SFTMix: Elevating Language Model Instruction Tuning with Mixup Recipe

Yuxin Xiao[1*], **Shujian Zhang**[2], **Marzyeh Ghassemi**[1], **Wenxuan Zhou**[2]
[1]Massachusetts Institute of Technology, [2]Zoom Video Communications
`yuxin102@mit.edu`

Abstract

To acquire instruction-following capabilities, large language models (LLMs) undergo instruction tuning, where they are trained on instruction-response pairs using next-token prediction (NTP). Efforts to improve instruction tuning often focus on higher-quality supervised fine-tuning (SFT) datasets, typically requiring data filtering with proprietary LLMs or human annotation. In this paper, we take a different approach by proposing SFTMix, a novel Mixup-based recipe that elevates LLM instruction tuning without relying on well-curated datasets. We observe that LLMs exhibit uneven confidence across the semantic representation space. We argue that examples with different confidence levels should play distinct roles in instruction tuning: Confident data is prone to overfitting, while unconfident data is harder to generalize. Based on this insight, SFTMix leverages training dynamics to identify examples with varying confidence levels. We then interpolate them to bridge the confidence gap and apply a Mixup-based regularization to support learning on these additional, interpolated examples. We demonstrate the effectiveness of SFTMix in both instruction-following and healthcare-specific SFT tasks, with consistent improvements across LLM families and SFT datasets of varying sizes and qualities. Extensive analyses across six directions highlight SFTMix's compatibility with data selection, adaptability to compute-constrained scenarios, and scalability to broader applications.

1 Introduction

LLMs have recently achieved strong performance across diverse natural language processing (NLP) tasks (Zhao et al., 2023; Minaee et al., 2024). After being pre-trained on large corpora of raw text, LLMs undergo a critical instruction-tuning stage (Ouyang et al., 2022; Zhang et al., 2023) to develop their instruction-following capabilities based on SFT datasets (Taori et al., 2023; Wang et al., 2023; Xu et al., 2024). During this stage, LLMs are usually trained via NTP, where they predict the next token in a response given both the instruction and the preceding tokens in that response.

Previous research in this field has predominantly focused on enhancing the quality of instruction-tuning datasets. One line of research direction seeks to better understand the intrinsic properties of these datasets (Kung et al., 2023; Lin et al., 2024) and selects informative instruction-response pairs through heuristic-based filters (Zhao et al., 2024) or LLM scoring (Chen et al., 2024a). Another line of work generates high-quality responses by querying advanced proprietary LLMs (Chen et al., 2024a) or relying on human annotators (Zhou et al., 2023).

In this paper, we take a different approach by proposing SFTMix, a novel Mixup-based (Zhang et al., 2018) recipe to elevate LLM instruction tuning without the need for well-curated datasets. Our design is motivated by the observation that an LLM's confidence distribution is uneven across the semantic representation space. Since confident data is prone to overfitting (Zhang and Vaidya, 2021; Han et al., 2024) and unconfident data is harder to generalize (Elsayed et al., 2018; Jiang et al., 2018), we argue that data with varying confidence levels should play distinct roles in instruction tuning. Hence, we first derive an LLM's confidence from its training dynamics (Swayamdipta et al., 2020) and divide the SFT dataset into confident and unconfident subsets accordingly. We then linearly interpolate between these subsets and introduce a Mixup-based regularization to support learning on these additional, interpolated examples. By propagating supervision signals across confidence regions (Bengio et al., 2009; Chapelle et al., 2009; Sohn et al., 2020) and encouraging linear behavior between them (Zhang et al., 2018; Verma et al., 2019), our recipe mitigates overfitting in confident

---

[*] Work done during an internship at Zoom.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1705–1718
July 2-7, 2026

examples while enhancing generalization in unconfident ones during LLM instruction tuning.

We demonstrate the effectiveness of our proposed SFTMix recipe in both instruction-following and domain-specific SFT settings. In particular, SFTMix significantly outperforms the conventional NTP baseline in both MT-Bench (Zheng et al., 2023) and AlpacaEval-2 (Dubois et al., 2024), with consistent improvements across LLM families (i.e., Llama (Dubey et al., 2024), Mistral (Jiang et al., 2023), and Qwen (Hui et al., 2024)) and SFT datasets of varying sizes and qualities (i.e., Alpaca-52K (Taori et al., 2023), UltraChat-200K (Tunstall et al., 2023), and Tulu3-939K (Lambert et al., 2024)). Moreover, in the healthcare domain, Llama-3.1-8B and Mistral-7B-v0.1, instruction-tuned on MedAlpaca-263K (Han et al., 2023) using SFTMix, increase the accuracy by an average of 1.5% absolutely across four question-answering benchmarks (Jin et al., 2019, 2021; Pal et al., 2022).

In addition, we conduct in-depth analyses across six directions to highlight SFTMix's versatility and scalability in LLM instruction tuning. Our results validate the importance of confidence-based data splitting for effective Mixup and show that Mixup works best as a regularization alongside NTP. Moreover, we demonstrate that SFTMix integrates seamlessly with data selection (Chen et al., 2024a; Zhao et al., 2024), adapts well to compute-constrained scenarios (Hu et al., 2022), and scales effectively to broader applications (Burns et al., 2024).

We summarize our contributions as follows:

- We introduce SFTMix, a novel recipe to elevate LLM instruction tuning without relying on well-curated SFT datasets, by interpolating semantic regions of varying confidence levels and applying a Mixup-based regularization.
- We show that SFTMix outperforms the NTP baseline across various instruction-following and healthcare-specific SFT tasks, with consistent improvements across LLM families and SFT datasets of varying sizes and qualities.
- Extensive analyses across six directions highlight that SFTMix is compatible with data selection, adaptable to compute-constrained scenarios, and scalable to broader applications.

2 Related Work

LLM Instruction Tuning. To align LLMs with user intents or domain-specific tasks, Ouyang et al. (2022) propose instruction-tuning LLMs on human-annotated demonstrations using supervised learning. The conventional NTP paradigm trains LLMs to predict response tokens sequentially given instruction-response pairs (Zhang et al., 2023). Enhancements include adding noise to token embeddings (Jain et al., 2024), commonality-aware partition (Rao et al., 2024), and explicitly modeling instructions (Shi et al., 2024). Previous work (Chiang et al., 2023; Ding et al., 2023; Taori et al., 2023; Wang et al., 2023; Xu et al., 2024) collects instruction-following datasets via LLM distillation or crowd-sourced user conversations. To improve data quality, researchers employ heuristic-based filters (Schoch et al., 2023; Zhao et al., 2024; Yang et al., 2025), importance weighting (Xie et al., 2023; Xia et al., 2024; Chen et al., 2024b), LLM scoring (Chen et al., 2024a), and human curation (Zhou et al., 2023). Other studies explore the intrinsic properties of SFT datasets (Kung et al., 2023; Lin et al., 2024; Fu et al., 2025; Zhang et al., 2025a), gradient-based methods (Wang et al., 2025; Zhao et al., 2025), and reinforcement learning from human feedback (Rafailov et al., 2023; Ethayarajh et al., 2024; Zeng et al., 2024). However, acquiring high-quality SFT data often entails substantial computational and labor costs. This paper aims to optimize data utilization through confidence-aware data interpretation and elevate instruction tuning beyond NTP without relying on well-curated datasets.

Data Characterization via Training Dynamics. Data characterization (Albalak et al., 2024; Wang et al., 2024) analyzes training data quality to improve downstream model performance. In particular, Swayamdipta et al. (2020) leverage training dynamics from a pre-trained language model (Liu et al., 2019) to create data maps. This idea has inspired advances in active learning (Zhang and Plank, 2021; Zhang et al., 2022; Kung et al., 2023), curriculum learning (Christopoulou et al., 2022; Lin et al., 2024; Poesina et al., 2024), dataset pruning (Chimoto et al., 2024; He et al., 2024; Lin et al., 2024; Seedat et al., 2024), and LLM scaling (Mircea et al., 2025; Qi et al., 2025; Zhang et al., 2025b). Here, we apply training dynamics to causal language generation by categorizing an SFT dataset into confident and unconfident subsets, which facilitates the subsequent Mixup-based regularization during LLM instruction tuning.

