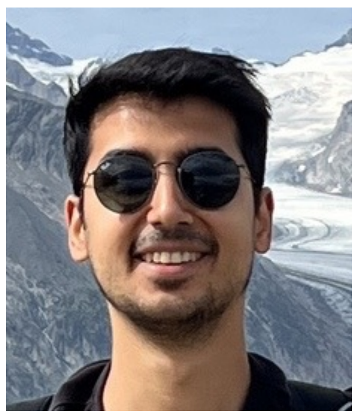
Mitigating Catastrophic Forgetting in the Data Oblivious Setting

Sujay Sanghavi

University of Texas, Austin

Sunny Sanyal



Ali Kavis

Hayden Prairie

Rudrajit Das

"Foundation Model Paradigm"

Pretraining

1. Train a large model on a lot of data in an unsupervised way

    Vision: CLIP

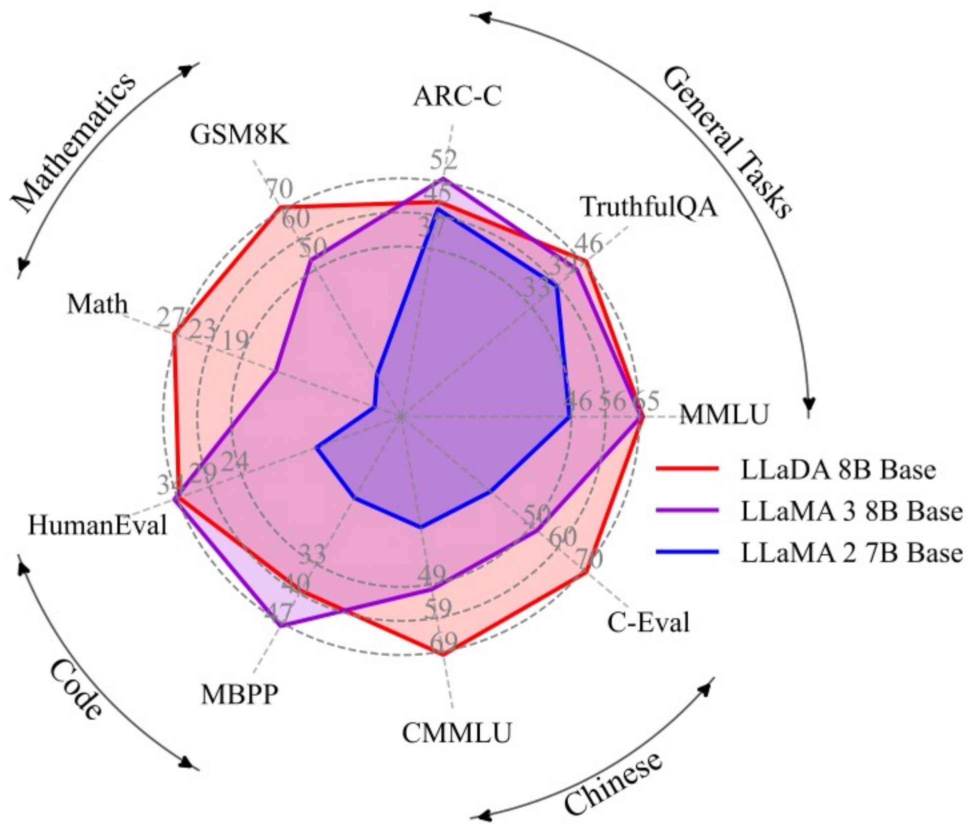
    Language: LLAMA, Mistral, Gemma, etc.

    Expensive, hard to re-create, **broadly performant**

Fine-tuning

2. Specialize this model for specific task(s)

Pre-trained Models are Valuable



(expensive) pre-training yields **broadly performant** models …

The same, “frozen” model (no specialization) has great **zero shot** or **few shot** performance on a range of tasks.

