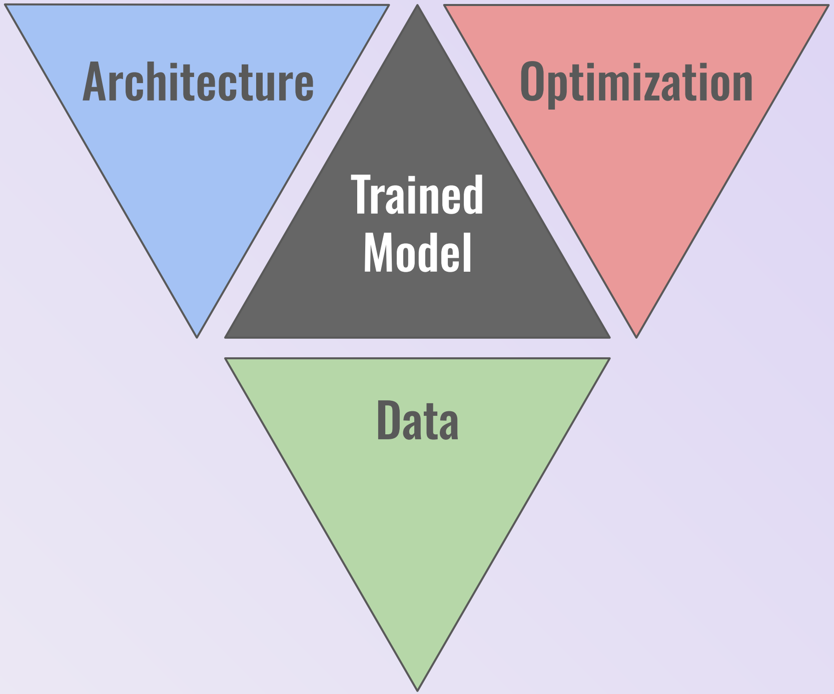
Understanding and Improving Models Through a Data-Centric Lens