Mixup-Based Learning. To mitigate memorization and adversarial sensitivity, Mixup Zhang et al. (2018) trains models on convex combinations of paired inputs and labels. Its variants (Verma et al., 2019; Hendrycks et al., 2020; Uddin et al., 2021; Choi et al., 2022) interpolate feature representations at various stages, guided by different training signals. Theoretical analyses (Zhang et al., 2021; Carratino et al., 2022; Chidambaram et al., 2022; Park et al., 2022; Pinto et al., 2022) highlight its adaptive regularization and generalization effects, yielding stronger out-of-distribution robustness and better uncertainty calibration. Empirical studies confirm its effectiveness in semi-supervised learning (Berthelot et al., 2019, 2020; Li et al., 2020, 2022) and NLP (Chen et al., 2020; Guo et al., 2020; Sun et al., 2020; Park and Caragea, 2022; Yang et al., 2022). In this paper, we extend Mixup to LLM instruction tuning, proposing a regularization method to reduce overfitting to confident examples while supporting learning for less confident ones.

3 SFTMix

Based on the preliminaries in §3.1, we discuss the motivation in §3.2, introduce SFTMix in §3.3, and analyze its effect on the tuning process in §3.4.

3.1 Preliminaries

The NTP Instruction-Tuning Paradigm. Consider an SFT dataset $\mathcal{D} = \{(\mathcal{X}_i, \mathcal{Y}_i)\}_{i=1}^{|\mathcal{D}|}$, which consists of pairs of instructions $\mathcal{X}_i$ and desired responses $\mathcal{Y}_i$. Here, $\mathcal{X}_i = (x_1, \ldots, x_{M_i})$ and $\mathcal{Y}_i = (y_1, \ldots, y_{N_i})$ are sequences of tokens. For an LLM with multiple transformer layers (Vaswani et al., 2017) and a linear causal language modeling head $\mathbf{W}$, the conventional NTP task minimizes the following loss for predicting $\mathcal{Y}_i$ given $\mathcal{X}_i$:

$$\begin{aligned}\ell_{\text{NTP}}(\mathcal{D}) &= -\sum_{i=1}^{|\mathcal{D}|}\sum_{n=1}^{N_i} \log p(y_n \mid \mathcal{X}_i, y_1, \ldots, y_{n-1}) \\ &= -\sum_{i=1}^{|\mathcal{D}|}\sum_{n=1}^{N_i} H(\mathbf{Y}_n, \sigma(\mathbf{Z}_n^\top \mathbf{W})).\end{aligned}$$

This loss equals the sum of negative cross-entropy H between $\mathbf{Y}_n$ and $\mathbf{Z}_n^\top \mathbf{W}$ after softmax σ, where $\mathbf{Y}_n$ is the one-hot encoding of the n-th token in $\mathcal{Y}_i$. The corresponding representation $\mathbf{Z}_n$ is the last hidden state from the LLM's transformer layers: $\mathbf{Z}_n = \text{Transformers}(\mathcal{X}_i, y_1, \ldots, y_{n-1})$.

LLM Confidence via Training Dynamics. Suppose we collect C checkpoints of an LLM when instruction-tuning it on a dataset $\mathcal{D}$ via NTP. We can capture the training dynamics (Swayamdipta et al., 2020) of this LLM by computing its confidence in generating each pair $(\mathcal{X}_i, \mathcal{Y}_i) \in \mathcal{D}$. Specifically, let $\text{Perp}_c(\mathcal{Y}_i \mid \mathcal{X}_i)$ denote the LLM's perplexity for an instruction-response pair $(\mathcal{X}_i, \mathcal{Y}_i)$ at checkpoint $c \in \{1, \ldots, C\}$. Since lower perplexity implies higher generation likelihood, we define its confidence in predicting $\mathcal{Y}_i$ given $\mathcal{X}_i$ as the negative average perplexity over the C checkpoints:

$$\text{Conf}\,(\mathcal{Y}_i \mid \mathcal{X}_i) = -\frac{1}{C}\sum_{c=1}^{C} \text{Perp}_c(\mathcal{Y}_i \mid \mathcal{X}_i).$$

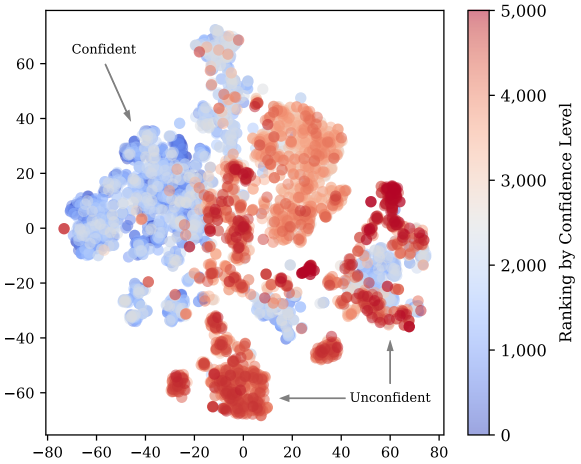


Figure 1: Embeddings of 2,500 most and 2,500 least confident examples in Alpaca-52K by Llama-3.1-8B trained using NTP. The clear separation between these embeddings suggests that the LLM exhibits varying confidence levels across different semantic regions.

3.2 Motivation

We motivate the design of SFTMix through a case study. Specifically, we instruction-tune Llama-3.1-8B (Dubey et al., 2024) on Alpaca-52K (Taori et al., 2023) and collect the LLM's confidence for each training data point across five checkpoints. Using the last hidden state $\mathbf{Z}$ of the final token in $(\mathcal{X}_i, \mathcal{Y}_i)$ as its semantic representation, we visualize 2,500 most and 2,500 least confident examples via t-SNE (Van der Maaten and Hinton, 2008). As shown in Figure 1, embeddings of data points with contrasting confidence levels are clearly separated, indicating that **the LLM exhibits uneven confidence across the semantic representation space.**

Furthermore, we analyze the distributions of instruction topics in the 50 most and 50 least confident examples. We find that the most confident examples primarily involve deterministic grammar tasks (e.g., "correct any grammar error in the following sentence"), while 56% of the least confident

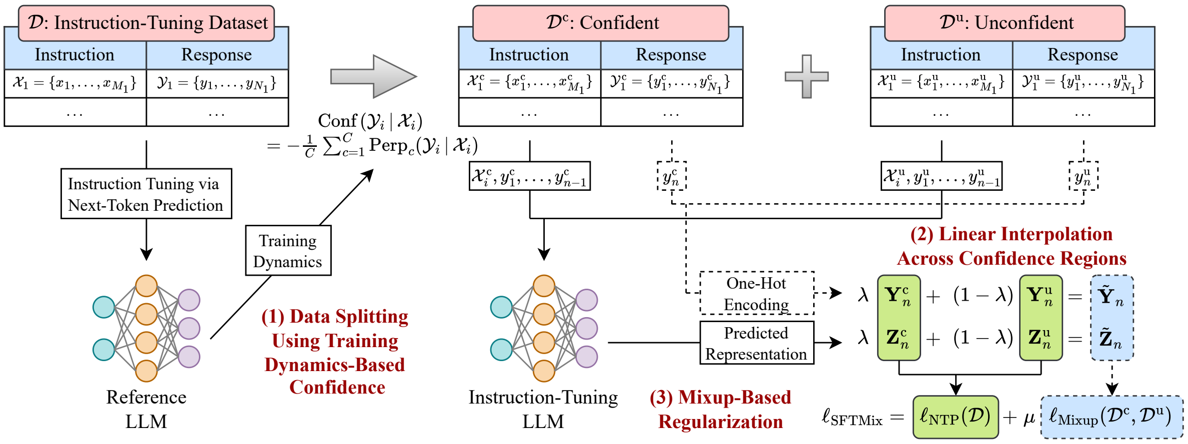


Figure 2: The overall pipeline of the three-stage SFTMix recipe for LLM instruction tuning.

examples require creative content generation (e.g., "find a name for an e-commerce website"), and the remaining 44% consist of noisy or unanswered instructions. This aligns with our observation in Figure 1, showing that the LLM's confidence varies across different semantic regions.

The insight from the case study motivates us to contend that **data with distinct confidence levels should play different roles during instruction tuning.** Highly confident data points typically lie further from the classification decision boundary, posing a higher risk of overfitting (Zhang and Vaidya, 2021; Han et al., 2024). In contrast, less confident data points are often closer to the boundary, making them harder to learn (Elsayed et al., 2018; Jiang et al., 2018).

