DEM: Distribution Edited Model for Training with Mixed Data Distributions

Dhananjay Ram ♠ **Aditya Rawal** ♠ **Momchil Hardalov** ♣
Nikolaos Pappas ♣ **Sheng Zha** ♠
♠AGI Foundations, Amazon ♣AWS AI Labs
{radhna, adirawal, momchilh, nppappa, zhasheng}@amazon.com

Abstract

Training with mixed data distributions is a common and important part of creating multi-task and instruction-following models. The diversity of the data distributions and cost of joint training makes the optimization procedure extremely challenging. Data mixing methods partially address this problem, albeit having a sub-optimal performance across data sources and require multiple expensive training runs. In this paper, we propose a simple and efficient alternative for better optimization of the data sources by combining models individually trained on each data source with the base model using basic element-wise vector operations. The resulting model, namely *Distribution Edited Model (DEM)*, is 11× cheaper than standard data mixing and outperforms strong baselines on a variety of benchmarks, yielding upto 6.2% improvement on MMLU, 11.5% on BBH, 16.1% on DROP, 6% on MathQA, and 9.3% on HELM with models of size 3B to 13B. Notably, DEM does not require full re-training when modifying a single data-source, thus making it very flexible and scalable for training with diverse data sources.

1 Introduction

Large Language Models (LLM) go through an extensive pretraining on billions or trillions of tokens (Brown et al., 2020; Zhang et al., 2022; Raffel et al., 2020; Touvron et al., 2023a,b; Geng and Liu, 2023), but they typically require supervised fine-tuning on diverse instruction-following datasets for properly following human instructions (Ouyang et al., 2022; Sanh et al., 2022; Iyer et al., 2022; Chung et al., 2024). Supervised training is crucial for ensuring that generated outputs meet user expectations and perform well on downstream tasks (Radford et al., 2019; Gururangan et al., 2020).

The datasets for supervised training are often of different sizes and follow different distributions. Recent state-of-the-art fine-tuning approaches (Iyer et al., 2022; Chung et al., 2024) demonstrate that training on multiple data distributions requires careful tuning of the mixing weights for each data source to capture the combined distribution and improve downstream task performance. Tuning these weights in a data-mixing approach is a computationally expensive process. Although, there are techniques to speed-up the search (Xie et al., 2023; Albalak et al., 2023), the process remains time-consuming. Moreover, when one or more new datasets are introduced, the weights for each dataset need to be re-tuned. This requirement makes the data-mixing approach inflexible and hard to maintain in a production environment.

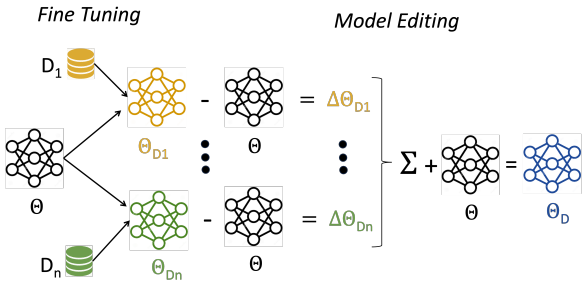

Figure 1: The *Distribution Edited Model* (Θ_D) results from fine-tuning a pretrained model (Θ) on n individual data distributions (D_i) and combining the resulting models with basic element-wise vector operations. Here, the combination is achieved by extracting *distribution vectors* ($\Delta\Theta_{D_i}$), multiplying them by weight coefficients (ω_i), and adding their weighted sum to the base model.

To address these challenges when fine-tuning an LLM on a set of diverse data distributions, we propose a simple and efficient approach that combines individually trained versions of the base model using element-wise vector operations. Our method focuses on the challenging setting of combining diverse data distributions that correspond to multiple tasks from different domains such as math, reasoning, conversations and coding. In particular, our goal is to better capture a diverse data distributions

as opposed to editing the model on a single downstream task (Ilharco et al., 2022a; Schumann et al., 2024). Hence, we call resulting model *Distribution Edited Model* (*DEM*, shown in Figure 1). Our experiments on a variety of downstream tasks show that *DEM* is an effective, highly capable and low cost alternative to the models trained using data mixing methods.

The primary benefit of the proposed approach is its ability to efficiently identify the optimal combination of data sources for training a model. Instead of exhaustively training and validating on all possible combinations of data sources, which can be computationally expensive, we take a more streamlined approach. First, we finetune the original model on each individual data source independently with early stopping to obtain the optimal model. Second, we extract source distribution vectors by subtracting the original model from the finetuned ones. Lastly, we create the final model by adding a weighted combination of these distribution vectors to the base model, allowing it to capture the joint distribution of different data sources in a controlled manner while enabling incremental updates with new datasets.
Our contributions can be summarized as follows:

- We propose a simple and efficient approach for training models on diverse data distributions that offers a flexible way for tuning the contributions of each data source individually without the need of full data re-training (Section 4).
- We show that *DEM* reduces the training cost by $11\times$ while improving the model performance. Compared to standard *data mixing* approaches, DEM yields up to 6.2% improvement on MMLU, 11.5% on BBH, 16.1% on DROP, 6% on MathQA and 9.3% on HELM with 3B, 7B, and 13B models.
- We perform an exhaustive analysis of the properties of the distribution vectors and their corresponding models, finding that DEM is better aligned with the individual models than baseline while remaining close to the original model.

2 Related Work

Multi-task Fine Tuning Instruction-based multi-task fine-tuning of language models has been previously shown to improve both zero and few-shot performance on unseen tasks (Wei et al., 2022a; Sanh et al., 2022). Instruction-tuning data can be sourced from diverse task categories (such as math, reasoning, dialog etc), and the model performance is often sensitive to the data-mixing strategy. For example, both (Chung et al., 2024) and (Iyer et al., 2022) carefully tune the data-mixing weights for various training data sources.

Hyperparameter tuning of data-mixing weights is a compute intensive process, and methods such as DoReMi (Xie et al., 2023) and Online Data Mixing (Albalak et al., 2023) have been proposed to speed-up the process for pretrained data-mixing either through a proxy-model training or through a multi-armed bandit approach respectively. Renduchintala et al. (2024) used a submodular function to assign importance scores to tasks which are then used to determine the mixture weights. Li et al. (2024) built a framework to find multiple diverse solutions in the Pareto front of many objectives. In this work, we propose an alternative strategy for training with multiple data sources by using vector arithmetic to combine models fine-tuned on individual datasets, rather than mixing training data in specific proportions.