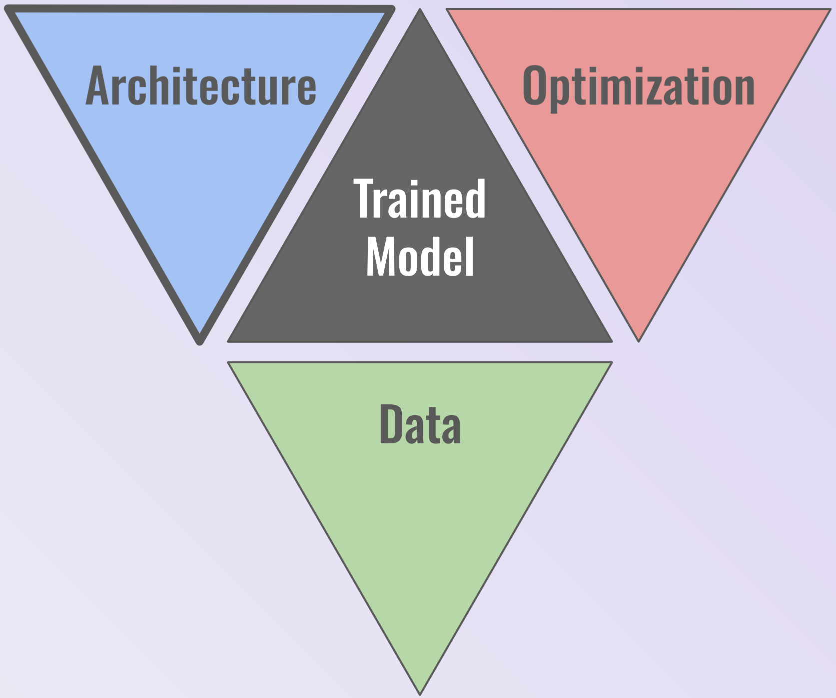
Alon Albalak

3 main components of training a model



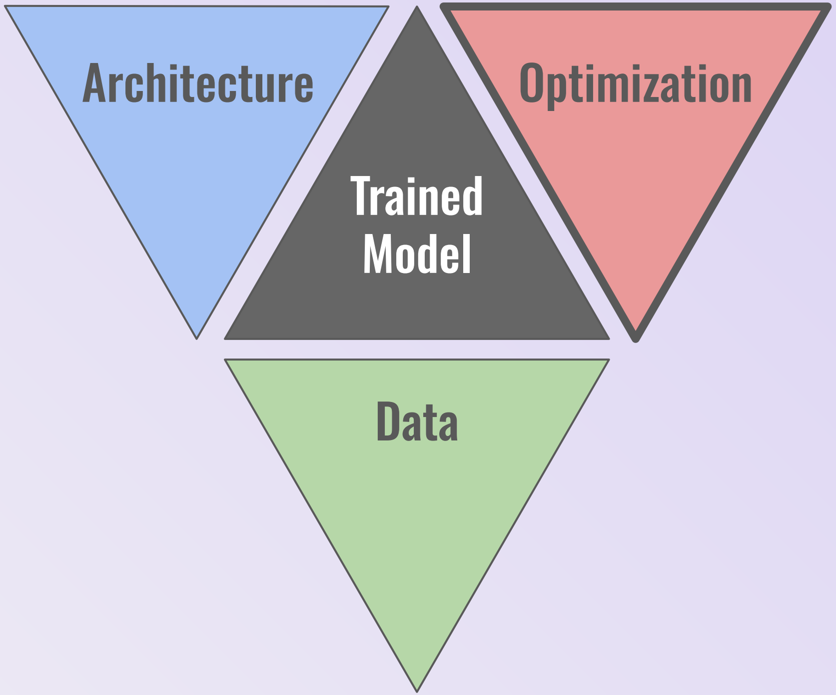
3 main components of training a model: Architecture

RWKV (EMNLP 2023)
Logic-LM (EMNLP 2023)
NeuPSL (IJCAI 2023)



3 main components of training a model: Optimization

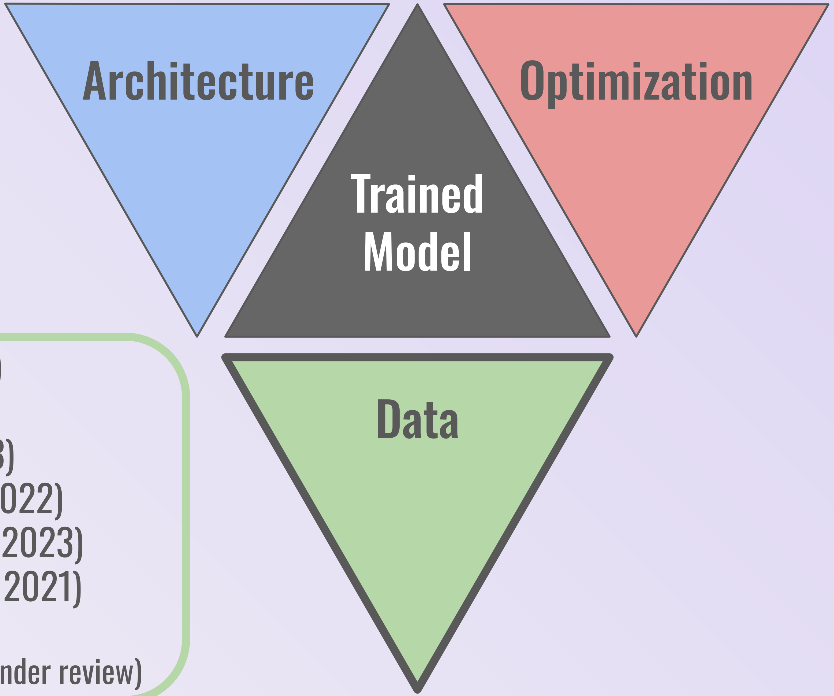
RWKV (EMNLP 2023)
Logic-LM (EMNLP 2023)
NeuPSL (IJCAI 2023)



D-REX (NLP4ConvAI 2022)
NeuPSL (IJCAI 2023)
CausalDialogue (ACL 2023)

3 main components of training a model: Data

RWKV (EMNLP 2023)
Logic-LM (EMNLP 2023)
NeuPSL (IJCAI 2023)

