); and the generation of synthetic data for instruction tuning (Liu et al., 2023; Dong et al., 2024; Mecklenburg et al., 2024). Other efforts involve modifying training objectives (Vernikos et al., 2020) or introducing architectural innovations, such as parameter-efficient fine-tuning modules like LoRA (Wang et al., 2022; Hu et al., 2021), and embedding-level noise injection techniques (Jain et al., 2023), all aimed at improving generalization under limited supervision.

NF addresses the challenge of data scarcity from a novel perspective—by mitigating noise and enhancing focus on task-relevant signals through the use of abundant irrelevant data, which is easily obtained by sampling from the base LLM.

2.2 Fine-Tuning with Data Prioritization

Certain strategies, such as Direct Preference Optimization (DPO) (Rafailov et al., 2024) and dataset pruning or distillation (Zhou et al., 2023), operate—either implicitly or explicitly—on the premise that not all data behaviors are equally valuable. DPO, for example, learns from pairs of preferred and dispreferred responses by optimizing for the former. NF aligns with this motivation, emphasizing the importance of task-relevant signals while suppressing noise.

However, the underlying mechanism of NF is fundamentally different. It guides the LoRA module to suppress responses to irrelevant data, thereby forcing it to learn features that are activated only by task-relevant inputs. By leveraging easily generated irrelevant data, NF provides a novel and efficient solution for extremely low-resource scenarios. Importantly, NF remains effective even in the absence of curated preference pairs, as required by DPO, or large initial datasets, as needed for pruning-based methods.

3 Methodology

Unless otherwise specified, bold uppercase letters denote matrices, and bold lowercase letters denote vectors. We use $\|\cdot\|_F$ to denote the Frobenius norm and $\|\cdot\|$ for the ℓ_2 norm.

3.1 Preliminary: Low-Rank Adaptation

In this work, we utilize Low-Rank Adaptation (LoRA), which updates pre-trained weights W with a low-rank matrix $\Delta W = AB$, where

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

```
1:  Initialize fine-tuned model with pre-trained LLM and LoRA
2:  $D_{\text{irr}} \leftarrow$ Sample QA pairs from LLM with $Q_{\text{irr}}$
3:  Prepare dataset by mixing $D_{\text{task}}$ and $D_{\text{irr}}$ with ratio $1 : r$.
4:  for each epoch do
5:      for each batch in shuffled combined dataset do
6:          Train LLM with LoRA on the batch.
7:          Update LoRA parameters.
8:      end for
9:  end for
10: return Fine-tuned LLM
```

$\text{rank}(\Delta W) \ll \text{rank}(W)$. The model response becomes $r(x) = Wx + BAx$.

Observation 1. Approximate Orthogonality between LoRA Activations and original representation . LoRA activations tend to be approximately orthogonal to the original representations from the pre-trained model. This suggests that LoRA operates in a distinct update subspace that is focused on the new task. A detailed empirical analysis is provided in Appendix B.

3.2 Problem Statement: Task and Noise Feature Entanglement

Instruction tuning with severely limited data suffers from the entanglement of task-specific features and sample-specific stochastic noise. We conceptually decompose the input feature space into an (unknown a priori) task feature subspace F and a noise feature subspace G, with corresponding (conceptual) projection operators P_F and P_G. For any input x, we can consider its features to be representable as a combination of components from these subspaces.

The ideal update ΔW^* learned from D_{task} should primarily reside in the task feature subspace F, meaning its projection onto the noise subspace G, i.e., $P_G \Delta W^*$, should be approximately zero. However, with limited task data, standard fine-tuning struggles to achieve this disentanglement because the subspaces F and G are not explicitly known or engineered. Our method, NF, works implicitly: the mixed-data training process guides the LoRA update to primarily capture components in F and ignore or suppress components in G, rather than requiring explicit knowledge or construction of these subspaces.

3.3 Noise Flushing Method

NF utilizes self-sampled irrelevant data D_{irr} alongside scarce task-specific data D_{task} to guide LoRA updates. The complete procedure is presented in Algorithm 1. As shown, the training process follows a standard supervised learning paradigm over mixed batches drawn from both D_{task} and D_{irr}.

Explanation. NF operates on the assumption that the task dataset D_{task} comprises both task-relevant signals—i.e., features residing in a task-specific subspace F—and noise, i.e., features within a noise subspace G, which may stem from spurious correlations common in low-data regimes. In contrast, the irrelevant dataset D_{irr} consists of general inputs for which the task-specific signal components $P_F x$ are negligible, i.e., $P_F x \approx 0$ for all $x \in D_{\text{irr}}$. From an optimization perspective, the key distinction between NF and standard LoRA lies in how the LoRA adapter ΔW learns from irrelevant data D_{irr}. For an input $x \in D_{\text{irr}}$, the target output y is generated by the base LLM itself, i.e., $y \approx Wx$. Minimizing the supervised loss $\mathcal{L}((W + \Delta W)x, y)$ on such examples implicitly encourages the LoRA-induced change to be negligible, i.e., $\Delta W x \approx 0$.