To address this, we propose SFTMix, a Mixup-based (Zhang et al., 2018) recipe (details in §3.3). Leveraging training dynamics-based confidence, we first linearly interpolate between confident and unconfident examples to bridge the confidence gap across the semantic representation space. Then, we introduce a Mixup-based regularization to support learning on these additional, interpolated examples. By promoting the flow of supervision signals between regions of differing confidence levels (Bengio et al., 2009; Chapelle et al., 2009) and encouraging linear behavior over a smoother decision boundary (Zhang et al., 2018), our regularization mitigates overfitting in confident examples and enhances generalization in unconfident ones during LLM instruction tuning.

3.3 Recipe

We now introduce the details of our three-step SFTMix recipe (illustrated in Figure 2).

Step 1: Determine Subspaces with Distinct Confidence Levels. Given an SFT dataset $\mathcal{D}$, we first instruction-tune a reference LLM via NTP and collect its confidence $\mathrm{Conf}\,(\mathcal{Y}_i \mid \mathcal{X}_i)$ as in §3.1 for each pair $(\mathcal{X}_i, \mathcal{Y}_i) \in \mathcal{D}$. We then divide $\mathcal{D}$ into two equal-sized subsets according to $\mathrm{Conf}\,(\mathcal{Y}_i \mid \mathcal{X}_i)$: a confident subset $\mathcal{D}^{\mathrm{c}}$ and an unconfident subset $\mathcal{D}^{\mathrm{u}}$.

Step 2: Linearly Interpolate Confident and Unconfident Examples. Consider a confident instruction-response pair $(\mathcal{X}^{\mathrm{c}}_i, \mathcal{Y}^{\mathrm{c}}_i) \in \mathcal{D}^{\mathrm{c}}$ and an unconfident pair $(\mathcal{X}^{\mathrm{u}}_i, \mathcal{Y}^{\mathrm{u}}_i) \in \mathcal{D}^{\mathrm{u}}$. Let $\mathbf{Y}^{\mathrm{c}}_n$ and $\mathbf{Y}^{\mathrm{u}}_n$ be the one-hot encoding vectors of the n-th token in $\mathcal{Y}^{\mathrm{c}}$ and $\mathcal{Y}^{\mathrm{u}}$, respectively, with $\mathbf{Z}^{\mathrm{c}}_n$ and $\mathbf{Z}^{\mathrm{u}}_n$ as the corresponding representations predicted by the target instruction-tuning LLM (different from the reference LLM used in Step 1). We linearly interpolate the two pairs as follows:

$$\tilde{\mathbf{Z}}_n = \lambda \mathbf{Z}^{\mathrm{c}}_n + (1-\lambda)\mathbf{Z}^{\mathrm{u}}_n,$$
$$\tilde{\mathbf{Y}}_n = \lambda \mathbf{Y}^{\mathrm{c}}_n + (1-\lambda)\mathbf{Y}^{\mathrm{u}}_n,$$

where $\lambda \sim \mathrm{Beta}(\alpha, \alpha)$ and α is a hyperparameter.

Step 3: Incorporate a Mixup-Based Regularization. Suppose that $N'_i = \min(N^{\mathrm{c}}_i, N^{\mathrm{u}}_i)$ represents the length of the shorter response between $\mathcal{Y}^{\mathrm{c}}_i$ and $\mathcal{Y}^{\mathrm{u}}_i$. We define the Mixup-based regularization $\ell_{\mathrm{Mixup}}(\mathcal{D}^{\mathrm{c}}, \mathcal{D}^{\mathrm{u}})$ between the confident and unconfident subsets and the overall instruction-tuning loss ℓ_{SFTMix} used in our SFTMix recipe as follows:

$$\ell_{\mathrm{Mixup}}(\mathcal{D}^{\mathrm{c}}, \mathcal{D}^{\mathrm{u}}) = -\sum_{i=1}^{|D|/2} \sum_{n=1}^{N'_i} H(\tilde{\mathbf{Y}}_n, \sigma(\tilde{\mathbf{Z}}_n^{\top} \mathbf{W})),$$
$$\ell_{\mathrm{SFTMix}}(\mathcal{D}) = \ell_{\mathrm{NTP}}(\mathcal{D}) + \mu\, \ell_{\mathrm{Mixup}}(\mathcal{D}^{\mathrm{c}}, \mathcal{D}^{\mathrm{u}}).$$

Here, μ is a hyperparameter to control the regularization effect. For ease of implementation, we ensure that each training batch contains an equal number of confident and unconfident examples, which are then randomly paired for the linear interpolation and the Mixup-based regularization.

3.4 Analysis

Here, we analyze how the Mixup-based regularization affects the direction of gradient descent in the original NTP paradigm. For notational simplicity, we focus on the cross-entropy $H(\tilde{\mathbf{Y}}, \sigma(\tilde{\mathbf{Z}}^\top \mathbf{W}))$ between the interpolated one-hot encoding vector $\tilde{\mathbf{Y}}$ and the corresponding interpolated representation $\tilde{\mathbf{Z}}$ from the target LLM's last transformer layer.

Let $\tilde{\mathbf{S}} = \tilde{\mathbf{Z}}^\top \mathbf{W}$, the gradient of H w.r.t. $\tilde{\mathbf{S}}$ is

$$\nabla_{\tilde{\mathbf{S}}} H(\tilde{\mathbf{Y}}, \sigma(\tilde{\mathbf{S}})) = \sigma(\tilde{\mathbf{S}}) - \tilde{\mathbf{Y}}.$$

Using the chain rule, we have

$$\nabla_{\mathbf{W}} H(\tilde{\mathbf{Y}}, \sigma(\tilde{\mathbf{Z}}^\top \mathbf{W})) = \tilde{\mathbf{Z}}^\top \left(\sigma(\tilde{\mathbf{Z}}^\top \mathbf{W}) - \tilde{\mathbf{Y}} \right).$$

Since the gradient w.r.t. $\mathbf{W}$ involves the nonlinear softmax operation σ, we have

$$\begin{aligned} \sigma(\tilde{\mathbf{Z}}^\top \mathbf{W}) &= \sigma(\lambda \mathbf{Z}^{\mathrm{c}\top} \mathbf{W} + (1-\lambda) \mathbf{Z}^{\mathrm{u}\top} \mathbf{W}) \\ &\neq \lambda \sigma(\mathbf{Z}^{\mathrm{c}\top} \mathbf{W}) + (1-\lambda) \sigma(\mathbf{Z}^{\mathrm{u}\top} \mathbf{W}). \end{aligned}$$

In other words, the gradient from the regularization with the interpolated example does not decompose into a weighted sum of the gradients from the NTP loss on the corresponding confident and unconfident examples. As a result, the Mixup-based regularization modifies the gradient descent direction in NTP by incorporating these interpolated examples.

4 Experiments

We assess the effectiveness of SFTMix against the NTP baseline in both instruction-following (§4.1) and domain-specific (§4.2) SFT tasks.

4.1 Instruction-Following SFT

Experiment Setup. We compare SFTMix with NTP and a sample-reweighting baseline, IR-DRO (Chen et al., 2024b), using three pre-trained LLMs: Llama-3.1-8B (Llama) (Dubey et al., 2024), Mistral-7B-v0.1 (Mistral) (Jiang et al., 2023), and Qwen-2.5-14B (Qwen) (Hui et al., 2024). Due to computational constraints, we train Qwen only on the smaller, uncurated Alpaca-52K (Taori et al., 2023), while Llama and Mistral are trained on both Alpaca-52K and the larger filtered datasets: UltraChat-200K (Tunstall et al., 2023) and Tulu3-939K (Lambert et al., 2024). For the same reason, IR-DRO is evaluated only on Llama and Mistral using Alpaca-52K. We then evaluate the instruction-tuned LLMs on two instruction-following benchmarks: MT-Bench (Zheng et al., 2023) and AlpacaEval-2 (Dubois et al., 2024). Following Zhao et al. (2024), we also conduct a human evaluation for head-to-head comparisons using the Vicuna subset in AlpacaEval-2.