Model Weight Interpolation Recently, model weight interpolation and task arithmetic techniques have been shown to improve the performance of pre-trained models on: single-task (Izmailov et al., 2018; Matena and Raffel, 2022; Yüce et al., 2022; Wortsman et al., 2022b) and multi-task (Ilharco et al., 2022b,a; Li et al., 2022; Wortsman et al., 2022a; Yadav et al., 2023; Daheim et al., 2024), out-of-domain generalization (Arpit et al., 2022; Rame et al., 2022; Jin et al., 2023; Ramé et al., 2023; Cha et al., 2024), and federated learning (McMahan et al., 2017; Li et al., 2020).

Going beyond simple weight averaging, (Matena and Raffel, 2022) explored merging using Fisher-weighted averaging for improving single-task model performance by leveraging other auxiliary tasks. Ilharco et al. (2022a) presented a model merging technique based on producing task vectors and performing arithmetic operations, such as addition, subtraction to obtain a multitask checkpoint and 'forget' unwanted behavior. Daheim et al. (2024) proposed a new uncertainty-based correction of the task vector coefficients to improve the performance by reducing the model mismatch.

While previous work focused on classification tasks in NLP or vision, we extend vector-arithmetic-

based model editing to multi-task fine-tuning on diverse data distributions. Our results show that the proposed approach outperforms and is more efficient than data-mixed fine-tuning.

3 Background: Data mixing

Let us consider a pretrained language model with parameters Θ, and $D_1, D_2, ..., D_n$ denote n different supervised fine tuning datasets. Each dataset can consist of a single or multiple tasks. The exact tasks may have an overlap between these datasets, however, the corresponding samples are unique to each dataset. Standard *data mixing* (Chung et al., 2024; Iyer et al., 2022) methods create training batches by performing a weighted sampling from each of the training datasets D_i. The goal is to learn a joint data distribution that can span all training datasets.

4 Proposed Approach: Distribution Edited Model (DEM)

In contrast to standard data mixing, we propose to learn each data distribution separately and combine them post training. In the following subsections, we present two variants of that lead to a *Distribution Edited Model* that achieves this goal.

4.1 Combined Distribution Vectors

Let us assume a set of training data sources $(D_1, D_2, ..., D_n)$. First, we fine tune our pretrained model (Θ) on each of these n datasets separately, with a different set of hyper-parameters (chosen for optimal validation loss). The corresponding fine-tuned models are noted as $\Theta_{D_1}, \Theta_{D_2}, ..., \Theta_{D_n}$. Next, we define a data distribution vector (DV) $\Delta\Theta_{D_i}$ (corresponding to the dataset D_i) as the element-wise difference of parameters between the pretrained model Θ and a fine-tuned model Θ_{D_i}, following a similar approach as presented in (Ilharco et al., 2022a).

$$\Delta\Theta_{D_i} = \Theta_{D_i} - \Theta, \tag{1}$$

Instead of task specific model editing, as in prior work, we focus on a mixture of large number of diverse NLP downstream tasks. These different tasks are represented with their own data distribution and we investigate how to combine different data DVs that we can extract by fine tuning the pretrained model using data from different distributions.

Lastly, we obtain a mixed data DV by computing a weighted combination of each $\Delta\Theta_{D_i}$ with corresponding weights $\omega_i \in \mathbb{R}$. Finally we add the pretrained model Θ to obtain our *Distribution Edited Model (DEM)* as follows:

$$\Theta_D = \Theta + \sum_{i=1}^{n} \omega_i \Delta\Theta_{D_i}. \tag{2}$$

4.2 Model Interpolation

Another way to combine the finetuned models (Θ_{D_i}) is through model weight interpolation. In this case, we do not extract data distribution vectors $(\Delta\Theta_{D_i})$, but rather use the finetuned models directly that capture information about the data distribution. Specifically, we take a weighted average of all the fine tuned models (Θ_{D_i}) where the weights, $\omega_i \in \mathbb{R}$ sum to 1. More formally,

$$\Theta_D = \sum_{i=1}^{n} \omega_i \Theta_{D_i}; \quad s.t. \quad \sum_{i=1}^{n} \omega_i = 1. \tag{3}$$

Note that, Eq 3 is a special case of Eq 2 when $\sum_{i=1}^{n} \omega_i = 1$. *DEM* using distribution vector (Eq 2) provides more flexibility in terms of choosing ω_i per data source which can yield further performance improvement (Section 6.1).

4.3 Computational Cost

To better understand the advantages of *DEM* over the *data mixing* approach we derive a formula to measure the cost for each method. Let us assume we have n different data sources and m number of weights per data source. The hyperparameter search space for both the approaches will have a total of m^n weight combinations. Intuitively, searching for *data mixing* weights is comparatively more expensive than *DEM* since full data re-training is required for validating each weight combination. On the other hand, *DEM* requires finding the optimal weights after individual training using only validation for each weight combination.

To formalize this, assume T and V as the average number training and validation steps respectively. The computational complexity for the weighted *data mixing* will be $O(m^n(T+V))$ and for the proposed *DEM* approach will be $O(n(T+V) + (m^n V))$. We can clearly see that $O(m^n(T+V)) \geq O(n(T+V) + (m^n V))$, and with *DEM* we reduce the number of training run by a factor of m^n/n.

Additionally, we can compare the exact training and validation cost of our proposed *DEM* approach with the baseline. Assuming k steps of training or

validation and each step takes t seconds, we can define the cost (c) in gpu-hours as follows:

$$c = (k * t * g)/3600 \quad (4)$$

where g is the total number of GPUs used. The exact cost for training (c_{train}) and validation (c_{val}) depends on the corresponding values of k, t and g and generally $c_{train} \gg c_{val}$.

5 Experimental Setup

5.1 Dataset

Here, we list the fine-tuning datasets, we use to enhance instruction following capability of our base pre-trained LLM. Previous work has shown that they improve the instruction following capabilities of the models (Chung et al., 2024; Iyer et al., 2022; Gupta et al., 2022; Amini et al., 2019; Sanh et al., 2022).

- **Chain of Thoughts (CoT) (Wei et al., 2022b)**: The CoT mixture (Chung et al., 2024) consists of nine datasets with manually written CoT annotations. Each task in these nine datasets have ten manually composed instruction templates, and the span arithmetic reasoning, multi-hop reasoning, and natural language inference.

- **Math QA (Amini et al., 2019)**: This dataset consists of 37K math-based multiple-choice word problems. The problem set includes geometry, counting, probability, physics, gain-loss and other general math topics.