Specifically, the suppression of $\Delta W x$ on irrelevant data $x \in D_{irr}$ encourages consistency with the model's pre-trained knowledge and behavior on general inputs. In low-data scenarios—where ΔW might otherwise overfit to noise in D_{task} by learning a spurious component ΔW_G—this consistency constraint acts as a regularizer, suppressing undesirable updates. As a result, the limited capacity of the LoRA adapter is guided to focus on learning genuine task-relevant features ΔW_F from D_{task}. By mixing D_{task} with irrelevant data D_{irr}, NF effectively steers LoRA towards selective adaptation: amplifying task-specific signals while mitigating the impact of noise. Details on the construction of D_{irr} can be found in Appendix C. The practical effectiveness of this mechanism is supported by our empirical results.

3.4 Theoretical Analysis of Data Efficiency with Noise Flushing

Building on the intuitive understanding of NF's mechanism, we now provide a theoretical analysis of its potential to reduce the need for task-specific training data. Given the complexity of LLMs, this formalization necessarily relies on strong simplifying assumptions. Key assumptions include bounded inputs $\|x\| \leq R$, bounded representation output $\|\Delta W^*(x)\| \leq D$ for task data, bounded LoRA updates $\|\Delta W\|_F \leq C$, where R, D, C are fixed constants, and the conceptual existence of orthogonal task feature subspace F and noise feature subspace G. Details are provided in Appendix A.

Before presenting our sample complexity analysis, we first introduce two types of errors—commonly referred to as risks in the framework of Probably Approximately Correct (PAC) learning, i.e., the **empirical risk** and the **true risk**, defined as follows:

- **Empirical risk** $R_{\text{emp}}(f)$ is the average error of a model f evaluated on the training dataset D_{task}.
- **True risk** $R(f)$ is the expected error of the model over the entire data distribution, reflecting its generalization performance.

In PAC learning, the empirical risk serves as an approximation of the true risk, with the quality of this approximation improving as the number of training samples increases.

Theorem 1. Task-Only Sample Complexity: when performing standard LoRA fine-tuning using only task data D_{task}, to guarantee that the empirical risk ϵ_{task} deviates from the true risk with a probability of at least $1 - \delta$, the required number of task samples n_{task} is given by:

$$n_{\text{task}} = O\left(\frac{1}{\epsilon_{\text{task}}^2}\right) \tag{1}$$

This indicates that the required number of task samples is inversely proportional to the square of the target precision.

Theorem 2. Mixed-Data Sample Complexity with NF consists of two parts:

1. **Noise Suppression:** By introducing n_{irr} irrelevant samples, the component of the LoRA activation ΔWx residing in the noise subspace G, denoted $\|P_G \Delta Wx\|_F$, can be effectively suppressed. To achieve a noise suppression accuracy of ϵ_{irr} with probability $1 - \delta$, the number of irrelevant samples n_{irr} required is:

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

   where d and k denote the intrinsic dimension of the ΔWx and F, respectively, thus $d-k$ is the effective dimension of the noise subspace G. ϵ_0 is the initial optimization error for the task.

2. **Task Sample Complexity:** Here ϵ_0 denotes initial error. After the noise components are suppressed, the number of task samples n_{task} required to achieve the final target task error ϵ_{task} is:

$$n_{\text{task}} = O\left(\frac{\log(\epsilon_0/\epsilon_{\text{task}})}{\alpha \cdot \epsilon_{\text{task}}^2}\right) \tag{3}$$

   where α is a factor related to the convergence rate of the optimization process on the task loss.

3.5 Discussion

Theorem 2 suggests that introducing a sufficient number of irrelevant samples n_{irr} effectively suppresses noise, thereby reducing the initial optimization error ϵ_0 for the task loss. As a result, the number of task-specific samples n_{task} required to achieve a target task precision ϵ_{task} is also reduced. Despite relying on simplifying assumptions, this formal analysis supports our earlier intuition and optimization-based explanation in Section 3.3—that NF facilitates more efficient extraction of task-relevant signals from a small set of examples.