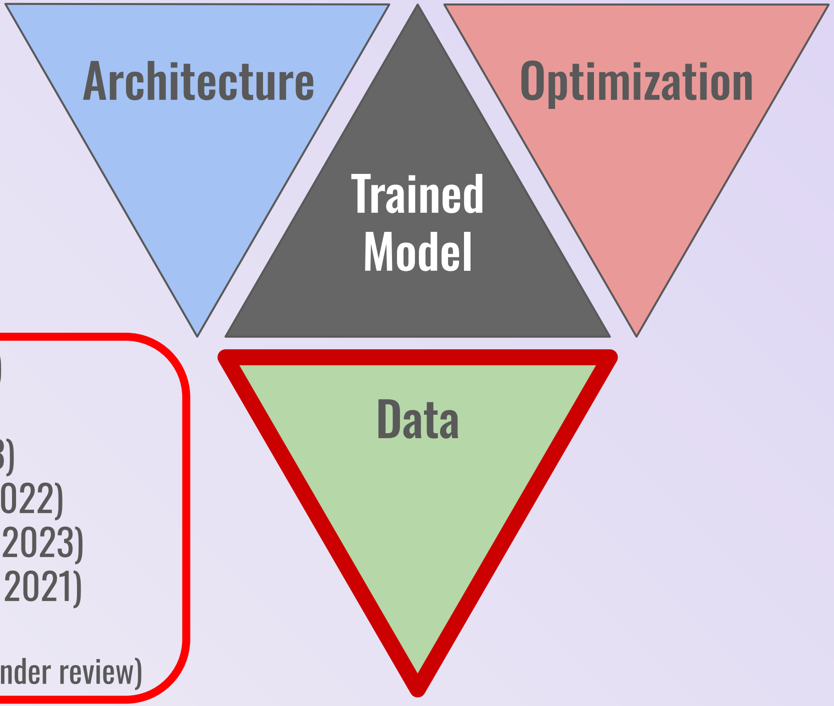


D-REX (NLP4ConvAI 2022)
NeuPSL (IJCAI 2023)
CausalDialogue (ACL 2023)

FLAD (NeurIPS 2023)
FETA (EMNLP 2022)
XL-ORQA (EACL 2023)
0-shot Xfer (ENLSP 2022)
CausalDialogue (ACL 2023)
Robust TOD (Taskbot 2021)
ODM (in progress)
Data Selection Survey (under review)

3 main components of training a model

RWKV (EMNLP 2023)
Logic-LM (EMNLP 2023)
NeuPSL (IJCAI 2023)

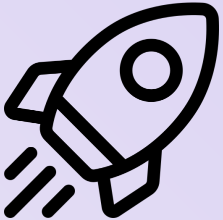


D-REX (NLP4ConvAI 2022)
NeuPSL (IJCAI 2023)
CausalDialogue (ACL 2023)

FLAD (NeurIPS 2023)
FETA (EMNLP 2022)
XL-ORQA (EACL 2023)
0-shot Xfer (ENLSP 2022)
CausalDialogue (ACL 2023)
Robust TOD (Taskbot 2021)
ODM (in progress)
Data Selection Survey (under review)

How can data affect a model?

- Improve model performance



- Reduce costs

- Ensure the integrity of evaluation

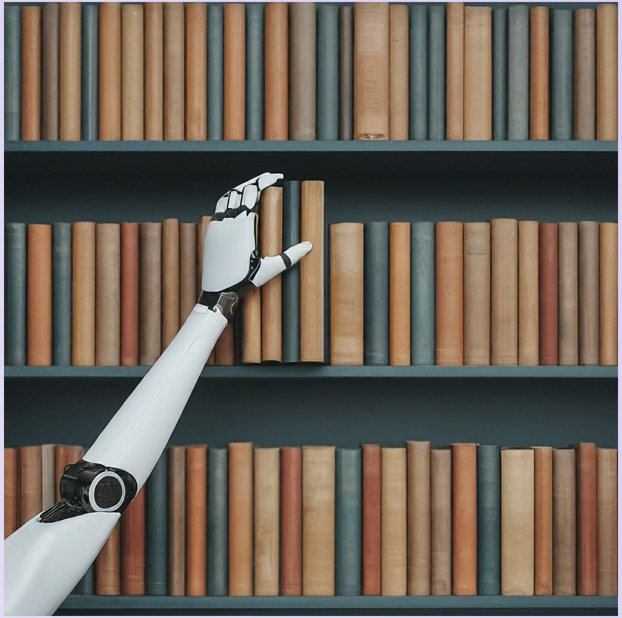
- Reduce undesirable behaviors

Data Selection: Can we train on better data?

- **Definition:**

  "The process of taking a collection of candidate data points and creating a dataset that will be used to train or evaluate a machine learning model"

- Often overlooked, we often are given data and assume that's what we should use
- How can we make any guarantees of optimality for a dataset?