- **Public Pool of Prompts (P3) (Sanh et al., 2022)**: P3 (Public Pool of Prompts) is a collection of prompted English datasets for a diverse set of NLP tasks, where each sample consists of a prompted input and a target text. Prompts can be considered as a functions that map an example from a dataset to a natural language input and target output. Promptsource (Bach et al., 2022) is used to interactively create prompts and gather prompt-specific metadata like evaluation metrics. As of writing of this paper, over 2,000 prompts from 270+ datasets are publicly available on Promptsource.

- **Instruct Dial (InstDial) (Gupta et al., 2022)**: This is an instruction tuning dataset designed for dialogues. It consists of 48 different dialogue tasks from 59 open dialogue datasets which is unified in text-to-text format suitable for decoder LLMs. It has been shown improve model performance in unseen datasets, specially for dialogue related tasks.

- **Super Natural Instructions (SNI) (Wang et al., 2022)**: This dataset consists of 1,616 diverse NLP tasks in text-to-text format with instructions written by experts. It covers 76 distinct task types, including but not limited to text composition, infilling, extraction, classification, sequence tagging and paraphrasing.

# Params	Context	Dims	# Heads	# Layers
3B	2048	3200	32	26
7B	2048	4096	32	40
13B	2048	5120	40	40

Table 1: Characteristics of different OpenLLaMA model sizes used in our experiments.

5.2 Model Architecture

We use OpenLLaMA (Geng and Liu, 2023) as our base LLM, which is trained on 1T tokens from the RedPajama Dataset (Computer, 2023). It follows the same architecture as the LLaMA model (Touvron et al., 2023a) – a decoder-only LLM with rotary positional embedding, SwiGLU activations and RMS Norm for pre-normalization. In our experiments, we cover three different sized models: 3B, 7B and 13B (see Table 1). We carry all ablations with the 7B model, while the 3B and 13B models are used to show generalization of the proposed approach to other sizes. The experimental results show that the properties of *DEM* are present across different model sizes.

5.3 Training

We fine-tune the OpenLLaMA model on all instruction following datasets (Section 5.1), both separately and jointly. We use AdamW optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay of 0.05, gradient clipping of 1 and a constant learning rate of 2e-5 with a 2000 step warmup. We also adjust batch size for different datasets based on the validation loss (see Appendix A for details). We use a greedy sample packing approach to fit multiple training samples into a single batch sample efficiently, padding to the max sequence length without overflowing into the next sample of a batch. To select the optimal mixing weights for *DEM* (Eq 2), we perform a grid search over ω_i values. For each coefficient combination we evaluate the validation

losses on the five datasets (Section 5.1), and select the model that minimizes their average (see Section C for details). We use an equal weight of $\omega_i = 0.25$ for all datasets in our experiments.

5.4 Evaluation Framework

We evaluate the instruction following capability of the models using three publicly available benchmarks, namely InstructEval (Chia et al., 2024), LM-evaluation-harness (Gao et al., 2024) and HELM (Liang et al., 2023). To have a holistic evaluation, we choose a diverse set of held-out datasets: (*i*) from InstructEval – MMLU, Big-Bench Hard and DROP, (*ii*) from LM-evaluation-harness – MathQA, and (*iii*) from HELM – twenty sets from six diverse task-groups – Classification, ClosedbookQA, OpenbookQA, Math, Reasoning and Conversational (see Table 3). We perform 5-shot evaluation on MMLU and HELM, and 3-shot evaluation on BBH and DROP, inline with the standardized setup and previous work.

5.5 Baseline Models

The pre-trained OpenLLaMA serves as the non instruction-tuned baseline for evaluation. Our primary instruction-tuned baseline is *data mixing* – the model is fine-tuned using a weighted mixture of 5 diverse datasets as described in 5.1 following (Chung et al., 2024; Iyer et al., 2022) which has been shown to produce SOTA performance with large scale diverse datasets. This model requires finding the optimal weights corresponding to each training dataset such that the validation loss reaches optimal value for each dataset at similar number of training steps. We experimented with several combinations of weights and chose the one that leads to the smallest validation loss (see Appendix B for details). Additionally, we create a simpler baseline where we concatenate all 5 training datasets and the samples are shuffled randomly during training. This technique is more cost-effective than the standard data mixing approach, as it does not require any weight optimization.

6 Experimental Analysis

6.1 Downstream Task Performance

In this section, we first use the Instruct-Eval framework to evaluate the performance of both the pre-trained and fine-tuned models. The performance on MMLU, BBH and DROP is shown in Table 2.[1] In addition to the pretrained OpenLLaMA model, we show the performance of LLaMA (Touvron et al., 2023a) and LLaMA2 (Touvron et al., 2023b) of same size as a reference. We also include three other supervised fine tuned models of larger sizes, namely FlanPaLM (8B) and OPT-IML (30, 175B).

Models	MMLU	BBH	DROP	MathQA
Open LLaMA	40.31	32.84	24.38	27.71
LLaMA (Touvron et al., 2023a)	35.10	30.30	-	-
LLaMA2 (Touvron et al., 2023b)	45.30	32.60	-	-
FlanPaLM (8B) (Chung et al., 2024)	49.3	36.4	-	-
OPT-IML (30B) (Iyer et al., 2022)	43.2	30.9	-	-
OPT-IML (175B) (Iyer et al., 2022)	47.1	35.7	-	-
CoT	41.67	33.98	24.20	29.31
Math QA	39.71	32.70	24.31	25.03
P3	35.69	14.00	23.29	27.14
InstDial	39.31	23.09	21.81	26.60
SNI	46.55	35.88	34.53	28.31
Data Mixing	47.77	36.38	32.71	30.35
Concatenated Datasets	43.43	21.34	23.21	27.71
DEM - Interpolation (Ours)	50.14	40.11	36.31	31.22
DEM - Distribution Vector (Ours)	**50.74**	**40.56**	**37.96**	**32.16**

Table 2: Downstream task performance of models trained on different instruction following datasets (Section 5.1). We compare it with different pretrained and fine-tuned baselines (Section 5.5) and our proposed approach in Section 4. The models are of size 7B, unless specified. The performance numbers for models with citation are taken from the corresponding paper, rest are evaluated using InstructEval and LM-evaluation-harness.