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

This section evaluates Noise Flushing's effectiveness in enhancing data efficiency on practical tasks: formatted text generation, translation, and text-to-SQL generation. 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).
- **Text-to-SQL Generation Task:** Qwen1.5-7B-Chat (Team, 2024) on the BIRD-SQL Mini dataset (Li et al., 2024). We use the 'sql-create-context' (b mc2, 2023) subset for scarce task data and also evaluate NF's synergy with 'synthetic_text_sql' (Meyer et al., 2024) data.

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 (for formatted text generation). (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. (5) For Text-to-SQL, we also compare NF with advanced LoRA variants: **DoRA** (Liu et al., 2024) and **AdaLoRA** (Zhang et al., 2023), both with and without NF.

Implementation Details: All experiments use LoRA (or its variants) with the following base 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).

Text-to-SQL Generation: Using Qwen1.5-7B-Chat on BIRD-SQL Mini, the task is to generate SQL queries from natural language questions. We report **Execution Accuracy (EX %)** on the 'Moderate' and 'Challenging' subsets of BIRD-SQL Mini. To simulate low data resource scenarios, we select the first 100 samples.

4.1.3 Results and Analysis

We selected these tasks for the following reasons: 1) LLMs suboptimally possess some problem-solving capability for these tasks. If an LLM completely lacked this capability, it wouldn't be appropriate to address the issue within a few-shot learning context. 2) To simulate real-world scenarios where training data is limited, such as in English-Icelandic news translation (WMT-21) or specialized Text-to-SQL applications.

Noise Flushing significantly boosts performance in formatted text generation (Table 1), translation (Table 2), and Text-to-SQL tasks (Table 3), especially under limited data conditions. In formatted text generation, NF achieves near-perfect accuracy (96.0% with 100 samples), dramatically surpassing vanilla LoRA (59.9%). For translation, NF shows substantial BLEURT gains (17-33%) compared to vanilla fine-tuning. In Text-to-SQL, NF notably improves performance on challenging examples (+200% when combined with Vanilla LoRA) and synergizes with LoRA variants like DoRA and AdaLoRA, as well as synthetic data. These results highlight NF's broad utility in mitigating noise and enhancing performance, particularly in difficult, low-signal conditions.

The findings underscore Noise Flushing's data efficiency in practical tasks through noise suppression and leveraging limited examples. Additional experiments are detailed in the Appendix.

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.

Base Method	Moderate EX (%) (vs. w/o NF)	Challenging EX (%) (vs. w/o NF)
Task Data: sql-create-context (Qwen1.5-7B-Chat)		
Vanilla LoRA w/o NF	**7.60**	0.98
Vanilla LoRA w/ NF	7.20 (-5.3%)	**2.94 (+200.0%)**
DoRA w/o NF	6.80	1.96
DoRA w/ NF	**7.20 (+5.9%)**	**2.94 (+50.0%)**
AdaLoRA w/o NF	6.00	1.96
AdaLoRA w/ NF	**6.80 (+13.3%)**	**4.90 (+150.0%)**
Task Data: synthetic_text_sql (Qwen1.5-7B-Chat)		
Synthetic Data w/o NF	6.40	**3.92**
Synthetic Data w/ NF	**7.60 (+18.8%)**	**3.92 (+0.0%)**

Table 3: Performance of Noise Flushing (NF) on the BIRD-SQL Mini (text2sql) task with Qwen1.5-7B-Chat, showing synergy with advanced LoRA variants and synthetic task data. Percentages in parentheses indicate relative change compared to the corresponding method without NF.

4.2 Ablation Study

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).

Evaluation Metrics: We use the same metrics as in the Section 4.1: 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 4's ablation study shows removing either task or irrelevant data severely degrades performance (e.g., formatted text accuracy from 96.0% to 59.9% without irrelevant, 7.4% without task data). This confirms Noise Flushing's effectiveness stems from the synergistic interaction of both data types for noise suppression and feature learning, not just augmentation.

Figure 2 supports this, showing more irrele-

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 4: 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.

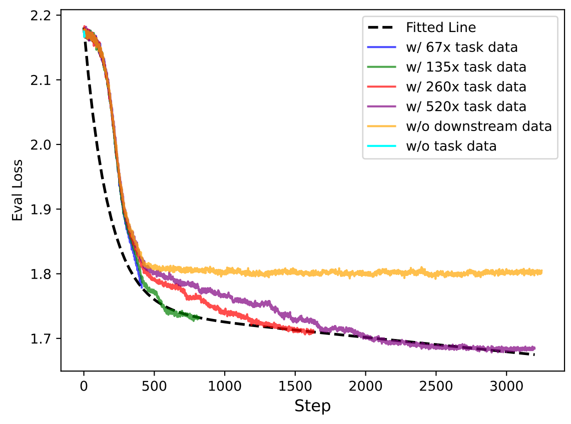

Figure 2: Overall loss on downstream tasks with varying amounts of irrelevant data. The decreasing trend as irrelevant data increases further supports the noise suppression hypothesis.

vant data improves performance by reducing downstream task loss, reinforcing NF's core mechanism.