LLM	Recipe	MT-Bench			AlpacaEval-2	
		ST	MT	Overall	WR	LC WR
Dataset: Alpaca-52K						
Llama	IR-DRO	4.8503	3.6121	4.2312	4.1019	8.7509
	NTP	4.9100	3.8150	4.3625	4.0714	8.6528
	SFTMix	**5.2125**	**3.9525**	**4.5825**	**4.9031**	**10.3195**
Mistral	IR-DRO	5.1127	4.0522	4.5825	4.3411	9.2137
	NTP	5.1650	4.0675	4.6163	4.3560	9.1759
	SFTMix	**5.2775**	**4.5425**	**4.9100**	**4.5386**	**9.4994**
Qwen	NTP	6.8177	5.5683	6.1930	7.0764	13.9508
	SFTMix	**7.1298**	**5.9196**	**6.5247**	**7.8810**	**15.0235**
Dataset: UltraChat-200K						
Llama	NTP	6.1875	5.0125	5.6000	5.0665	8.4505
	SFTMix	**6.2750**	**5.3500**	**5.8125**	**5.1149**	**9.3810**
Mistral	NTP	5.7625	4.6938	5.2281	4.4899	7.7732
	SFTMix	**5.9813**	**4.8813**	**5.4313**	**4.6117**	**8.7650**
Dataset: Tulu3-939K						
Llama	NTP	6.2500	4.9625	5.6063	7.1045	12.0811
	SFTMix	**6.6721**	**5.3331**	**6.0026**	**8.0129**	**13.1345**
Mistral	NTP	5.9246	4.0128	4.9687	6.9926	11.2471
	SFTMix	**6.3626**	**4.3398**	**5.3512**	**7.5421**	**11.8681**

Table 1: Evaluation of instruction-following capabilities of LLMs trained with NTP, SFTMix, and IR-DRO. We highlight the best-performing instruction-tuning recipe in bold. SFTMix outperforms the baselines consistently across instruction-tuning datasets and LLMs.

Implementation Details. By default, we use a separate instance of the same type as the target instruction-tuning LLM to obtain $\mathrm{Conf}(\mathcal{Y}_i \mid \mathcal{X}_i)$ in Step 1 of SFTMix. We train each LLM on Alpaca-52K for three epochs and on UltraChat-200K and Tulu3-939K for one epoch, with a batch size of 32 on eight H100 GPUs. The tuning process employs the AdamW optimizer with a learning rate of 2e$-$6, a weight decay of 0.1, and a cosine learning rate scheduler with a warm-up ratio of 0.1. Based on our hyperparameter search in §A.1, we set $\alpha = 0.5$ for sampling λ and $\mu = 0.2$ when constructing ℓ_{SFTMix}. The NTP baseline follows the same setup but excludes the Mixup-based regularization ℓ_{Mixup}. When training on UltraChat-200K and Tulu3-939K, we expand each multi-turn interaction into multiple single-turn interactions by incorporating the chat history into the instructions. In MT-Bench and AlpacaEval-2, we employ GPT-4 (Achiam et al., 2023) for LLM-as-a-judge and report the results averaged over five evaluation rounds.

Evaluation Results. As illustrated in Table 1, instruction-tuning with SFTMix consistently outperforms NTP and IR-DRO across all metrics in both evaluation benchmarks, regardless of the base LLM or SFT dataset. Notably, SFTMix yields a greater improvement in the multi-turn (MT) conversational ability (an average increase of 0.32) compared to single-turn (ST) performance (an average increase of 0.27) in MT-Bench. In AlpacaEval-2, the improvement is particularly evident in the length-controlled win rate (Dubois et al., 2024) (LC WR), which better aligns with human judgment by adjusting for GPT-4's preference for longer responses. While instruction-tuning with the larger, higher-quality UltraChat-200K dataset results in higher scores in MT-Bench and raw win rates (WRs) in AlpacaEval-2, it also produces longer responses, leading to relatively lower LC WRs. In contrast, SFTMix with Tulu3-939K delivers larger gains across all metrics than UltraChat-200K.

Further Analysis. Our human evaluation indicates that instruction-tuning with SFTMix wins 42.5% of the head-to-head comparisons, while NTP wins only 26.5% (details in §A.2). This agrees with the conclusion from LLM-as-a-judge evaluations. We also provide per-category scores and qualitative examples from MT-Bench in §A.3 to illustrate these differences and demonstrate the effectiveness of SFTMix in multilingual instruction tuning in §A.4. Furthermore, we compare the confidence distribution of Llama instruction-tuned with SFTMix versus NTP on Alpaca-52K. SFTMix reduces the standard deviation of confidence scores by 7%, indicating a more uniform distribution. This suggests SFTMix helps prevent overfitting on confident samples while better supporting learning from less confident ones.

4.2 Domain-Specific SFT

Experiment Setup. In healthcare-specific SFT, we train Llama and Mistral on the MedAlpaca-263K medical dataset (Han et al., 2023) using either NTP or SFTMix for two epochs, keeping other hyperparameters as in §4.1. We assess their performance on four healthcare-related question-answering benchmarks: MedQA (Jin et al., 2021), its five-option variant MedQA-5, PubMedQA (Jin et al., 2019), and MedMCQA (Pal et al., 2022). Following Labrak et al. (2024), we adopt a three-shot setting and report the mean accuracy over three evaluation runs. For comparison, we also include prior biomedical LLMs of similar size: MedAlpaca-7B (Han et al., 2023), PMC-LLaMA-7B (Wu et al., 2024), BioMedGPT-LM-7B (Luo et al., 2023), Meditron-7B (Chen et al., 2023a), and BioMistral-7B (Labrak et al., 2024).

LLM	**Med QA**	**Med QA-5**	**PubMed QA**	**MedMC QA**	**Ave**
Existing 7B Biomedical LLMs					
MedAlpaca	38.94	33.96	57.20	34.90	41.25
PMC-LLaMA	27.94	21.24	54.87	24.57	32.16
BioMedGPT	38.62	34.72	58.27	35.57	41.80
Meditron	35.09	26.73	56.93	34.03	38.20
BioMistral	43.86	37.58	50.13	44.14	43.93
Dataset: MedAlpaca-263K					
Llama	59.68	53.23	73.40	52.79	59.78
+ NTP	59.31	54.52	75.40	53.65	60.72
or SFTMix	**60.88**	**55.38**	**77.80**	**54.15**	**62.05**
Mistral	49.18	43.94	72.33	47.98	53.36
+ NTP	49.10	44.62	75.40	48.15	54.32
or SFTMix	**51.77**	**45.72**	**77.40**	**49.03**	**55.98**

Table 2: Evaluation results on four healthcare-related benchmarks by prior biomedical LLMs and LLMs trained using either NTP or SFTMix. We bold the scores from the best-performing instruction-tuning recipe. SFTMix achieves a 1.5% absolute increase in average accuracy compared to NTP for both Llama and Mistral.

Reference LLM	**MT-Bench**			**AlpacaEval-2**	
	ST	**MT**	**Overall**	**WR**	**LC WR**
Same	**5.2125**	3.9525	**4.5825**	**4.9031**	**10.3195**
Weaker	4.8500	**4.2625**	4.5563	4.5786	10.0483

Table 3: Evaluation of using different reference LLMs to obtain confidence in SFTMix. By default, SFTMix uses a reference LLM of the same type as the target instruction-tuning LLM, while "Weaker" refers to using a less capable reference LLM. Generalizing training dynamics from a weaker reference LLM performs comparably to using the same reference LLM.

Evaluation Results. Table 2 shows that SFTMix consistently surpasses NTP across all benchmarks for both LLMs. In particular, SFTMix leads to a 1.33% absolute improvement (from 60.72% to 62.05%) for Llama and a 1.66% increase (from 54.32% to 55.98%) for Mistral in average accuracy

NTP Data Quality	Mixup Included?	MT-Bench ST	MT-Bench MT	MT-Bench Overall	AlpacaEval-2 WR	AlpacaEval-2 LC WR
High	No	**6.1175**	**5.2575**	**5.6875**	**7.2636**	11.4490
High + Low	No	5.9000	5.1825	5.5412	6.5871	**11.9590**
High + Low	Yes	5.8025	5.0975	5.4500	5.9382	11.1768

Table 4: Evaluation of performing Mixup based on known data quality. "High" refers to the higher-quality examples from GPT-4, while "Low" refers to the lower-quality original examples in Alpaca-52K. Simply applying Mixup regularization between these subsets does not necessarily improve performance further.

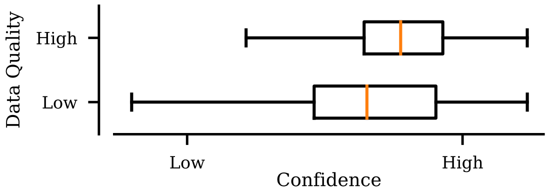


Figure 3: Confidence distributions from instruction-tuning Llama on datasets of varying qualities. On the y-axis, "High" represents higher-quality examples from GPT-4, while "Low" denotes lower-quality original examples from Alpaca-52K. Llama's confidence distributions show substantial overlap across these datasets.

across the four benchmarks. These models also significantly outperform existing biomedical LLMs across all benchmarks by a clear margin.