We present the performance of fine tuned models on each dataset separately and observe that the performance degrades compared to Open-LLaMA model for P3, InstDial and MathQA. On the other hand, we observe significant improvement with CoT and SNI datasets in all four task families. We compare these models with *data mixing* baseline and note that it performs significantly better than the pretrained OpenLLaMA, while the improvement compared to the best single dataset fine-tuned model (i.e. SNI) is much smaller, even worse for DROP. The *concatenated datasets* baseline performs significantly worse than *data mixing* method, only improving for MMLU compared to OpenLLaMA and significantly worse than SNI fine-tuned model. This highlights the importance of choosing the optimal weights for *data mixing* and training a strong baseline.

[1]The low performance on BBH after training on P3 is inline with previous findings (Iyer et al., 2022). The T0pp 11B's (Sanh et al., 2022) accuracy is 13.0, after being fine-tuned only on the P3 dataset.

Models	Classification	Closedbook QA	OpenBook QA	Math	Reasoning	Conversational
OpenLLaMA	49.68	23.21	48.55	**10.45**	50.40	33.03
Data Mixing	56.52	**28.71**	44.36	5.15	51.13	34.51
DEM (Ours)	**56.94**	28.24	**54.34**	7.78	**53.31**	**40.22**

Table 3: Summary results of the HELM evaluations on held-out scenarios, grouped by task-category for the 7B model. *DEM* outperforms *data mixing* approach in five out of six HELM task clusters.

Next, we combine the models fine tuned on single datasets with distribution vector and interpolation method using Eq 2 and 3 respectively. The corresponding results are shown in Table 2. We observe that both approaches perform significantly better than the best *data mixing* model for all 3 scenarios showing their effectiveness (see Appendix D for MMLU performance per domain). We also compare *DEM* with larger fine-tuned models (Flan-PaLM (8B), OPT-IML (30B, 175B) and observe that *DEM* performs better although these models were trained on a larger mix of tasks and datasets compared to our model. Additionally, *DEM - Distribution Vector* performs better than *DEM - Interpolation* due to more flexible choice of ω_i (Section 4.2) and we use it in the rest of the paper.

We further expand our evaluation setting to include HELM scenarios. Here, we compare the performance of the pretrained model with *DEM* and *data mixing* model on multiple HELM held-out task clusters (see Table 3). *DEM* outperforms the *data mixing* approach in five out of six HELM task clusters. Surprisingly, for Math task category, the fine-tuned model performance degrades as compared to the pretrained model. Closer inspection reveals that this degradation is partly due to the fact that the instruction-tuned model does not output the answer in the correct format (as expected by HELM evaluation metric). The detailed HELM evaluation results (including results on 'seen' tasks) are reported in the Appendix D (Table 12).

6.2 Effect of Model Size Scaling

We evaluate the performance of the proposed *DEM* approach with increasing model sizes using OpenLLaMA 3B, 7B, and 13B models, quantifying the impact with both smaller and larger models. We trained the baseline *Data Mixing* model using the method discussed in Section 5.5. On the other hand, we fine-tuned the models on each dataset separately and combined them using Eq 2, similar to the 7B model as discussed in Section 6.1. We use the same model mixing weight of $\omega_i = 0.25$ (optimized for 7B model) for models of all sizes and present the results in Table 4. We observe that the model performance increases as we scale up the model size from 3B to 13B for both *Data Mixing* and *DEM*. Additionally, *DEM* yields performance improvement for each model size, showing the effectiveness and generalizability of the proposed approach with model size.

# Params	Models	MMLU	BBH	DROP	MathQA
3B	Data Mixing	41.08	31.36	25.98	28.54
	DEM	**43.67**	**34.14**	**28.89**	**29.78**
7B	Data Mixing	47.77	36.38	32.71	30.35
	DEM	**50.74**	**40.56**	**37.96**	**32.16**
13B	Data Mixing	52.7	40.48	43.15	30.72
	DEM	**54.53**	**42.65**	**46.59**	**33.13**

Table 4: Effect of model size on the performance of the proposed approach. We observe performance improvement using *DEM* for both smaller (3B) and larger (13B) models compared to *Data Mixing* baseline.

6.3 Impact of Different Training Datasets

In this section, we analyze the impact of each training dataset included in the mixture on the downstream task performance. For this, we progressively add the data distribution vector corresponding to each dataset to the base model (following Eq. 2) and evaluate the resulting model. We use $\omega_i = 0.25$ for all datasets to keep the setup simple. The performance of the resulting models are presented in Table 5. We observe that these data sources yield different levels of performance gains, as expected. This can be due to the various levels of mismatch between the train and test distribution. We observe that combining the pretrained model with single-task distribution vectors (e.g Math QA) or smaller mix of tasks (e.g., CoT) leads to smaller improvement whereas large scale multi-task distribution vectors (e.g., P3 and SNI) yields a much larger performance gain, in comparison. It can also be due to the large diversity of tasks and samples in P3 and SNI. InstructDial is an exception, which can be due

Training Dataset	MMLU	BBH	DROP	MathQA
Open LLaMA	40.31	32.84	24.38	27.71
+ CoT	41.30	33.68	25.46	28.44
+ MathQA	41.67	33.73	26.05	28.68
+ P3	47.12	36.58	30.82	30.35
+ InstDial	47.44	38.20	31.15	30.65
+ SNI	**50.74**	**40.56**	**37.96**	**32.16**

Table 5: Effect of progressively adding distribution vectors (Eq 1) from different data sources to the pretrained model using *DEM* (Eq 2). The performance increases as we add more data sources.

OpenLLaMA vs.	**Euclidean**
P3	35.1
InstDial	85.1
SNI	34.1
CoT	3.2
MathQA	4.1
Data Mixing	74.6
DEM	20.8

Table 6: Euclidean distance between the base model (OpenLLaMA) and the fine-tuned models.

to the conversational nature of this dataset, making it very different from the evaluation tasks.

6.4 Properties of Distribution Vectors

To better understand the behavior of *DEM*, we examine the characteristics of the fine-tuned models and their corresponding distribution vectors, as defined in (Eq 1). We evaluate the similarity between models by calculating the Euclidean distance and the cosine similarity after converting their weights into a single flattened vector representation.