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 5 show that Noise Flushing excels at learning semantic representations for the new token <sep>. It achieves significantly lower L2 norms to "json" and "thought" across all transformer blocks compared to baselines. For instance, in the final block, Noise Flushing's L2 norm to "json" is 46.1, far surpassing the next best baseline (Fact-based at 100.5) and Vanilla LoRA Finetuning (182.3). This demonstrates Noise Flushing's unique effectiveness in aligning <sep>'s internal representation with its intended meaning for formatted text generation. This strong semantic alignment, even with limited data, supports the idea that Noise Flushing enables robust task feature learning by suppressing noise and focusing the model on underlying task semantics, including for new 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'}

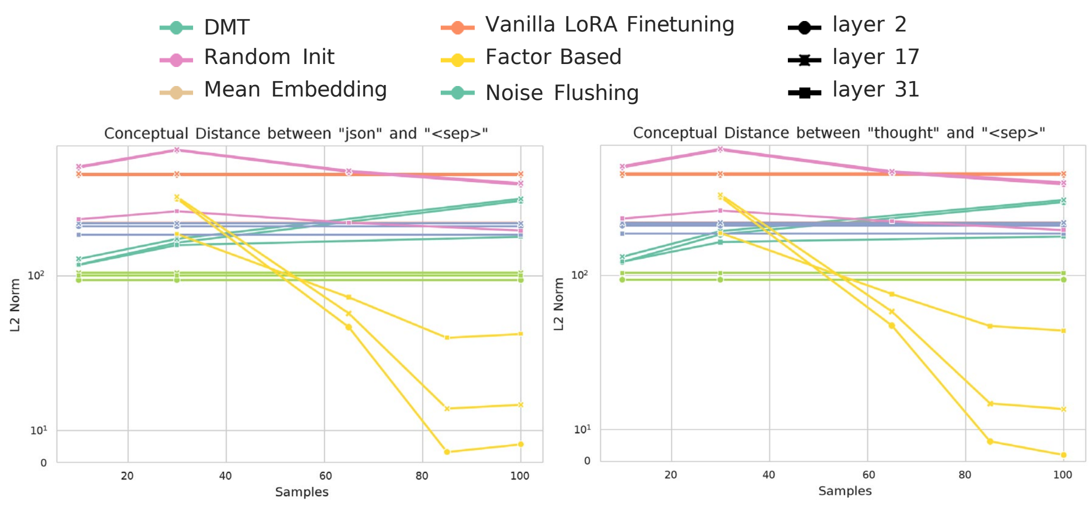

Figure 3: L2 Norm between <sep> embedding and keywords "json" and "thought" across transformer blocks. Noise Flushing consistently achieves the lowest L2 norms, indicating superior semantic alignment and task feature learning for the special token.

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 5: 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 paper introduces Noise Flushing (NF), a novel LLM fine-tuning method prioritizing noise removal in low-data settings. NF uses abundant, self-sampled irrelevant data during LoRA fine-tuning to suppress noise and focus on task-relevant features. Both theoretically and empirically, NF shows significant performance gains with drastically fewer task samples in areas like translation and text-to-SQL, even improving new special token understanding. NF reframes data-efficient tuning as noise disentanglement, enabling LLMs to learn effectively in sparse data environments.

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.

b mc2. 2023. sql-create-context dataset. This dataset was created by modifying data from the following sources: (**??**).

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.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *Preprint*, arXiv:2106.09685.

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.

Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, et al. 2024. Can llm already serve as a database interface? a big bench for large-scale database grounded text-to-sqls. *Advances in Neural Information Processing Systems*, 36.

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

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024. Dora: weight-decomposed low-rank adaptation. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

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.

Yev Meyer, Marjan Emadi, Dhruv Nathawani, Lipika Ramaswamy, Kendrick Boyd, Maarten Van Segbroeck, Matthew Grossman, Piotr Mlocek, and Drew Newberry. 2024. Synthetic-Text-To-SQL: A synthetic dataset for training language models to generate sql queries from natural language prompts.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Preprint*, arXiv:2305.18290.

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.

Qwen Team. 2024. Introducing qwen1.5.

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. 2022. 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.

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.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *Preprint*, arXiv:2303.10512.

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.

Liang Zhu, Feiteng Fang, Yuelin Bai, Longze Chen, Zhexiang Zhang, Minghuan Tan, and Min Yang. 2024. DEFT: Distribution-guided efficient fine-tuning for human alignment. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 15318–15331, Miami, Florida, USA. Association for Computational Linguistics.