Source: Nie et. al., “Large language diffusion models”

Eg: Gemma

Benchmark	metric	Gemma-1 2B	Gemma-2 2B	Mistral 7B	LLaMA-3 8B	Gemma-1 7B	Gemma-2 9B	Gemma-2 27B
MMLU	5-shot	42.3	**52.2**	62.5	66.6	64.4	**71.3**	75.2
ARC-C	25-shot	48.5	**55.7**	60.5	59.2	61.1	**68.4**	71.4
GSM8K	5-shot	15.1	**24.3**	39.6	45.7	51.8	**68.6**	74.0
AGIEval	3-5-shot	24.2	**31.5**	44.0†	45.9†	44.9†	**52.8**	55.1
DROP	3-shot, F1	48.5	**51.2**	63.8*	58.4	56.3	**69.4**	74.2
BBH	3-shot, CoT	35.2	**41.9**	56.0◇	61.1◇	59.0◇	**68.2**	74.9
Winogrande	5-shot	66.8	**71.3**	78.5	76.1	79.0	**80.6**	83.7
HellaSwag	10-shot	71.7	**72.9**	**83.0**	82.0	82.3	81.9	86.4
MATH	4-shot	11.8	**16.0**	12.7	-	24.3	**36.6**	42.3
ARC-e	0-shot	73.2	**80.6**	80.5	-	81.5	**88.0**	88.6
PIQA	0-shot	77.3	**78.4**	**82.2**	-	81.2	81.7	83.2
SIQA	0-shot	49.7	**51.9**	47.0*	-	51.8	**53.4**	53.7
Boolq	0-shot	69.4	**72.7**	83.2*	-	83.2	**84.2**	84.8
TriviaQA	5-shot	53.2	**60.4**	62.5	-	63.4	**76.6**	83.7
NQ	5-shot	12.5	**17.1**	23.2	-	23.0	**29.2**	34.5
HumanEval	pass@1	**22.0**	20.1	26.2	-	32.3	**40.2**	51.8
MBPP	3-shot	29.2	**30.2**	40.2*	-	44.4	**52.4**	62.6
Average (8)		44.0	**50.0**	61.0	61.9	62.4	**70.2**	74.4
Average (all)		44.2	**48.7**	55.6	-	57.9	**64.9**	69.4

Eg: DeepSeek

	Benchmark (Metric)	Claude-3.5-Sonnet-1022	GPT-4o 0513	DeepSeek V3	OpenAI o1-mini	OpenAI o1-1217	DeepSeek R1
	Architecture	-	-	MoE	-	-	MoE
	# Activated Params	-	-	37B	-	-	37B
	# Total Params	-	-	671B	-	-	671B
English	MMLU (Pass@1)	88.3	87.2	88.5	85.2	**91.8**	90.8
	MMLU-Redux (EM)	88.9	88.0	89.1	86.7	-	**92.9**
	MMLU-Pro (EM)	78.0	72.6	75.9	80.3	-	**84.0**
	DROP (3-shot F1)	88.3	83.7	91.6	83.9	90.2	**92.2**
	IF-Eval (Prompt Strict)	**86.5**	84.3	86.1	84.8	-	83.3
	GPQA Diamond (Pass@1)	65.0	49.9	59.1	60.0	**75.7**	71.5
	SimpleQA (Correct)	28.4	38.2	24.9	7.0	**47.0**	30.1
	FRAMES (Acc.)	72.5	80.5	73.3	76.9	-	**82.5**
	AlpacaEval2.0 (LC-winrate)	52.0	51.1	70.0	57.8	-	**87.6**
	ArenaHard (GPT-4-1106)	85.2	80.4	85.5	92.0	-	**92.3**
Code	LiveCodeBench (Pass@1-COT)	38.9	32.9	36.2	53.8	63.4	**65.9**
	Codeforces (Percentile)	20.3	23.6	58.7	93.4	**96.6**	96.3
	Codeforces (Rating)	717	759	1134	1820	**2061**	2029
	SWE Verified (Resolved)	**50.8**	38.8	42.0	41.6	48.9	49.2
	Aider-Polyglot (Acc.)	45.3	16.0	49.6	32.9	**61.7**	53.3
Math	AIME 2024 (Pass@1)	16.0	9.3	39.2	63.6	79.2	**79.8**
	MATH-500 (Pass@1)	78.3	74.6	90.2	90.0	96.4	**97.3**
	CNMO 2024 (Pass@1)	13.1	10.8	43.2	67.6	-	**78.8**
Chinese	CLUEWSC (EM)	85.4	87.9	90.9	89.9	-	**92.8**
	C-Eval (EM)	76.7	76.0	86.5	68.9	-	**91.8**
	C-SimpleQA (Correct)	55.4	58.7	**68.0**	40.3	-	63.7

Forgetting

What happens to these models when they are fine-tuned ?