Individual model distance from base. In Table 6, we show the Euclidean distance from the base model to each fine-tuned model. Datasets with more examples (P3, Instruct Dial, and SNI) lead to models that are further away from the base. The largest change is caused by Instruct Dial (x3 compared to the second largest), since it introduces a very specific domain (i.e., conversations), and requires higher adaptation of the model. In contrast, smaller datasets (CoT, and Math QA) only contribute small changes (3-4 points). As expected, the distribution edited model (*DEM*) is closer to the base model than the models trained on the largest datasets. This is because DEM is derived from a weighted average of the individual vectors. Finally, we observe that the *Data Mixing* model has significantly higher euclidean distance (x3) from the base

↓ **Dist. Vector** →	**P3**	**InstDial**	**SNI**	**CoT**	**MathQA**
InstDial	0.07	-			
SNI	0.09	0.08	-		
CoT	0.02	0.01	0.02	-	
MathQA	0.01	0.01	0.01	1.0	-
Data Mixing	0.27	0.29	0.19	0.10	0.10
DEM	0.44	0.87	0.43	0.06	0.06

Table 7: Cosine similarity between distribution vectors.

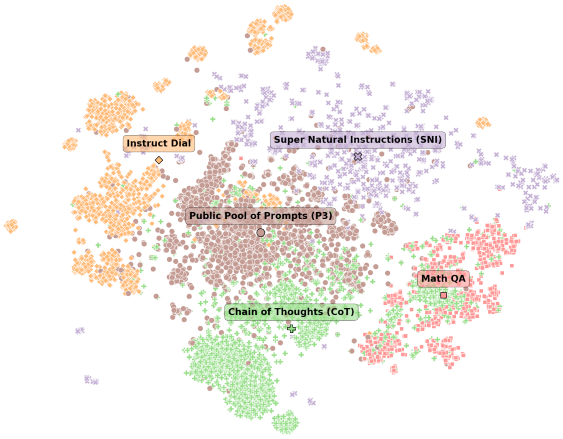

Figure 2: tSNE representation of the fine-tuning datasets. The centroids of the datasets are marked as larger points with captions.

model compared to *DEM*, indicating that the *Data Mixing* approach introduces a larger change.

Pairwise similarity between distribution vectors. Next, in Table 7 we compare the pairwise cosine similarity between the DVs from the fine-tuned models. We show that most of the individual DVs are almost orthogonal, except CoT and Math QA. This suggests that fine-tuning on these datasets does not lead to interference and introduces different abilities into the model.

To understand this, we sample *2,000* points from each dataset and plot their embedding representations into a common space using tSNE (van der Maaten and Hinton (2008), see Figure 2).[2] We observe a large number of CoT prompts that are close to the centroid of the MathQA dataset, which may explain the high similarity between their DVs. Note that CoT also has a small overlap with P3 but further away from their centroids, making the two DVs almost orthogonal. All other datasets have a minimal overlap between each other, and form

[2]We encode all texts after formatting them into their corresponding prompt using sentence-transformers/all-MiniLM-L12-v2.

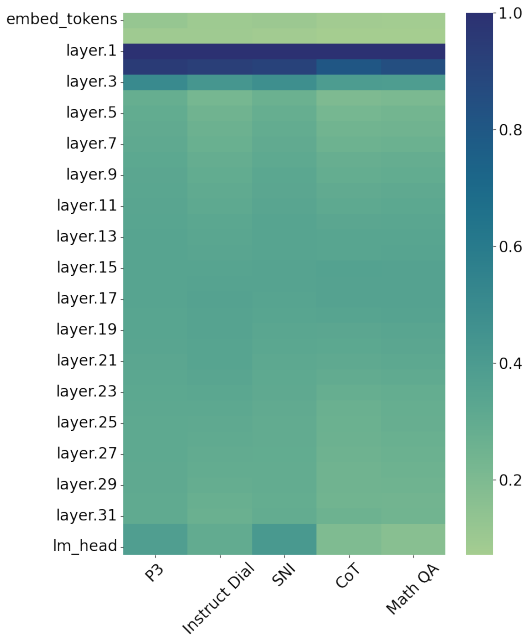

Figure 3: Layer-wise Euclidean distance, comparison between the base OpenLLaMA model, and the tuned models. Darker colors mean higher absolute difference. The euclidean distance values are normalized per-model by the highest layer-distance for that model. The plots are invariant to the scale of the weight change.

independent clusters. We also study the relation between the combined DV and individual DVs (last row in Table 7). We observe that *DEM* is oriented towards the DVs from the models with the highest euclidean distance from the pretrained model.

Finally, we compare how the *data mixing* model is oriented relative to the individual DVs. The cosine similarity with all DVs is less than 0.3, however, the model is oriented towards the DVs of bigger and more diverse datasets (InstDail, P3, SNI). The similarity with the CoT and MathQA is a bit higher, but it still remains within 0.1.

Layer-wise distance of individual models from base. Finally, to fully understand the changes in the models and why *DEM* is an effective strategy for data distribution mixing, we zoom in even further into the layer-wise euclidean distance (Figure 3) between the individual task vectors and the base model (OpenLLaMA 7B). From Figure 3, it is evident that the changes in the tuned models occur mostly in the first three layers. The embedding layers remain relatively stable across different domains and dataset sizes, indicating that the fundamental properties are preserved. New knowledge is primarily acquired by the 2-3 layers, which contributes to the success of the proposed approach. Furthermore, this study suggests that when combining weights, it is not necessary to take into account all the weights involved. Instead, it is possible to safely remove or prune certain weights in the combination without significantly impacting the outcome, as also shown by Yadav et al. (2023).

Train/Val Runs	time / step	# steps	# gpus	Cost
DEM				
- CoT	6.5	550	8	8
- Math QA	6.5	600	8	8.7
- P3	4.8	6000	32	256
- InstDial	5.2	23000	16	530
- SNI	5.24	6000	16	140
- Validation (10x)	2.1	500	8	23
Total				**966**
Data-Mixing (50x)	5.24	15000	16	**11650**

Table 8: Training cost (in gpu-hours) of 7B model on different instruction following datasets computed using Eq 4. Note that the number of steps is not equal to the number of examples.

6.5 Compute Cost Comparison

We use Eq. 4 to compare the real compute cost of the proposed *DEM* approach with the baseline *data mixing* method for 7B model on Nvidia A100 machines (with 8 gpus per node). Note that, this cost is specific to our setup and it can change depending on the model size, training parallelization scheme and other factors. In Table 8, we present the gpu-hours used by different training runs, as well as the validation runs needed for finding optimal model mixing weight ω_i in Eq 2. In each case, we did early stopping to obtain the best validation loss, which results in varying number of training steps ('# steps' in Table 8). As discussed in Appendix B, we use a combination of 10 weights to get the best model for *DEM*, which costs 23 gpu-hours. The total cost (training+validation) for *DEM* is 966 gpu-hours.