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 fewer task samples is needed for final correction.

Here we use the norm minimization of the difference between the activation generated by LoRA and the ideal activation to represent the training objective. This is for the convenience of demonstration, because we are not concerned about the impact of other components of LLM, such as lm_head, on this training dynamic. Only caring about the difference in representation provides a microscopic perspective for analysis

We make the following assumptions throughout:

1. **Bounded Inputs:** For both task data and irrelevant data, we assume

   $$\|x\| \leq R \quad \text{and} \quad \|x'\| \leq R$$

   Here, R represents the bound on the norm of input features.

2. **Bounded Target:** For task data, the target satisfies

   $$\|\Delta(x)\| \leq D$$

   Here, D represents the bound on the norm of the target function for task data. Specifically, $\Delta(x)$ is $\Delta W^*(x)$ in main paper, we use it for convenience.

3. **Bounded Model Parameters:** We consider a low-rank update represented as $M = AB$, with

   $$\|M\|_F \leq C$$

   Here, C represents the bound on the Frobenius norm of the model parameters (specifically, the low-rank update matrix M, which is ΔW in main paper. We use it to convey the idea that LoRA activation values provide a correction to the representation produced by the original weights, and this correction, as our experiments show, has independent underlying meaning and does not lose its significance when detached from W; therefore, we use a new letter).

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

$$\left|\hat{L}_{\text{task}}(M) - L_{\text{task}}(M)\right| \leq \epsilon_{\text{task}}$$

for all M with $\|M\|_F \leq C$, 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

$$\|Mx - \Delta(x)\| \leq \|Mx\| + \|\Delta(x)\| \leq CR + D$$

so that the loss is bounded by

$$B = (CR + D)^2$$

Step 2. Rademacher Complexity Bound. Let

$$\mathcal{F} = \{f_M : x \mapsto \|Mx - \Delta(x)\|^2 \mid \|M\|_F \leq C\}$$

Bartlett and Mendelson (2003) yield that with probability at least $1 - \delta$, for all $f \in \mathcal{F}$,

$$\left|\mathbb{E}[f(x)] - \frac{1}{n_{\text{task}}} \sum_{i=1}^{n_{\text{task}}} f(x_i)\right| \leq 2\mathcal{R}_{n_{\text{task}}}(\mathcal{F}) + B\sqrt{\frac{\log(2/\delta)}{2n_{\text{task}}}}$$

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

$$\mathcal{H} = \{h_M : x \mapsto Mx \mid \|M\|_F \leq C\}$$

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

$$\left|\mathbb{E}[f_M(x)] - \frac{1}{n_{\text{task}}}\sum_{i=1}^{n_{\text{task}}} f_M(x_i)\right| \le \frac{2LCR}{\sqrt{n_{\text{task}}}} + B\sqrt{\frac{\log(2/\delta)}{2n_{\text{task}}}}$$

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

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

Define the function class

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

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

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

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

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

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 $1 - \delta'$,

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

where

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

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

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

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

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

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 $\|M - M^*\|_F \le \epsilon_{\text{task}}$ through iterative optimization. After k iterations, the error is approximately $\|M_k - M^*\|_F \le \epsilon_0(1-\alpha)^k$. To reach ϵ_{task}, we require:

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

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

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

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 6.

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 6: 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