1. They improve on the task / domain they are fine tuned on

2. They **degrade on everything else** they were originally measured on

Method	Performance	General Capability			
	GSM8K	Commensense	MMLU	HumanEval	Average
Llama-2-7B	13.7	65.6	42.0	24.2	43.9
Full FT	**49.4**	62.3 -3.3	36.6 -5.4	16.1 -8.1	38.3 -5.6
HFT	47.5	65.5 -0.1	42.3 +0.3	23.6 -0.6	43.8 -0.1
L_1-regularization	39.0	65.1 -0.5	38.3 -3.7	**27.4** **+3.2**	43.6 -0.3
L_2-regularization	44.5	65.5 -0.1	39.2 -2.8	25.9 +1.7	43.5 -0.4
MoFO ($\alpha\% = 15\%$)	47.7	**65.7** **+0.1**	**42.7** **+0.7**	24.6 +0.4	**44.3** **+0.4**

e.g. LLAMA-2-7B finetuned on MathQA

Source: Chen et. Al. “MoFO: Momentum-Filtered Optimizer for Mitigating Forgetting in LLM Finetuning”

Data-oblivious Setting

Typically, we do not know how pretraining was done

- Dataset not public, or even if public too cumbersome
- Precise description of training choices not always available

Q: How do we mitigate forgetting when we **only have access to the model**, but not to the data + methodology used to train it ?

Notation

θ Weights of the model

$f_i(\theta)$ Loss function of the ith finetuning sample

θ_* Weights of the pre-trained model

Standard fine-tuning

$$\min_{\theta} \frac{1}{N} \sum_i f_i(\theta)$$ starting from θ_*

Approach 1: Regularize ($\ell_2 - \text{reg}$)

Overcoming catastrophic forgetting in neural networks

James Kirkpatrick[a], Razvan Pascanu[a], Neil Rabinowitz[a], Joel Veness[a], Guillaume Desjardins[a], Andrei A. Rusu[a], Kieran Milan[a], John Quan[a], Tiago Ramalho[a], Agnieszka Grabska-Barwinska [a], Demis Hassabis[a], Claudia Clopath[b], Dharshan Kumaran[a], and Raia Hadsell[a]

Penalize deviation from pretrained model parameters during fine-tuning

$$\min_{\theta} \; \frac{1}{N} \sum_i f_i(\theta) \; + \; \|\theta - \theta_*\|_2^2$$

Approach 2: Do not finetune everything

MoFO: Momentum-Filtered Optimizer for Mitigating Forgetting in LLM Fine-Tuning

Yupeng Chen*[1], Senmiao Wang*[1], Zhihang Lin[1], Zeyu Qin[3], Yushun Zhang[1,2], Tian Ding[2], and Ruoyu Sun†[1,2]

In each iteration, finds the parameters with the biggest momentum term and updates those

Approach 3: PEFT (LoRA)

LoRA Learns Less and Forgets Less

Dan Biderman[1,2], Jacob Portes[2], Jose Javier Gonzalez Ortiz[2], Mansheej Paul[2], Philip Greengard[1], Connor Jennings[2], Daniel King[2], Sam Havens[2], Vitaliy Chiley[2], Jonathan Frankle[2], Cody Blakeney[2], John P. Cunningham[1]

Title says it all …

$$\min_{\theta_{small}} \frac{1}{N} \sum_i f_i(\theta_{small} + \theta_*)$$

Approach 4: Weight ensemble (Wise-FT)