For the baseline *data mixing*, we trained 50 models with different weight combination (the exact weight selection process is described in Appendix B). Each run costs an average of 233 gpu-hours, resulting in a total cost of 11650 gpu-hours. This is more than 11 times the total cost of *DEM*.

7 Conclusions and Future Work

We proposed a simple and efficient approach for training on diverse data distributions that trains

checkpoints individually on each data source and then combines them with basic element-wise vector operations. *DEM* siginficantly outperforms the standard weighted data mixing in terms of downstream performance and overall compute cost. Our experiments demonstrate that *DEM* works with both single-task (e.g. Math QA) and multi-task data distributions (e.g. SNI, P3), and that they can be incrementally added to the pretrained model, resulting in improved downstream performance. We further performed extensive analysis to better understand the properties of the learned distribution vectors, finding that *DEM* is better aligned with the individual models than baseline while remaining close to the original model.

In future, it is important to evaluate the proposed approach using other model architectures e.g. encoder-decoder or mixture of experts model to better understand its effectiveness with other model designs. Additionally, *DEM* can be further improved by using more sophisticated methods for combining the individual checkpoints that can reduce the negative effects of interfering data distributions.

Acknowledgments

We thank the anonymous reviewers for their helpful questions and comments, which have helped us improve the quality of the paper. We also want to thank Yang Li for their help in setting up HELM evaluation framework, and Thomas Müller and Lluís Màrquez for helpful discussions.

Limitations

While this paper proposes an efficient and effective alternative to data mixing for training multi-task and instruction-following models, it is important to acknowledge its limitations:

- *Task granularity.* The distribution vectors of *DEM* are applicable to data distributions that span a single or multiple tasks. Our experimentation focused on existing data sources with different granularities ranging from several hundred tasks (e.g. P3) to a single one (e.g. MathQA), hence, the resulting distribution vectors captured varying task granularities. A detailed investigation of granularities and how to automatically group the data is an open area of investigation.

- *Architecture type.* The proposed approach makes no specific assumptions regarding the architecture and should be, in principle, applicable to any architecture variant including Mixture-of-Expert models (Fedus et al., 2022; Xue et al., 2024; Jiang et al., 2024; Sukhbaatar et al., 2024; Hu et al., 2024). Due to budget constraints, the evaluation of different architecture types was not included in the experiment plan. Therefore, the compatibility of *DEM* with different architecture types remains to be evaluated.

- *Storage Requirements. DEM* reduces the computational cost of training models, but it requires storing a number of distribution vectors in the hard drive. For very large models, this creates the need for large storage capacity that may not be always available. One straight-forward solution to this problem is to use parameter-efficient methods to train the distribution vectors instead of full training or discard the distribution vectors once the optimal combination has been identified.

References

Alon Albalak, Liangming Pan, Colin Raffel, and William Yang Wang. 2023. Efficient online data mixing for language model pre-training. *arXiv 2312.02406*.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '19, pages 2357–2367, Minneapolis, Minnesota, USA.

Devansh Arpit, Huan Wang, Yingbo Zhou, and Caiming Xiong. 2022. Ensemble of averages: Improving model selection and boosting performance in domain generalization. In *The Thirty-Sixth Annual Conference on Neural Information Processing Systems*, volume 36 of *NeurIPS '22*, pages 8265–8277, New Orleans, Louisiana, USA.

Stephen Bach, Victor Sanh, Zheng Xin Yong, Albert Webson, Colin Raffel, Nihal V. Nayak, Abheesht Sharma, Taewoon Kim, M Saiful Bari, Thibault Fevry, Zaid Alyafeai, Manan Dey, Andrea Santilli, Zhiqing Sun, Srulik Ben-david, Canwen Xu, Gunjan Chhablani, Han Wang, Jason Fries, Maged Alshaibani, Shanya Sharma, Urmish Thakker, Khalid Almubarak, Xiangru Tang, Dragomir Radev, Mike Tian-jian Jiang, and Alexander Rush. 2022. PromptSource: An integrated development environment and repository for natural language prompts. In *Proceedings of the 60th Annual Meeting of the Association for*

Computational Linguistics: System Demonstrations, ACL '21, pages 93–104, Dublin, Ireland.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of the 33th International Conference on Neural Information Processing Systems*, NeurIPS '20, Virtual.

Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. 2024. SWAD: domain generalization by seeking flat minima. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NeurIPS '21, Virtual.

Yew Ken Chia, Pengfei Hong, Lidong Bing, and Soujanya Poria. 2024. InstructEval: Towards holistic evaluation of instruction-tuned large language models. In *Proceedings of the First edition of the Workshop on the Scaling Behavior of Large Language Models*, SCALE-LLM '24, pages 35–64, St. Julian's, Malta.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Together Computer. 2023. RedPajama-Data: An open source recipe to reproduce LLaMA training dataset.

Nico Daheim, Thomas Möllenhoff, Edoardo Ponti, Iryna Gurevych, and Mohammad Emtiyaz Khan. 2024. Model merging by uncertainty-based gradient matching. In *The Twelfth International Conference on Learning Representations*, ICLR '24, Vienna, Austria.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2024. A framework for few-shot language model evaluation.

Xinyang Geng and Hao Liu. 2023. OpenLLaMA: An open reproduction of LLaMA.

Prakhar Gupta, Cathy Jiao, Yi-Ting Yeh, Shikib Mehri, Maxine Eskenazi, and Jeffrey Bigham. 2022. InstructDial: Improving zero and few-shot generalization in dialogue through instruction tuning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, EMNLP '22, pages 505–525, Abu Dhabi, United Arab Emirates.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, ACL '20, pages 8342–8360, Online.

Xinshuo Hu, Dongfang Li, Baotian Hu, Zihao Zheng, Zhenyu Liu, and Min Zhang. 2024. Separate the wheat from the chaff: Model deficiency unlearning via parameter-efficient module operation. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence*, volume 38 of *AAAI '20*, pages 18252–18260, Vancouver, Canada.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2022a. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*, volume abs/2212.04089 of *ICLR '23*, Kigali, Rwanda.

Gabriel Ilharco, Mitchell Wortsman, Samir Yitzhak Gadre, Shuran Song, Hannaneh Hajishirzi, Simon Kornblith, Ali Farhadi, and Ludwig Schmidt. 2022b. Patching open-vocabulary models by interpolating weights. In *The Thirty-Sixth Annual Conference on Neural Information Processing Systems*, volume 35 of *NeurIPS '22*, pages 29262–29277, New Orleans, Louisiana, USA.