5 Analysis

Building on SFTMix's effectiveness in §4, we analyze SFTMix in depth across six directions by instruction-tuning Llama on Alpaca-52K.

5.1 Generalizing the Training Dynamics from a Weaker Reference LLM Is Feasible

Inspired by Burns et al. (2024), we explore whether training dynamics from a weaker reference LLM transfer to a stronger one. Specifically, we use Gemma-2B (Team et al., 2024) to split Alpaca-52K into confident and unconfident subsets, then use them for Mixup regularization when tuning Llama.

In Table 3, this alternative approach yields comparable scores on MT-Bench and AlpacaEval-2 to the default SFTMix recipe, which uses the same LLM for both training dynamics and Mixup-based instruction tuning. This finding aligns with the weak-to-strong generalization Burns et al. (2024) and suggests that confidence scores precomputed from weaker LLMs can be reused for stronger models, amortizing the computational cost over time.

Role of ℓ_{NTP}	Role of ℓ_{Mixup}	MT-Bench ST	MT-Bench MT	MT-Bench Overall	AlpacaEval-2 WR	AlpacaEval-2 LC WR
Loss	-	4.9100	3.8150	4.3625	4.0714	8.6528
Loss	Reg.	**5.2125**	3.9525	**4.5825**	**4.9031**	**10.3195**
Loss	Loss	4.7050	**4.1075**	4.4062	3.9450	8.2856
-	Loss	5.0125	4.0000	4.5062	3.5821	7.2964

Table 5: Evaluation of the optimal role of ℓ_{Mixup} alongside ℓ_{NTP}. By default, SFTMix incorporates ℓ_{Mixup} as a regularization together with the NTP loss ℓ_{NTP}. This setting achieves the highest scores across most metrics.

NTP Dataset	Mixup Included?	MT-Bench ST	MT-Bench MT	MT-Bench Overall	AlpacaEval-2 WR	AlpacaEval-2 LC WR
Full	Yes	**5.2125**	**3.9525**	**4.5825**	**4.9031**	**10.3195**
	No	4.9100	3.8150	4.3625	4.0714	8.6528
Conf.	Yes	**4.9775**	**4.1075**	**4.5425**	**4.4496**	**9.7824**
	No	4.7620	3.8206	4.2913	3.9012	8.0425
Unconf.	Yes	**5.1800**	**3.9050**	**4.5425**	**4.2030**	**8.9392**
	No	4.7164	3.8392	4.2778	3.6552	7.9889

Table 6: Evaluation of using the confident subset, the unconfident subset, or the full dataset for NTP. By default, SFTMix applies ℓ_{NTP} to the full dataset alongside ℓ_{Mixup}. This setting achieves the best results among the variants, demonstrating SFTMix's effectiveness in leveraging a larger set of training examples.

5.2 Training Dynamics-Based Confidence Is Crucial for Performing Mixup

We now explore whether we can substitute training dynamics-based confidence with known data quality. To test this hypothesis, we replace half of the original responses in Alpaca-52K with higher-quality GPT-4-generated versions (Peng et al., 2023), forming the "High" subset, while referring to the remaining lower-quality original responses as "Low". We then train Llama using three approaches: (1) NTP on High, (2) NTP on the combined High + Low dataset, and (3) NTP on High + Low with the Mixup regularization applied between them.

The use of higher-quality responses from GPT-4 indeed enhances instruction-tuning performance on both MT-Bench and AlpacaEval-2, as shown in Table 4. However, simply applying Mixup between the two datasets of varying quality does not necessarily improve performance further, as indicated by the drop in the overall MT-Bench score from 5.5412 to 5.4500 and the LC WR in AlpacaEval-2 from 11.9590 to 11.1768. To investigate this observation, we plot the LLM's confidence distributions for both the High and Low subsets in Figure 3. The substantial overlap in confidence distributions sug-

Data Selection	Recipe	MT-Bench			AlpacaEval-2	
		ST	MT	Overall	WR	LC WR
AlpaGasus	NTP	4.9787	3.5275	4.2531	4.0752	8.7182
	SFTMix	**5.1725**	**3.9663**	**4.5694**	**4.9006**	**10.3089**
Long	NTP	4.9338	3.8936	4.4137	4.2691	8.8523
	SFTMix	**5.3162**	**3.9262**	**4.6212**	**5.0230**	**10.4514**
Uncurated	NTP	4.9100	3.8150	4.3625	4.0714	8.6528
	SFTMix	**5.2125**	**3.9525**	**4.5825**	**4.9031**	**10.3195**

Table 7: Evaluation of SFTMix's compatibility with data selection methods. SFTMix seamlessly integrates with them to further enhance LLM instruction tuning.

Using LoRA?	Recipe	MT-Bench			AlpacaEval-2	
		ST	MT	Overall	WR	LC WR
Yes	NTP	4.9350	3.7600	4.3475	3.8841	8.5104
	SFTMix	**5.3350**	**3.8088**	**4.5719**	**4.8785**	**9.8030**
No	NTP	4.9100	3.8150	4.3625	4.0714	8.6528
	SFTMix	**5.2125**	**3.9525**	**4.5825**	**4.9031**	**10.3195**

Table 8: Evaluation of SFTMix's adaptability to LoRA. SFTMix outperforms NTP when using LoRA, adapting well to compute-constrained scenarios.

gests that data quality does not necessarily correlate with training dynamics-based confidence. This highlights the importance of training dynamics in determining the model-specific role of data points, which is crucial for effectively applying SFTMix.

5.3 Incorporating Mixup as a Regularization Is More Effective

To fully explore the effect of our proposed Mixup regularization ℓ_{Mixup}, we experiment two alternative treatments: (1) treating ℓ_{Mixup} as an additional loss alongside ℓ_{NTP} (i.e., $\ell = \ell_{\text{NTP}} + \ell_{\text{Mixup}}$), rather than as a regularization; and (2) minimizing only ℓ_{Mixup} without ℓ_{NTP} (i.e., $\ell = \ell_{\text{Mixup}}$).

Table 5 shows that these two variants achieve higher scores on MT-Bench but perform worse on AlpacaEval-2 compared to the NTP baseline (i.e., using only the NTP loss). Furthermore, our SFTMix recipe, which employs ℓ_{Mixup} as a regularization together with ℓ_{NTP}, still outperforms both variants across both benchmarks. This finding highlights the importance of incorporating the traditional NTP task during SFT and supports the conclusion that Mixup is more effective when used as a regularization alongside the standard cross-entropy loss in LLM instruction tuning.

5.4 SFTMix Effectively Utilizes Entire Instruction-Tuning Datasets

As part of our SFTMix recipe, we apply the NTP loss ℓ_{NTP} to the full SFT dataset. Here, we consider variants where ℓ_{NTP} is applied selectively to either the confident or unconfident halves of the dataset, with or without the Mixup regularization ℓ_{Mixup}.

As shown in Table 6, both variants that apply ℓ_{NTP} to only half of the dataset while incorporating Mixup achieve the same overall score on MT-Bench. However, the variant applying ℓ_{NTP} to the confident subset performs better on AlpacaEval-2. Notably, both variants—where ℓ_{NTP} is applied to only half of the dataset while including Mixup—outperform the NTP baseline that applies ℓ_{NTP} to the entire dataset without Mixup. We attribute this improvement to the impact introduced by our Mixup regularization ℓ_{Mixup}. Nevertheless, our SFTMix recipe, which leverages the full dataset for NTP and applies ℓ_{Mixup}, outperforms all these variants, demonstrating its ability to effectively utilize a larger set of potentially lower-quality training examples during instruction tuning.

5.5 SFTMix Integrates Well with Data Selection Methods

Although SFTMix performs effectively with the uncurated Alpaca-52K in §4.1, it can be seamlessly integrated with various data selection methods. Here, we first select 1,000 high-quality examples from Alpaca-52K using either AlpaGasus (Chen et al., 2024a), which grades responses with proprietary LLMs, or Long (Zhao et al., 2024), which chooses the longest responses. We then apply either NTP or SFTMix to the selected examples.