Robust fine-tuning of zero-shot models

Mitchell Wortsman[*†] Gabriel Ilharco[*†] Jong Wook Kim[§] Mike Li[‡]
Simon Kornblith[◇] Rebecca Roelofs[◇] Raphael Gontijo-Lopes[◇]
Hannaneh Hajishirzi[†○] Ali Farhadi[⋆†] Hongseok Namkoong[⋆‡] Ludwig Schmidt[†△]

First finetune normally, then average the weights of final and pretrained

$$\theta_1 \leftarrow \min_\theta \frac{1}{N} \sum_i f_i(\theta)$$ starting from θ_*

$$\theta_{final} \leftarrow \alpha\theta_1 + (1-\alpha)\theta_*$$

Approach 5: Ours

A new approach to mitigate forgetting

- Works for both generative and discriminative models
- Bests previous approaches (in our experiments)
- **Complementary to and additive with the other approaches**

Idea : Sample Weighting

At a high level, all approaches try to **discourage big moves** away from θ_*

Each sample *i* "causes" a move away from θ_* of magnitude $\nabla_\theta f_i(\theta_*)$

Idea: de-prioritize the samples which cause big moves

$$\min_\theta \sum_i \pi_i f_i(\theta)$$

Per-sample weight that depends on θ_*

Idea : Sample Weighting

Large $\nabla_\theta f_i(\theta_*)$ ⟷ Low π_i

Issue: calculating gradients is compute and memory intensive, so instead

Large $f_i(\theta_*)$ ⟷ Low π_i

1. For each sample find its weight based on it’s loss in the pretrained model
2. Solve weighted loss

Determining π_i

Two requirements:

1. For all $i \neq j$ such that $f_i(\theta_*) \geq f_j(\theta_*)$ we should have $\pi_i \leq \pi_j$

2. The distribution π should be spread out

Both requirements can be met by **entropic regularization**

$$\min_{\pi} \sum_i \pi_i f_i(\theta_*) + \tau \sum_i \pi_i \log \pi_i$$

Weighted pretrain loss

Spreads π out

Determining π_i

The optimal π that solves this entropic regularization objective is

$$\pi_i^* = \frac{1}{Z} \exp\left(-\frac{f_i(\boldsymbol{\theta}^*)}{\tau}\right)$$

We use this in our method - **FLOW**

Algorithm : FLOW

Algorithm 1 **F**ine-tuning with Pre-trained **L**oss-**O**riented **W**eighting (**FLOW**)

Input: Pre-trained model $\boldsymbol{\theta}^*$, dataset $\{(\mathbf{x}_i, \mathrm{y}_i)\}_{i=1}^n$ for the new task, temperature parameter τ.

$f_i(\boldsymbol{\theta}) \to i^{\text{th}}$ sample's loss at $\boldsymbol{\theta}$, with a non-negative loss function (e.g., cross-entropy loss).

1. Compute sample weights: $w_i = \exp\left(-\frac{f_i(\boldsymbol{\theta}^*)}{\tau}\right)$.
2. Weighted loss: $\mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^n w_i f_i(\boldsymbol{\theta})$.
3. Fine-tune with weighted loss: $\widehat{\boldsymbol{\theta}}^* := \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta})$.

Output: Fine-tuned model $\widehat{\boldsymbol{\theta}}^*$.

In our expts: τ is chosen to be the median pretrain loss on the entire finetuning dataset

But one could have it be the median of the mini-batch, a running / online median, etc.

Relationship to DRO

Our form is evocative of distributionally robust optimization

$$\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\pi} \in \boldsymbol{\Delta}_n} \sum_{i=1}^{n} \pi_i f_i(\boldsymbol{\theta}) - \tau \sum_{i=1}^{n} \pi_i \log \pi_i.$$

Except that ours is the exact opposite.