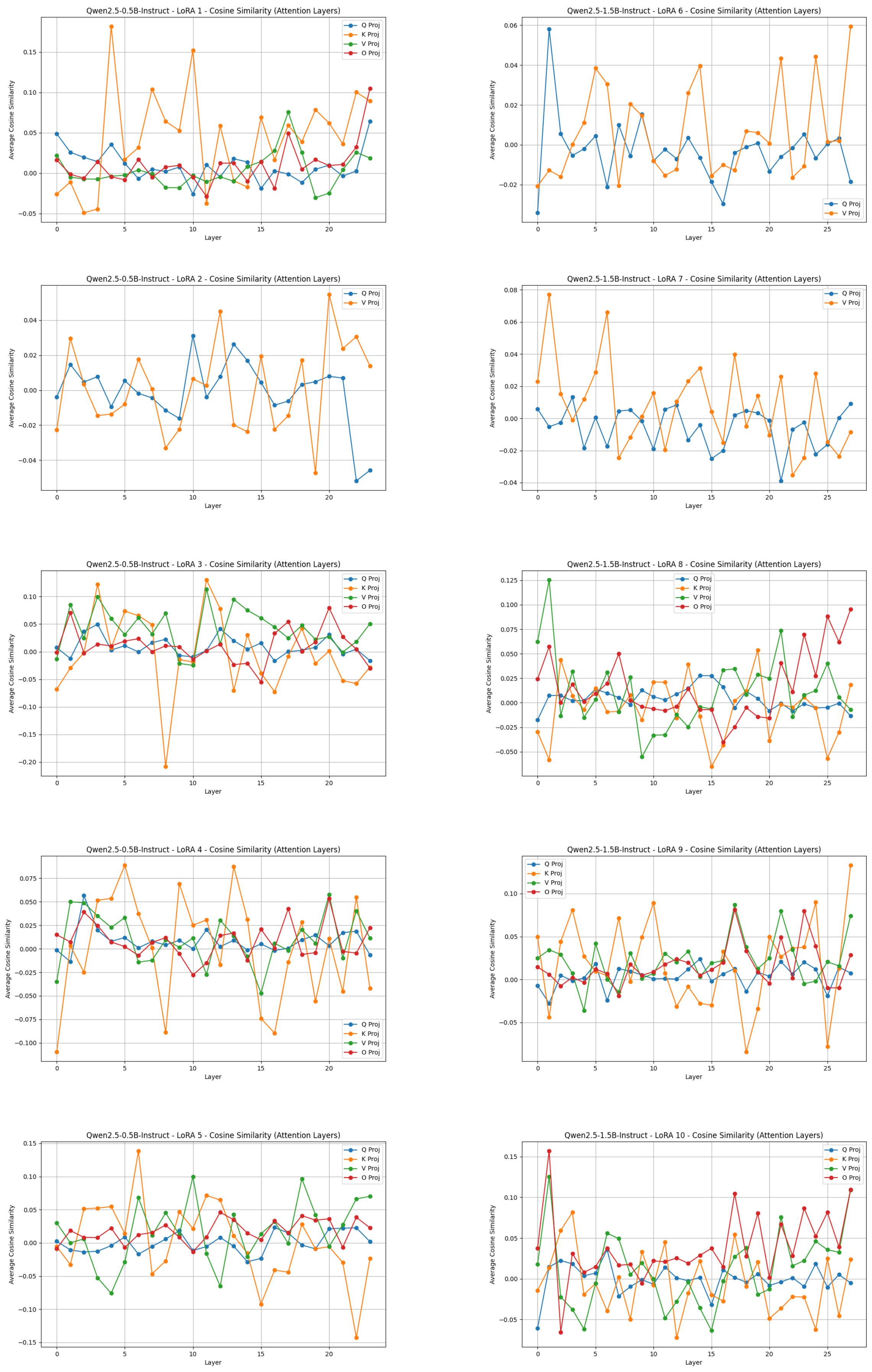

Figure 4: Cosine Similarity between Original Representation and LoRA Correction for Different Qwen2.5-Instruct LoRA Adapters (Continued on next page).