Srinivasan Iyer, Xi Victoria Lin, Ramakanth Pasunuru, Todor Mihaylov, Daniel Simig, Ping Yu, Kurt Shuster, Tianlu Wang, Qing Liu, Punit Singh Koura, et al. 2022. OPT-IML: Scaling language model instruction meta learning through the lens of generalization. In *ArXiv abs/2212.12017*.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. 2018. Averaging weights leads to wider optima and better generalization. In *The 34th Conference on Uncertainty in Artificial Intelligence*, UAI '18, pages 876–885, Monterey, California, USA.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. 2023. Dataless knowledge fusion by merging weights of language models. In *The Eleventh International Conference on Learning Representations*, ICLR '23, Kigali, Rwanda.

Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A. Smith, and Luke Zettlemoyer. 2022. Branch-Train-Merge: Embarrassingly parallel training of expert language models. In *First Workshop on Interpolation Regularizers and Beyond at NeurIPS 2022*, INTERPOLATION '22, New Orleans, Louisiana, USA.

Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. 2020. On the convergence of FedAvg on non-IID data. In *The Eigth International Conference on Learning Representations*, ICLR '20, Addis Ababa, Ethiopia.

Ziyue Li, Tian Li, Virginia Smith, Jeff Bilmes, and Tianyi Zhou. 2024. Many-objective multi-solution transport. *arXiv preprint arXiv:2403.04099*.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Alexander Cosgrove, Christopher D Manning, Christopher Re, Diana Acosta-Navas, Drew Arad Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue WANG, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri S. Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Andrew Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. 2023. Holistic evaluation of language models. *Transactions on Machine Learning Research*.

Michael S Matena and Colin A Raffel. 2022. Merging models with fisher-weighted averaging. In *The Thirty-Sixth Annual Conference on Neural Information Processing Systems*, volume 35 of *NeurIPS '22*, pages 17703–17716, New Orleans, Louisiana, USA.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, volume 36 of *NeurIPS '22*, pages 27730–27744, New Orleans, Louisiana, USA.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(1):140:1–140:67.

Alexandre Ramé, Kartik Ahuja, Jianyu Zhang, Matthieu Cord, Léon Bottou, and David Lopez-Paz. 2023. Model ratatouille: recycling diverse models for out-of-distribution generalization. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23, Honolulu, Hawaii, USA.

Alexandre Rame, Matthieu Kirchmeyer, Thibaud Rahier, Alain Rakotomamonjy, patrick gallinari, and Matthieu Cord. 2022. Diverse weight averaging for out-of-distribution generalization. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, volume 36 of *NeurIPS '22*, New Orleans, Louisiana, USA.

H S V N S Kowndinya Renduchintala, Sumit Bhatia, and Ganesh Ramakrishnan. 2024. SMART: Submodular data mixture strategy for instruction tuning. In *Findings of the Association for Computational Linguistics ACL 2024*, Findings '24, pages 12916–12934, Bangkok, Thailand and virtual meeting.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. 2022. Multitask prompted training enables zero-shot task generalization. In *The Tenth International Conference on Learning Representations*, ICLR '22, Virtual Event.

Raphael Schumann, Elman Mansimov, Yi-An Lai, Nikolaos Pappas, Xibin Gao, and Yi Zhang. 2024. Backward compatibility during data updates by weight interpolation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2846–2861, St. Julian's, Malta.

Sainbayar Sukhbaatar, Olga Golovneva, Vasu Sharma, Hu Xu, Xi Victoria Lin, Baptiste Rozière, Jacob Kahn, Daniel Li, Wen-tau Yih, Jason Weston, et al. 2024. Branch-Train-MiX: Mixing expert LLMs into a mixture-of-experts LLM. *arXiv preprint arXiv:2403.07816*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. LLaMA: Open and efficient foundation language models. *ArXiv preprint*, abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *ArXiv preprint*, abs/2307.09288.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. 2022. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, EMNLP '22, pages 5085–5109, Abu Dhabi, United Arab Emirates.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022a. Finetuned language models are zero-shot learners. In *The Tenth International Conference on Learning Representations*, ICLR '22, Virtual.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022b. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, volume 36 of *NeurIPS '22*, pages 24824–24837, New Orleans, Louisiana, USA.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. 2022a. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *The Tenth International Conference on Learning Representations*, volume 162 of *ICLR '22*, pages 23965–23998, Virtual.

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, and Ludwig Schmidt. 2022b. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR '22, pages 7959–7971, New Orleans, Louisiana, USA.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. 2023. DoReMi: Optimizing data mixtures speeds up language model pretraining. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NeurIPS '23, New Orleans, Louisiana, USA.

Fuzhao Xue, Zian Zheng, Yao Fu, Jinjie Ni, Zangwei Zheng, Wangchunshu Zhou, and Yang You. 2024. OpenMoE: An early effort on open mixture-of-experts language models. In *Proceedings of the Forty-first International Conference on Machine Learning*, ICML '24, Vancouver, Canada.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. 2023. TIES-merging: Resolving interference when merging models. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NeurIPS '23, New Orleans, Louisiana, USA.

Gizem Yüce, Guillermo Ortiz-Jiménez, Beril Besbinar, and Pascal Frossard. 2022. A structured dictionary perspective on implicit neural representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, CVPR '22, pages 19206–19216, New Orleans, Louisiana, USA.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. OPT: Open pre-trained transformer language models. *ArXiv preprint*, abs/2205.01068.

Appendix

A Training Hyperparameters

In this section, we describe the detailed hyperparameters that we used for fine tuning the OpenLLaMA model using different datasets separately for *DEM* and combined for *Data Mixing* and *Concatenated Datasets*. In all these cases, we use a constant learning rate of 2e-5 with a 2000 step warmup. We tested other learning rate schedules with cosine and linear decay in preliminary experiments, however, they lead to worse performance. We use AdamW optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay of 0.05 and gradient clipping of 1. We also adjust batch size for different datasets based on the validation loss. We use example packing to fit multiple training examples into a single sample of a batch for efficient training. This is a greedy packing approach where we pack training examples until we reach the max sequence length that we can fit into the model. We do not overflow an example into the next sample of a batch (as generally done during pretraining (Brown et al., 2020)), rather use padding to fill the sample. The full setting is presented in Table 9. We show the batch size using total number of token after sample packing. The number of training steps indicates the step with best validation loss, and its variable for different datasets. Note that for InstDial, this value is particularly high because of the different kind of samples (i.e. dialog) that consists of this dataset.