In particular, DRO will give $\pi_i^* \propto \exp\left(\frac{f_i(\boldsymbol{\theta})}{\tau}\right)$

DRO focuses on the hard samples, and so would accentuate forgetting

Experiment Setup

Recall: we want to measure the degradation in general capability when we finetune for a specific downstream task

Two pretrained models: Gemma 2 2B and LLAMA 3.2 3B

Downstream task:

fine-tune on MetaMathQA, measure on GSM8K

use CommonSense, MMLU and MBPP as proxies for general capability

Evaluations using 1m-evaluation-harness

We follow the setup used in both “LoRA learns less and forgets less” and “MoFO”

Results

	Method	General Capability Acc.			Target Acc.	
		Commonsense	MMLU	MBPP	GSM8K	Average
Gemma 2 2B	Pre-trained	57.23 (+0.00)	49.59 (+0.00)	**28.40** (+0.00)	24.49 (-38.89)	40.79
	Standard Fine-tuning	55.07 (-2.16)	45.59 (-4.00)	16.80 (-11.60)	**63.38** (+0.00)	46.31
	WiSE-FT ($\alpha = 0.5$)	57.28 (+0.05)	**50.13** (+0.54)	25.60 (-2.80)	53.30 (-10.08)	47.60
	LoRA ($r = 64$)	55.67 (-1.56)	44.28 (-5.31)	25.80 (-2.60)	60.43 (-2.95)	47.05
	ℓ_2-Regularization	57.01 (-0.22)	48.43 (-1.16)	24.80 (-3.60)	62.85 (-0.53)	49.19
	FLOW (Ours)	**57.59** (+0.36)	49.31 (-0.28)	26.80 (-1.60)	62.55 (-0.83)	**49.98**
Llama 3.2 3B	Pre-trained	54.48 (+0.00)	**54.34** (+0.00)	**38.00** (+0.00)	26.01 (-40.94)	44.28
	Standard Fine-tuning	50.68 (-3.80)	45.29 (-9.05)	17.80 (-20.20)	**66.95** (+0.00)	46.10
	WiSE-FT ($\alpha = 0.5$)	**54.54** (+0.04)	53.33 (-1.01)	34.60 (-3.40)	57.01 (-9.94)	50.75
	LoRA ($r = 64$)	53.10 (-1.38)	50.95 (-3.39)	34.00 (-4.00)	63.84 (-3.15)	51.66
	ℓ_2-Regularization	53.60 (-0.88)	51.28 (-3.06)	33.60 (-4.40)	66.87 (-0.08)	52.30
	FLOW (Ours)	54.30 (-0.18)	51.86 (-2.48)	36.00 (-2.00)	65.58 (-1.37)	**52.87**

Additiveness to ℓ_2 and LoRA

	Common Sense	MMLU	MBPP	GSM8K	
Method	**A1**	**A2**	**A3**	**B1**	**Avg.**
ℓ_2	57.01	48.43	24.80	**62.85**	49.19
ℓ_2+	**57.53**	**49.38**	**26.60**	62.02	**49.79**
LoRA	55.67	44.28	25.80	60.43	47.05
LoRA+	**56.74**	**47.68**	**28.80**	**61.49**	**49.31**

What about task-specific heads ?

In many applications (e.g. image classification) we may only borrow the Body/trunk of a pretrained model, and then add on a new prediction head

E.g. take a model trained on Imagenet, add on a prediction head for CIFAR-100, and then Linear probe or fine-tune

In this case, FLOW operates as follows:

1. Make a **linear probed** prediction head (on frozen pretrained body) for the downstream dataset

2. Use this to determine the weights

3. Weighted fine-tuning of this head + un-frozen body

Expt Setup

Pretrained models: ResNets trained on Imagenet 1K by [Russakovsky et. al.]

Datasets. We select six widely-used image classification datasets: CIFAR-10 [Krizhevsky, 2009], CIFAR-100 [Krizhevsky, 2009], Flowers102 [Nilsback and Zisserman, 2008], Caltech101 [Li et al., 2022], Cars [Krause et al., 2013], and Dogs [Parkhi et al., 2012].