As shown in Table 7, applying SFTMix to the uncurated dataset outperforms NTP with either data selection strategy. Instruction tuning on the AlpaGasus-selected subset matches performance on the full dataset, while using the longest examples performs slightly better. Nevertheless, combining SFTMix with either method yields substantial improvements over the NTP baseline. This suggests that integrating SFTMix with existing data selection strategies (Albalak et al., 2024; Wang et al., 2024) could further enhance performance in LLM instruction tuning.

5.6 SFTMix Is Compatible with Parameter-Efficient Fine-Tuning

To enable parameter-efficient fine-tuning, we test SFTMix's compatibility with low-rank adaptation (LoRA) (Hu et al., 2022). Specifically, we compare SFTMix and NTP using both LoRA and full-parameter fine-tuning, with the results in Table 8.

Overall, LoRA performs comparably to full-parameter SFT in MT-Bench but slightly underperforms in AlpacaEval-2. Even with LoRA-based instruction tuning, SFTMix effectively improves performance over the NTP baseline, demonstrating its adaptability to compute-constrained scenarios.

6 Conclusion

In this paper, we propose SFTMix, a novel recipe for elevating LLM instruction tuning. We observe that LLMs exhibit uneven confidence distributions across the semantic space, and argue that data with different confidence levels should play distinct roles in tuning. To this end, we partition an SFT dataset into confident and unconfident subsets, interpolate them to bridge the confidence gap, and introduce a Mixup-based regularization to facilitate learning. Extensive experiments show that SFTMix outperforms the conventional NTP paradigm across diverse LLM families and SFT datasets. Our analyses further highlight its versatility and scalability. Applying dynamic scheduling to Mixup regularization and extending it to LLM pre-training are promising directions for future work.

Limitation

Due to computational constraints, we do not apply SFTMix to LLM pre-training or instruction-tune models larger than 14B. We also acknowledge that SFTMix requires an initial fine-tuning round to estimate training dynamics-based confidence. While this adds computational cost, it is comparable to prior data selection methods such as LESS (Xia et al., 2024) and Rho-1 (Lin et al., 2024), which also depend on reference models. To mitigate this cost, we explore using confidence scores from weaker LLMs in §5.1, though further experiments are needed to robustly establish their generalization. We also examine SFTMix's compatibility with lightweight data selection methods in §5.5, but additional evaluation is required to validate its effectiveness across varied strategies.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint*.

Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, and 1 others. 2024. A survey on data selection for language models. *arXiv preprint*.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *ICML*.

David Berthelot, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. 2020. Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring. In *ICLR*.

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. 2019. Mixmatch: A holistic approach to semi-supervised learning. *NeurIPS*.

Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, Ilya Sutskever, and Jeffrey Wu. 2024. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. In *ICML*.

Luigi Carratino, Moustapha Cissé, Rodolphe Jenatton, and Jean-Philippe Vert. 2022. On mixup regularization. *JMLR*.

Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. 2009. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*.

Jiaao Chen, Zichao Yang, and Diyi Yang. 2020. Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification. In *ACL*.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. 2024a. Alpagasus: Training a better alpaca with fewer data. In *ICLR*.

Xuxi Chen, Zhendong Wang, Daouda Sow, Junjie Yang, Tianlong Chen, Yingbin Liang, Mingyuan Zhou, and Zhangyang Wang. 2024b. Take the bull by the horns: Hard sample-reweighted continual training improves llm generalization. *arXiv preprint*.

Zeming Chen, Alejandro Hernández Cano, Angelika Romanou, Antoine Bonnet, Kyle Matoba, Francesco Salvi, Matteo Pagliardini, Simin Fan, Andreas Köpf, Amirkeivan Mohtashami, and 1 others. 2023a. Meditron-70b: Scaling medical pretraining for large language models. *arXiv preprint*.

Zhihong Chen, Shuo Yan, Juhao Liang, Feng Jiang, Xiangbo Wu, Fei Yu, Guiming Hardy Chen, Junying Chen, Hongbo Zhang, Li Jianquan, and 1 others. 2023b. Multilingualsift: Multilingual supervised instruction fine-tuning.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Muthu Chidambaram, Xiang Wang, Yuzheng Hu, Chenwei Wu, and Rong Ge. 2022. Towards understanding the data dependency of mixup-style training. In *ICLR*.

Everlyn Chimoto, Jay Gala, Orevaoghene Ahia, Julia Kreutzer, Bruce Bassett, and Sara Hooker. 2024. Critical learning periods: Leveraging early training dynamics for efficient data pruning. In *ACL Findings*.

Hyeong Kyu Choi, Joonmyung Choi, and Hyunwoo J. Kim. 2022. Tokenmixup: Efficient attention-guided token-level data augmentation for transformers. In *NeurIPS*.

Fenia Christopoulou, Gerasimos Lampouras, and Ignacio Iacobacci. 2022. Training dynamics for curriculum learning: A study on monolingual and cross-lingual nlu. In *EMNLP*.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *EMNLP*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint*.

Yann Dubois, Percy Liang, and Tatsunori Hashimoto. 2024. Length-controlled alpacaeval: A simple debiasing of automatic evaluators. In *COLM*.

Gamaleldin F Elsayed, Dilip Krishnan, Hossein Mobahi, Kevin Regan, and Samy Bengio. 2018. Large margin deep networks for classification. In *NeurIPS*.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. Model alignment as prospect theoretic optimization. In *ICML*.

Yanjun Fu, Faisal Hamman, and Sanghamitra Dutta. 2025. T-shirt: Token-selective hierarchical data selection for instruction tuning. In *NeurIPS*.

Demi Guo, Yoon Kim, and Alexander M Rush. 2020. Sequence-level mixed sample data augmentation. In *EMNLP*.

Tianyu Han, Lisa C Adams, Jens-Michalis Papaioannou, Paul Grundmann, Tom Oberhauser, Alexander Löser, Daniel Truhn, and Keno K Bressem. 2023. Medalpaca–an open-source collection of medical conversational ai models and training data. *arXiv preprint*.

Zongbo Han, Yifeng Yang, Changqing Zhang, Linjun Zhang, Joey Tianyi Zhou, and Qinghua Hu. 2024. Selective learning: Towards robust calibration with dynamic regularization. *arXiv preprint*.

Muyang He, Shuo Yang, Tiejun Huang, and Bo Zhao. 2024. Large-scale dataset pruning with dynamic uncertainty. In *CVPR*.

Dan Hendrycks, Norman Mu, Ekin Dogus Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. 2020. Augmix: A simple method to improve robustness and uncertainty under data shift. In *ICLR*.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *ICLR*.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, and 1 others. 2024. Qwen2. 5-coder technical report. *arXiv preprint*.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint*.

Neel Jain, Ping-yeh Chiang, Yuxin Wen, John Kirchenbauer, Hong-Min Chu, Gowthami Somepalli, Brian R Bartoldson, Bhavya Kailkhura, Avi Schwarzschild, Aniruddha Saha, and 1 others. 2024. Neftune: Noisy embeddings improve instruction finetuning. In *ICLR*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, and 1 others. 2023. Mistral 7b. *arXiv preprint*.

Yiding Jiang, Dilip Krishnan, Hossein Mobahi, and Samy Bengio. 2018. Predicting the generalization gap in deep networks with margin distributions. In *ICLR*.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. Pubmedqa: A dataset for biomedical research question answering. In *EMNLP*.

Po-Nien Kung, Fan Yin, Di Wu, Kai-Wei Chang, and Nanyun Peng. 2023. Active instruction tuning: Improving cross-task generalization by training on prompt sensitive tasks. In *EMNLP*.

Yanis Labrak, Adrien Bazoge, Emmanuel Morin, Pierre-Antoine Gourraud, Mickael Rouvier, and Richard Dufour. 2024. BioMistral: A collection of open-source pretrained large language models for medical domains. In *ACL Findings*.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, and 1 others. 2024. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint*.

Changchun Li, Ximing Li, Lei Feng, and Jihong Ouyang. 2022. Who is your right mixup partner in positive and unlabeled learning. In *ICLR*.

Junnan Li, Richard Socher, and Steven C.H. Hoi. 2020. Dividemix: Learning with noisy labels as semi-supervised learning. In *ICLR*.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, yelong shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, and Weizhu Chen. 2024. Not all tokens are what you need for pretraining. In *NeurIPS*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint*.

Yizhen Luo, Jiahuan Zhang, Siqi Fan, Kai Yang, Yushuai Wu, Mu Qiao, and Zaiqing Nie. 2023. Biomedgpt: Open multimodal generative pre-trained transformer for biomedicine. *arXiv preprint*.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint*.