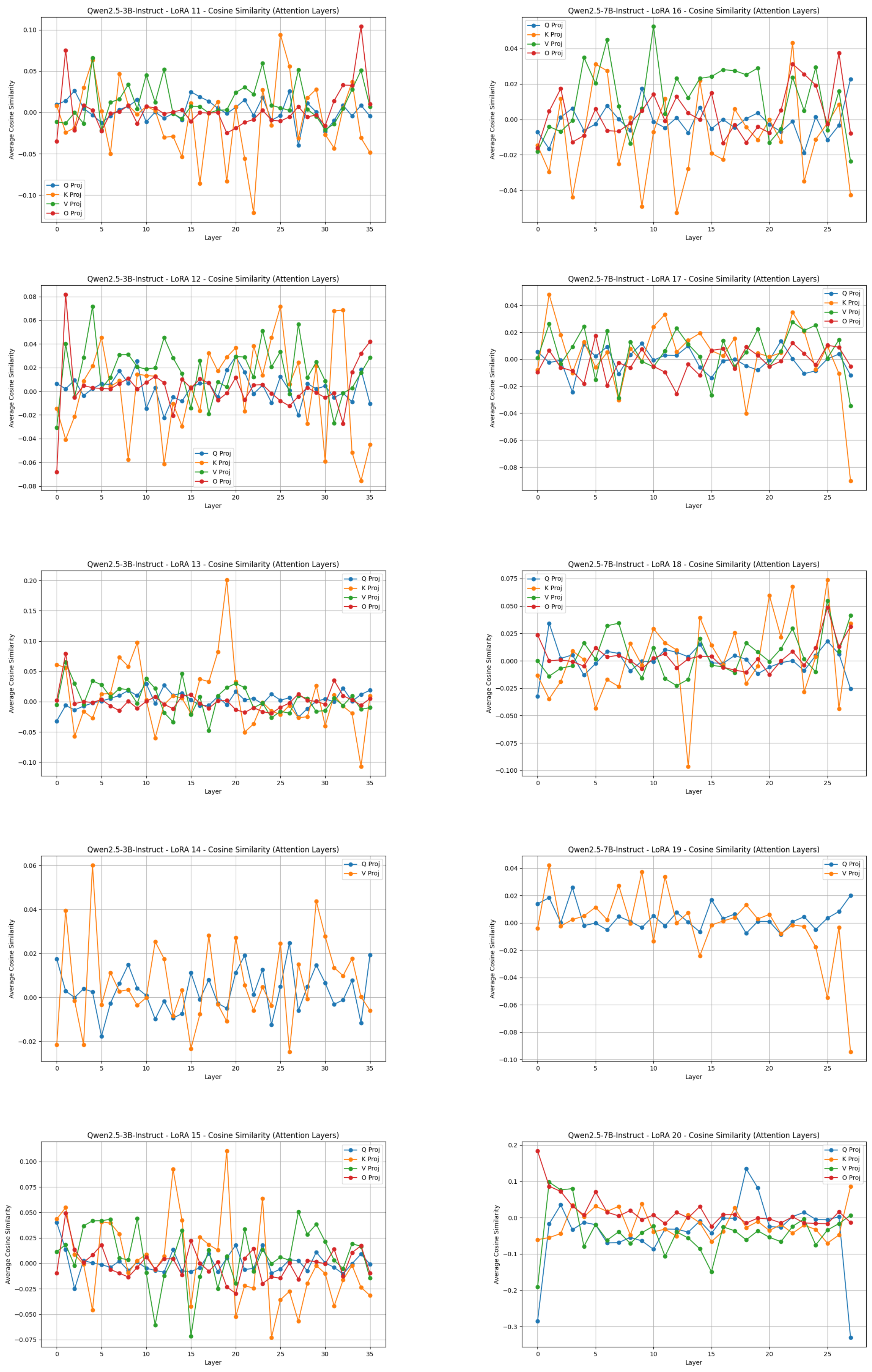

Figure 5: Cosine Similarity between Original Representation and LoRA Correction for Different Qwen2.5-Instruct LoRA Adapters (Continued from previous page).

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.

C Appendix: Creation of Irrelevant Data

A crucial component of Noise Flushing is the generation of the irrelevant dataset, D_{irr}. This dataset serves to guide the LoRA adapter to preserve the model's general capabilities and prevent it from overfitting to noise in the scarce task data.

Methodology The "self-sampled" irrelevant data (D_{irr}) is generated by leveraging the base LLM that is being fine-tuned. We take a set of diverse queries, $Q_{\text{source\_irr}}$, from a general-purpose instruction-following dataset, such as the Alpaca dataset (Taori et al., 2023). For each query $q_{\text{irr}} \in Q_{\text{source\_irr}}$, we feed it to the open-source *base LLM* (before any fine-tuning with D_{task} or D_{irr} begins) to obtain a corresponding response y_{irr}. The pair $(q_{\text{irr}}, y_{\text{irr}})$ then forms an instance in D_{irr}. This process ensures that D_{irr} reflects the model's inherent knowledge and response style on general inputs.

Ensuring Irrelevance To ensure that D_{irr} remains genuinely irrelevant to the specific target task for which D_{task} is being used, several strategies are employed:

1. **Source of Queries:** $Q_{\text{source\_irr}}$ is chosen from a domain or task category distinct from that of D_{task}. For example, if D_{task} is for a specialized medical text summarization task, $Q_{\text{source\_irr}}$ might consist of general knowledge questions, creative writing prompts, or simple conversational exchanges.

2. **Keyword Filtering:** Queries in $Q_{\text{source\_irr}}$ that inadvertently contain keywords highly specific to the target task can be filtered out. Similarly, responses y_{irr} can be checked.

3. **Instruction Differentiation:** The instructions or prompts used to generate D_{irr} are explicitly different from those defining the target task in D_{task}. The aim is for D_{irr} to represent a broad distribution of inputs where the task-specific fine-tuning should ideally have minimal impact.

By constructing D_{irr} in this manner, we provide the model with a wide range of examples where it should maintain its existing behavior, thereby regularizing the updates made by the LoRA adapter when learning from D_{task}.

D Appendix: Wall-Clock Time Comparison

Noise Flushing introduces computational overhead due to processing additional irrelevant data. With 100x irrelevant data (where marginal benefit begins diminishing), wall-clock time for NF was 82.88x that of vanilla LoRA on an NVIDIA H20 with batch size 32. This highlights the trade-off for improved performance in data-scarce scenarios.