Dataset / Method	Batch Size	Learning Rate	# Training Steps
CoT	65k	2e-5	550
MathQA	65k	2e-5	600
P3	262k	2e-5	6k
InstDial	131k	2e-5	23k
SNI	131k	2e-5	6k
Data Mixing	131k	2e-5	15k
Concatenated Datasets	131K	2e-5	17K

Table 9: Training hyperparameters for different models.

B Choosing Data Mixing Weights

Based on initial experiments, we determined the following hyperparameter ranges for the baseline data mixing approach – CoT: [0.05, 0.1, 0.15, 0.20], Math QA: [0.05, 0.1, 0.15, 0.20], P3: [0.25, 0.30, 0.35, 0.40], InstructDial: [0.30, 0.35, 0.40, 0.45], Super Natural Instructions: [0.15, 0.20, 0.25, 0.30]. Out of the 1024 possible weight combinations above, we randomly selected 50 combinations for training and selected the best weight setting based on validation-loss. The optimal data mixing setting was the following: P3 - 0.30, SNI - 0.20, InstructDial - 0.40, MathQA - 0.05, CoT - 0.05 The total cost for this hyperparameter search procedure is listed in Table 8).

Models	MMLU	BBH	DROP
Open LLaMA	40.31	32.84	24.38
DEM - Distribution Vector			
$\omega = 0.25$	50.74	**40.56**	37.96
Random Search, x50	**50.98**	40.55	**40.83**

Table 10: Downstream task performance of the DEM w/ Distribution Vector. We compare the weight selection strategies: single-coefficient vs. random search with 50 iterations.

C Choosing DEM Weights

In order to select the optimal mixing weights for *DEM - Distribution Vector* (Eq 2), we perform a grid search over ω_i values. For each coefficient combination we evaluate the validation losses on the five datasets used for fine-tuning (Section 5.1), and select the model that minimizes their average. However, exhaustive grid search is expensive as the number of combinations grows exponentially. Thus, we simplify Eq 2 and optimize a single coefficient ω for all datasets. We found $\omega = 0.25$ (out of 10 values) to produce the best validation loss and use it for all our experiments.

We chose the weights for *DEM - Interpolation* (Eq 3) in a similar manner as *DEM - Distribution Vector* by randomly sampling weights from the search grid and normalizing them to sum to 1. Additionally, we also tried the same weights as data mixing and equal weight of 0.2 for each of 5 datasets. The simplest strategy of equal weight performed on par with the best weight combination in terms of average val loss. So, we chose this and reported the corresponding results in Table 2

To measure the effect of using a single coefficient, we perform a limited budget experiment with 50 weight combinations, which are produced using individual weights for each distribution vector (Eq 1), sampled uniformly from the interval $[0, 1]$. Our results show that the best single-coefficient models perform better or on par with the sampled models in terms of average validation loss. This formulation was also adopted in other model interpolation works (Ilharco et al., 2022a; Yadav et al.,

2023). In Table 10, we show the differences in performance on three benchmarks (MMLU, BBH, DROP) using the Open LLaMa 7B model. The two strategies have similar performance on MMLU and BBH but the random search has an advantage of 3 points on DROP. However, this increase comes at the expense of 5x increase in cost (10 evaluations for uniform vs. 50 evaluations for random search). The best distribution weights we found are: CoT - 0.1, InstDial - 0.12, MathQA - 0.1, P3 - 0.23, SNI - 0.45. We hypothesize that the single-vector weights will not be an optimal choice if there is high negative correlation between the vectors, i.e., the data distributions are conflicting.

D Fine-Grained Results

In Table 11 we show the model performance per domain on the MMLU benchmarking datasets. It covers five different categories, on all of which DEM outperforms the other alternatives.

In Table 12 we show the per-dataset results on HELM benchmark. We can see that our approach significantly outperforms data mixing and improves over the baseline model in most of the categories. Due to space limitations we show different datasets on different rows.

Training Dataset	STEM	Humanities	Social Sciences	Others	Average
Open LLaMA v2	33.4	36.8	45.1	47.3	40.3
LLaMA 1 (Touvron et al., 2023a)	34.0	30.5	38.3	38.1	35.1
LLaMA 2 (Touvron et al., 2023b)	42.9	36.4	51.2	52.2	45.3
Public Pool of Prompts (P3)	25.4	32.9	44.2	41.2	35.7
Instruct Dial	31.9	37.8	44.5	43.5	39.3
Super Natural Instructions (SNI)	38.4	42.6	53.9	52.9	46.5
Chain of Thoughts (CoT)	34.4	38.3	47.1	48.1	41.7
Math QA	32.8	36.6	44.1	46.5	39.7
Data Mixing	39.2	44.8	55.6	52.6	47.8
Concatenated Datasets (1-5)	37.9	41.4	49.8	46.1	43.4
DEM - Interpolation (Ours)	39.7	47.2	58.5	56.2	50.1
DEM - Distribution Vector (Ours)	**40.4**	**47.8**	**58.8**	**57.0**	**50.7**

Table 11: MMLU domain specific task performance of models trained on different instruction following datasets (Section 5.1). We compare it with different pretrained and fine-tuned baselines (Section 5.5) and our proposed approach in Section 4.

Models	MMLU	BoolQ	NarrativeQA	NaturalQ closed	NaturalQ open	QUAC	TruthfulQA	IMDB	CivilComments	RAFT	Wikifact
OpenLLaMA-v2	39.37	72.3	63.96	26.08	61.15	33.03	18.65	93.2	53.96	60.0	24.89
Data Mixing	43.96	85.3	71.24	21.84	19.5	34.52	42.35	87.0	64.8	69.09	21.94
DEM (ours)	46.61	82.4	71.24	28.1	69.39	40.22	29.82	96.6	53.75	67.27	26.82
Models	ReasonAbstract	ReasonNatural	bABI	Dyck	GSM-8K	Math-Eq	Math-CoT	LSAT	Legal	Imputation	EntityMatch
OpenLLaMA-v2	18.51	21.1	45.25	52.0	5.5	12.58	8.33	20.43	48.67	81.66	83.89
Data Mixing	20.58	29.64	54.52	40.0	0.5	8.79	1.52	24.35	62.37	76.4	85.62
DEM (ours)	23.24	34.73	56.62	48.4	6.3	11.06	4.52	18.26	58.49	71.56	85.32

Table 12: Detailed HELM results on 22 scenarios. HELM datasets that are part of model-training (BoolQ, GSM-8K and IMDB), are excluded from the aggregated results presented in Table 3