Andrei Mircea, Supriyo Chakraborty, Nima Chitsazan, Milind Naphade, Sambit Sahu, Irina Rish, and Ekaterina Lobacheva. 2025. Training dynamics underlying language model scaling laws: Loss deceleration and zero-sum learning. In *ACL*.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. In *NeurIPS*.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. 2022. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *CHIL*.

Chanwoo Park, Sangdoo Yun, and Sanghyuk Chun. 2022. A unified analysis of mixed sample data augmentation: A loss function perspective. In *NeurIPS*.

Seo Yeon Park and Cornelia Caragea. 2022. A data cartography based mixup for pre-trained language models. In *NAACL*.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint*.

Francesco Pinto, Harry Yang, Ser Nam Lim, Philip Torr, and Puneet Dokania. 2022. Using mixup as a regularizer can surprisingly improve accuracy & out-of-distribution robustness. *NeurIPS*.

Eduard Poesina, Cornelia Caragea, and Radu Ionescu. 2024. A novel cartography-based curriculum learning method applied on RoNLI: The first Romanian natural language inference corpus. In *ACL*.

Zhenting Qi, Fan Nie, Alexandre Alahi, James Zou, Himabindu Lakkaraju, Yilun Du, Eric Xing, Sham Kakade, and Hanlin Zhang. 2025. Evolm: In search of lost language model training dynamics. In *NeurIPS*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*.

Jun Rao, Xuebo Liu, Lian Lian, Shengjun Cheng, Yunjie Liao, and Min Zhang. 2024. Commonit: Commonality-aware instruction tuning for large language models via data partitions. In *EMNLP*.

Stephanie Schoch, Ritwick Mishra, and Yangfeng Ji. 2023. Data selection for fine-tuning large language models using transferred shapley values. In *ACL Workshop*.

Nabeel Seedat, Nicolas Huynh, Boris van Breugel, and Mihaela van der Schaar. 2024. Curated LLM: Synergy of LLMs and data curation for tabular augmentation in low-data regimes. In *ICML*.

Zhengyan Shi, Adam X. Yang, Bin Wu, Laurence Aitchison, Emine Yilmaz, and Aldo Lipani. 2024. Instruction tuning with loss over instructions. In *NeurIPS*.

Shivalika Singh, Freddie Vargus, Daniel D'souza, Börje Karlsson, Abinaya Mahendiran, Wei-Yin Ko, Herumb Shandilya, Jay Patel, Deividas Mataciunas, Laura O'Mahony, and 1 others. 2024. Aya dataset: An open-access collection for multilingual instruction tuning. In *ACL*.

Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. 2020. Fixmatch: simplifying semi-supervised learning with consistency and confidence. In *NeurIPS*.

Lichao Sun, Congying Xia, Wenpeng Yin, Tingting Liang, S Yu Philip, and Lifang He. 2020. Mixup-transformer: Dynamic data augmentation for nlp tasks. In *COLING*.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *EMNLP*.

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.

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

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, and 1 others. 2023. Zephyr: Direct distillation of lm alignment. *arXiv preprint*.

A F M Shahab Uddin, Mst. Sirazam Monira, Wheemyung Shin, TaeChoong Chung, and Sung-Ho Bae. 2021. Saliencymix: A saliency guided data augmentation strategy for better regularization. In *ICLR*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *JMLR*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. 2019. Manifold mixup: Better representations by interpolating hidden states. In *ICML*.

Jiahao Wang, Bolin Zhang, Qianlong Du, Jiajun Zhang, and Dianhui Chu. 2024. A survey on data selection for llm instruction tuning. *arXiv preprint*.

Jingtan Wang, Xiaoqiang Lin, Rui Qiao, Pang Wei Koh, Chuan-Sheng Foo, and Bryan Kian Hsiang Low. 2025. NICE data selection for instruction tuning in LLMs with non-differentiable evaluation metric. In *ICML*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *ACL*.

Chaoyi Wu, Weixiong Lin, Xiaoman Zhang, Ya Zhang, Weidi Xie, and Yanfeng Wang. 2024. Pmc-llama: toward building open-source language models for medicine. *JAMIA*.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. 2024. LESS: Selecting influential data for targeted instruction tuning. In *ICML*.

Sang Michael Xie, Shibani Santurkar, Tengyu Ma, and Percy Liang. 2023. Data selection for language models via importance resampling. In *NeurIPS*.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. 2024. Wizardlm: Empowering large pre-trained language models to follow complex instructions. In *ICLR*.

Huiyun Yang, Huadong Chen, Hao Zhou, and Lei Li. 2022. Enhancing cross-lingual transfer by manifold mixup. In *ICLR*.

Yuming Yang, Yang Nan, Junjie Ye, Shihan Dou, Xiao Wang, Shuo Li, Huijie Lv, Mingqi Wu, Tao Gui, Qi Zhang, and 1 others. 2025. Measuring data diversity for instruction tuning: A systematic analysis and a reliable metric. In *ACL*.

Yongcheng Zeng, Guoqing Liu, Weiyu Ma, Ning Yang, Haifeng Zhang, and Jun Wang. 2024. Token-level direct preference optimization. In *ICML*.

Dylan Zhang, Qirun Dai, and Hao Peng. 2025a. The best instruction-tuning data are those that fit. In *NeurIPS*.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *ICLR*.

Linjun Zhang, Zhun Deng, Kenji Kawaguchi, Amirata Ghorbani, and James Zou. 2021. How does mixup help with robustness and generalization? In *ICLR*.

Mike Zhang and Barbara Plank. 2021. Cartography active learning. In *EMNLP Findings*.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and 1 others. 2023. Instruction tuning for large language models: A survey. *arXiv preprint*.

Shujian Zhang, Chengyue Gong, Xingchao Liu, Pengcheng He, Weizhu Chen, and Mingyuan Zhou. 2022. Allsh: Active learning guided by local sensitivity and hardness. In *NAACL Findings*.

Wancong Zhang and Ieshan Vaidya. 2021. Mixup training leads to reduced overfitting and improved calibration for the transformer architecture. *arXiv preprint*.

Yedi Zhang, Aaditya K Singh, Peter E. Latham, and Andrew M Saxe. 2025b. Training dynamics of in-context learning in linear attention. In *ICML*.

Hao Zhao, Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. 2024. Long is more for alignment: A simple but tough-to-beat baseline for instruction fine-tuning. In *ICML*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, and 1 others. 2023. A survey of large language models. *arXiv preprint*.

Yang Zhao, Li Du, Xiao Ding, Yangou Ouyang, Hepeng Wang, Kai Xiong, Jinglong Gao, Zhouhao Sun, Dongliang Xu, Yang Qing, and 1 others. 2025. Beyond similarity: A gradient-based graph method for instruction tuning data selection. In *ACL*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. In *NeurIPS*.

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. In *NeurIPS*.

A Additional Experiment Results

A.1 Hyperparameter Search

In SFTMix, we use the hyperparameter μ to control the regularization effect in the training loss ℓ_{SFTMix} and α to control the sampling distribution of λ. To explore their impacts, we experiment with $\mu \in \{0.05, 0.1, 0.2, 0.5, 1\}$ and $\alpha \in \{0.1, 0.2, 0.5, 0.7, 1\}$ by instruction-tuning Llama on Alpaca-52K and UltraChat-200K. As shown in Table 9, $\mu = 0.2$ achieves the highest performance across most metrics. Similarly, Table 10 shows that $\alpha = 0.5$ yields the best overall score on MT-Bench and the highest LC WR on AlpacaEval-2. Hence, we set $\mu = 0.2$ and $\alpha = 0.5$ in §4 and §5. We also observe that performance remains stable for μ between 0.1 and 0.2 and α between 0.2 and 0.5. However, performance declines when α approaches 0.1 or 1, where the Beta distribution becomes too flat and the interpolated examples deviate excessively from either endpoint.

In SFTMix, we apply Mixup regularization by interpolating between the most and least confident halves of the dataset. We adopt an equal split for its simplicity and ease of implementation, avoiding additional hyperparameters. To examine this design choice, we also fine-tune Llama-3.1-8B using Mixup between the most and least confident thirds. As shown in Table 11, the equal split continues to yield the largest improvement over NTP. These results suggest that equal partitioning serves as a strong default, while more adaptive strategies remain a promising direction for future work.