For each method, we finetune on a per-dataset basis and then report average scores

Results

	Method	IN-1K Acc.		Target Acc.		Average
ResNet-18	Pre-trained	**69.76**	(+0.00)	–		–
	Standard FT	19.58	(-50.18)	**89.07**	(+0.00)	54.60
	Linear Probe	**69.76**	(+0.00)	73.57	(-15.50)	<u>71.63</u>
	ℓ_2-Reg.	34.78	(-34.98)	<u>88.12</u>	(-0.95)	61.45
	WiSE-FT	54.15	(-15.61)	80.23	(-8.84)	67.19
	FLOW (Ours)	<u>65.21</u>	(-4.55)	83.93	(-5.14)	**74.57**
ResNet-50	Pre-trained	**79.02**	(+0.00)	–		–
	Standard FT	36.91	(-42.11)	**91.78**	(+0.00)	64.34
	Linear Probe	**79.02**	(+0.00)	76.45	(-15.33)	<u>77.73</u>
	ℓ_2-Reg.	44.78	(-34.24)	<u>91.58</u>	(-0.20)	68.18
	WiSE-FT	61.65	(-17.37)	81.38	(-10.40)	71.52
	FLOW (Ours)	<u>76.09</u>	(-2.93)	86.25	(-5.53)	**81.17**

Additivity

	Method	IN-1K Acc.	Target Acc.	Average
ResNet-18	WiSE-FT	54.15	80.23	67.19
	WiSE-FT+	68.71	74.03	**71.37**
ResNet-50	WiSE-FT	61.65	81.38	71.52
	WiSE-FT+	78.29	73.80	**76.04**

Theory Summary

We analyze FLOW for the linear setting, and find the minimal set of conditions on four quantities:

- covariance matrix of the finetuning data $\widetilde{\Sigma}$
- covariance matrix of the pretraining data Σ
- parameters of the pretrained model θ_*
- optimal parameters of the model for downstream task only $\widetilde{\theta}_*$

Under which FLOW provably outperforms model averaging and ℓ_2

Theory Summary

$$\operatorname{err}_1(\boldsymbol{\theta}) := \mathbb{E}_{\mathcal{D}}\Big[\big(\mathrm{y} - \langle \boldsymbol{\theta}, \mathbf{x}\rangle\big)^2\Big] = \big(\boldsymbol{\theta} - \boldsymbol{\theta}_*\big)^\top \boldsymbol{\Sigma}\big(\boldsymbol{\theta} - \boldsymbol{\theta}_*\big),$$

$$\operatorname{err}_2(\boldsymbol{\theta}) := \mathbb{E}_{\widetilde{\mathcal{D}}}\Big[\big(\widetilde{\mathrm{y}} - \langle \boldsymbol{\theta}, \widetilde{\mathbf{x}}\rangle\big)^2\Big] = \big(\boldsymbol{\theta} - \widetilde{\boldsymbol{\theta}}_*\big)^\top \widetilde{\boldsymbol{\Sigma}}\big(\boldsymbol{\theta} - \widetilde{\boldsymbol{\theta}}_*\big)$$

$$\mathbf{e} := \boldsymbol{\theta}_* - \widetilde{\boldsymbol{\theta}}_*,$$

Theorem 7.2 (**FLOW**). *Let* $\mu = \left(\frac{\tau}{\tau + 2\|\mathbf{e}\|_2^2}\right)^{1/2}$. *Then:*

$$\widehat{\boldsymbol{\theta}}_K = \widetilde{\boldsymbol{\theta}}_* + \left(\mathbf{I}_d - 2\widehat{\eta}\widetilde{\boldsymbol{\Sigma}}'\right)^K \mathbf{e},$$

where $\widetilde{\boldsymbol{\Sigma}}' := \mu\big(\mathbf{I}_d - \mathbf{Q}\big)$ *with*

$$\mathbf{Q} = (1-\mu^2)\overline{\mathbf{e}}\overline{\mathbf{e}}^\top + \rho^2(1-\mu^2)\overline{\mathbf{e}}_\perp\overline{\mathbf{e}}_\perp^\top - \rho\mu^2\big(\overline{\mathbf{e}}\overline{\mathbf{e}}_\perp^\top + \overline{\mathbf{e}}_\perp\overline{\mathbf{e}}^\top\big).$$

By controlling τ we can find a Q that can stall convergence to $\widetilde{\theta}_*$ in bad directions

Summary

Selecting samples is a cheap and complementary way to mitigate model Forgetting

Solving catastrophic forgetting is a crucial step in continual learning