E Appendix: Robustness of Noise Flushing

E.1 Robustness to Different Irrelevant Data Distributions

To investigate NF's robustness to the source of irrelevant data (D_{irr}), we tested different datasets for generating D_{irr} on the text2sql task (Qwen1.5-7B-Chat, BIRD-SQL Mini, 'sql-create-context' task data). Results in Table 7 suggest NF is robust, consistently outperforming the baseline.

Irrelevant Data Source (Metric Type)	EX (%)
Alpaca (w/ NF) - Moderate EX	7.20
Alpaca (w/ NF) - Challenging EX	2.94
Open-Platypus (w/ NF) - Moderate EX	8.40
Open-Platypus (w/ NF) - Challenging EX	4.90
COIG-CQIA (w/ NF) - Moderate EX	7.60
COIG-CQIA (w/ NF) - Challenging EX	3.92
Baseline (w/o NF) - Moderate EX	7.60
Baseline (w/o NF) - Challenging EX	0.98

Table 7: NF Performance with Varying Irrelevant Data Sources on BIRD-SQL Mini (text2sql, Qwen1.5-7B-Chat).

E.2 Comparison with Traditional Regularization (Dropout)

We compared LoRA with varying dropout rates against NF on the text2sql task (Qwen1.5-7B-Chat, BIRD-SQL Mini, 'sql-create-context' task data). Table 8 shows dropout had minimal impact, while NF provided clearer gains, especially on challenging examples.

Method (Metric Type)	EX (%)
LoRA (Dropout 0.0) (Baseline) - Moderate EX	7.60
LoRA (Dropout 0.0) (Baseline) - Challenging EX	0.98
LoRA (Dropout 0.05) - Moderate EX	8.00
LoRA (Dropout 0.05) - Challenging EX	0.98
LoRA (Dropout 0.1) - Moderate EX	7.20
LoRA (Dropout 0.1) - Challenging EX	0.98
LoRA (Dropout 0.2) - Moderate EX	7.20
LoRA (Dropout 0.2) - Challenging EX	0.98
LoRA (Dropout 0.3) - Moderate EX	7.60
LoRA (Dropout 0.3) - Challenging EX	0.98
LoRA w/ NF (No Dropout) - Moderate EX	7.20
LoRA w/ NF (No Dropout) - Challenging EX	2.94

Table 8: Comparison of NF with Dropout as Regularization on BIRD-SQL Mini (text2sql, Qwen1.5-7B-Chat).

E.3 Effect of Irrelevant Data Quantity on Generalization Benchmarks

We evaluated the impact of varying amounts of irrelevant data (D_{irr} from Alpaca) used in NF (Qwen1.5-7B-Chat, fine-tuned on a small, fixed task dataset) on standard benchmarks. Results (Tables 9 and 10) show NF largely preserves general capabilities at moderate D_{irr} ratios.

Irrelevant Samples (Log Scale)	GSM8K Acc (%)
0 (Baseline task-only fine-tune)	28.96
51	29.34
153	29.11
510	29.34
1632	29.04
5151	28.20
16371	27.37

Table 9: GSM8K Performance vs. Irrelevant Data Quantity with NF (Qwen1.5-7B-Chat).

Irrelevant Samples (Log Scale)	AGIEval Avg Acc (%)
0 (Baseline task-only fine-tune)	44.57
391	45.37
1173	44.69
3128	44.81
8602	43.47
21114	43.65

Table 10: AGIEval Average Accuracy vs. Irrelevant Data Quantity with NF (Qwen1.5-7B-Chat).

E.4 Performance on Larger Models

To assess scalability, we tested NF on Qwen1.5-32B-Chat for the text2sql task (BIRD-SQL Mini, 'sql-create-context' task data). Table 11 shows NF continues to provide benefits.

Model (Metric Type)	EX (%)
Qwen1.5-7B-Chat (w/o NF) - Moderate EX	7.60
Qwen1.5-7B-Chat (w/o NF) - Challenging EX	0.98
Qwen1.5-7B-Chat (w/ NF) - Moderate EX	7.20
Qwen1.5-7B-Chat (w/ NF) - Challenging EX	2.94
Qwen1.5-32B-Chat (w/o NF) - Moderate EX	19.60
Qwen1.5-32B-Chat (w/o NF) - Challenging EX	12.75
Qwen1.5-32B-Chat (w/ NF) - Moderate EX	24.40 (+24.5%)
Qwen1.5-32B-Chat (w/ NF) - Challenging EX	16.67 (+30.7%)

Table 11: NF Performance on Qwen1.5-32B-Chat for text2sql (BIRD-SQL Mini). Percentages in parentheses indicate relative change compared to the 32B model without NF.