μ	**MT-Bench**			**AlpacaEval-2**	
	ST	**MT**	**Overall**	**WR**	**LC WR**
Dataset: Alpaca-52K					
0.05	5.0030	3.9623	4.4827	4.7100	9.9138
0.1	5.0600	4.0238	4.5419	4.7715	10.0172
0.2	**5.2125**	3.9525	**4.5825**	**4.9031**	**10.3195**
0.5	4.9606	3.8968	4.4287	4.5092	9.5034
1	4.7050	**4.1075**	4.4062	3.9450	8.2856
Dataset: UltraChat-200K					
0.05	6.1764	4.9208	5.5486	5.0433	8.5521
0.1	**6.2988**	5.2713	5.7851	5.0823	9.1174
0.2	6.2750	**5.3500**	**5.8125**	**5.1149**	**9.3810**
0.5	5.9701	4.5899	5.2800	4.8991	8.6712
1	5.6621	4.2490	4.9556	4.5661	8.2908

Table 9: Hyperparameter search for μ. We set $\mu = \underline{0.2}$ as the default in SFTMix, as it achieves the highest performance across most metrics.

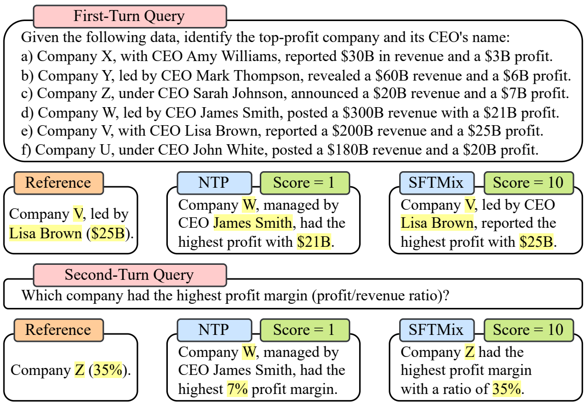
α	**MT-Bench**			**AlpacaEval-2**	
	ST	**MT**	**Overall**	**WR**	**LC WR**
Dataset: Alpaca-52K					
0.1	5.1632	3.8991	4.5312	4.8813	10.0259
0.2	**5.2237**	3.9103	4.5670	**4.9112**	10.1865
0.5	5.2125	**3.9525**	**4.5825**	4.9031	**10.3195**
0.7	5.1209	3.9291	4.5250	4.7912	9.8247
1	5.0826	3.9200	4.5013	4.7526	9.9835
Dataset: UltraChat-200K					
0.1	5.7461	4.9800	5.3631	4.9001	8.5527
0.2	6.2001	**5.4137**	5.8069	5.0221	9.1846
0.5	**6.2750**	5.3500	**5.8125**	**5.1149**	**9.3810**
0.7	6.0138	5.0012	5.5075	5.0811	8.9312
1	5.9112	4.8641	5.3877	4.8813	8.7512

Table 10: Hyperparameter search for α. We set $\mu = \underline{0.5}$ as the default in SFTMix, as it achieves the highest performance across most metrics.

A.2 Human Evaluation on AlpacaEval-2

To complement the LLM-as-a-judge evaluation in §4.1, we conduct a human evaluation following the setup in (Zhao et al., 2024). Specifically, we use the 80 instructions from the Vicuna subset in AlpacaEval-2 and compare responses generated by Llama, instruction-tuned on Alpaca-52K using either NTP or SFTMix, in a head-to-head fashion. As in (Zhao et al., 2024), we instruct evaluators to disregard response length in their judgments.

We collect 200 human preference judgments, where Llama instruction-tuned with SFTMix wins 42.5% of the time, NTP wins 26.5%, and the remaining 31% are ties. This result aligns with our observation in §4.1 that SFTMix outperforms NTP in instruction-following SFT tasks.



Given the following data, identify the top-profit company and its CEO's name:
a) Company X, with CEO Amy Williams, reported $30B in revenue and a $3B profit.
b) Company Y, led by CEO Mark Thompson, revealed a $60B revenue and a $6B profit.
c) Company Z, under CEO Sarah Johnson, announced a $20B revenue and a $7B profit.
d) Company W, led by CEO James Smith, posted a $300B revenue with a $21B profit.
e) Company V, with CEO Lisa Brown, reported a $200B revenue and a $25B profit.
f) Company U, under CEO John White, posted a $180B revenue and a $20B profit.

Company V, led by Lisa Brown ($25B).

Company W, managed by CEO James Smith, had the highest profit with $21B.

Company V, led by CEO Lisa Brown, reported the highest profit with $25B.

Which company had the highest profit margin (profit/revenue ratio)?

Company Z (35%).

Company W, managed by CEO James Smith, had the highest 7% profit margin.

Company Z had the highest profit margin with a ratio of 35%.

Figure 4: A qualitative example from the extraction category in MT-Bench. Compared to its NTP-tuned counterpart, Llama instruction-tuned by SFTMix accurately interprets the queries from both turns and correctly extracts the relevant information from the prompt.

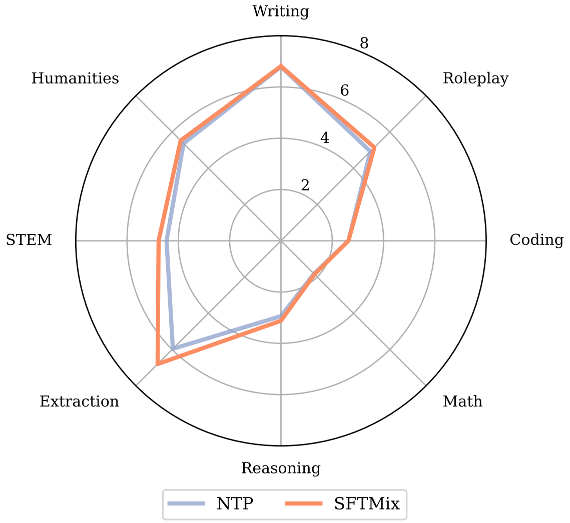


Figure 5: Per-category MT-Bench scores for Llama-3.1-8B tuned on Alpaca-52K. SFTMix consistently outperforms NTP, notably in extraction, STEM, and roleplay.

A.3 MT-Bench Result Analysis

Figure 4 presents a test example from the extraction category in MT-Bench, showing responses generated by Llama-3.1-8B instruction-tuned on Alpaca-52K using either NTP or SFTMix. In this example, the LLM trained with SFTMix accurately interprets the instructions from both the first- and second-turn queries, correctly extracting the relevant information from the prompt. Notably, it succeeds in answering the second-turn query, which involves calculating the profit margin and performing a ratio comparison. In contrast, the LLM trained with NTP struggles to differentiate between revenue and profit, leading to incorrect responses in both turns.

Recipe	Mixup	MT-Bench			AlpacaEval-2	
		ST	MT	Overall	WR	LC WR
NTP	-	4.9100	3.8150	4.3625	4.0714	8.6528
SFTMix	Half	**5.2125**	3.9525	**4.5825**	**4.9031**	**10.3195**
SFTMix	Third	5.1450	**3.9850**	4.5650	4.2073	8.8978

Table 11: Evaluation of the Mixup ratio in SFTMix. By default, SFTMix applies Mixup between the most and least confident halves, yielding greater improvement.

Language	Recipe	Win	Tie	Loss
Chinese	NTP	30.4%	35.6%	34.0%
	SFTMix	**38.8%**	33.2%	28.0%
French	NTP	34.5%	24.5%	41.0%
	SFTMix	**39.0%**	22.5%	38.5%

Table 12: Multilingual evaluation of SFTMix. The method consistently improves win rates in both Chinese and French instruction tuning.

To provide finer-grained insights, we report per-category MT-Bench scores for Llama-3.1-8B instruction-tuned on Alpaca-52K. As shown in Figure 5, SFTMix consistently improves over NTP across all categories, with the largest gains in extraction, STEM, and roleplay. However, both methods remain limited in coding, math, and reasoning, likely due to the lack of such data in Alpaca-52K.

A.4 Multilingual Instruction Tuning

To assess SFTMix in multilingual settings, we fine-tune Llama-3.1-8B on the Chinese (Peng et al., 2023) and French (Chen et al., 2023b) variants of Alpaca-52K, following the setup in §4.1. We then compare the resulting models with human-annotated baselines from Aya (Singh et al., 2024) in the respective languages, using GPT-4o (Hurst et al., 2024) as the judge. As shown in Table 12, SFTMix increases the win rate in both languages, demonstrating its cross-lingual generalizability.