Data mixing

- A sub-problem within data selection

- **Problem formulation:**
  - Given data from multiple domains/tasks
  - Determine the optimal mixture weight for each domain/task

Background on Multi-Armed Bandits (MAB)

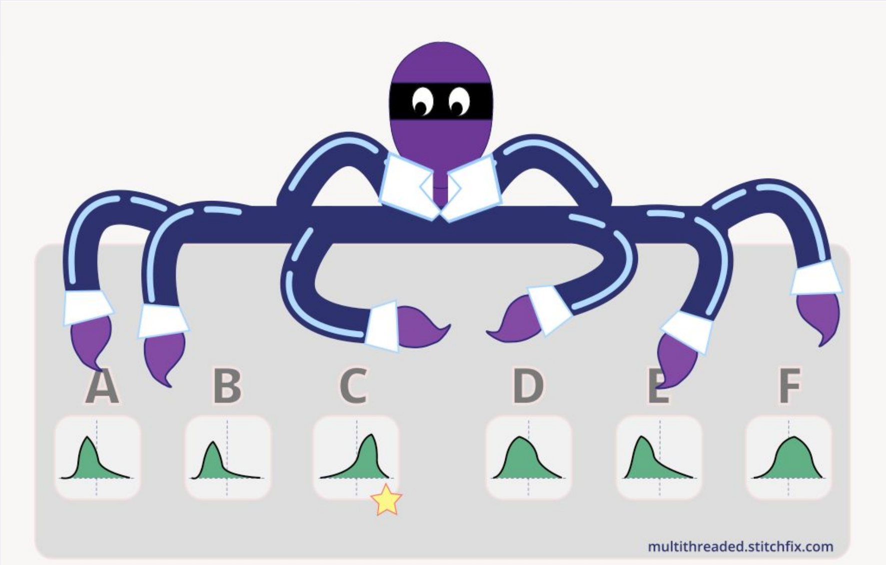
- MAB is a series of methods that solve the online decision making problem

- **Formulation**:
    - On each of N turns, select one of K arms
    - After being selected, arms return a reward generated by an unknown distribution.
    - Selection mechanism is the policy (**π**) of our bandit algorithm

- **Goal**: Accumulate the largest sum of rewards possible

- MAB defines a clear tradeoff between exploring and exploiting actions

Background on Multi-Armed Bandits (MAB)



MAB Algorithms

MAB Algorithms

EXP3

- **Exp**onential-weight algorithm for **Exp**loration and **Exp**loitation
- Defines a stochastic policy as linear combination of Gibbs and uniform distributions

$$\pi_t(a) = (1 - K\mathcal{E}_t)\frac{\exp(\mathcal{E}_{t-1}\hat{R}_a)}{\sum_{a'}\exp(\mathcal{E}_{t-1}\hat{R}_{a'})} + \mathcal{E}_t$$

$\mathcal{E}_t$ = exploration rate at turn t

MAB Algorithms

EXP3

- **Exp**onential-weight algorithm for **Exp**loration and **Exp**loitation
- Defines a stochastic policy as linear combination of Gibbs and uniform distributions

$$\pi_t(a) = (1 - K\mathcal{E}_t)\frac{\exp(\mathcal{E}_{t-1}\hat{R}_a)}{\sum_{a'}\exp(\mathcal{E}_{t-1}\hat{R}_{a'})} + \mathcal{E}_t$$

$\mathcal{E}_t$ = exploration rate at turn t

UCB1

- Upper Confidence Bound algorithm
- Assigns each arm an upper confidence bound for the reward

$$UCB_{a,t} = \hat{\mathcal{R}}_a + \sqrt{\frac{2\ln t}{n_a}}$$

- Samples greedily

$\sqrt{\frac{2\ln t}{n_a}}$ = exploration rate at turn t

Improving Few-Shot Generalization by Exploring and Exploiting Auxiliary Data

Presented at NeurIPS 2023

Alon Albalak

Colin Raffel

William Wang

Challenges of Few-Shot Learning

Goal: Train a model given a limited number of samples for a target task (e.g. 20 samples for coreference resolution)

Challenges:

- Learn structure of feature/label space
- Prevent overfitting to small sample size
- Need to interpolate/extrapolate between potentially massive gaps in sample space

Side note on extrapolation (IMPORTANT)

There are 2 types of scientist:

1. Those that can extrapolate from incomplete results

End side note

Challenges of Few-Shot Learning

Goal: Train a model given a limited number of samples for a target task (e.g. 20 samples for coreference resolution)

Challenges:

- Learn structure of feature/label space
- Prevent overfitting to small sample size
- Need to interpolate/extrapolate between potentially massive gaps in sample space

Few-Shot Learning

Few-shot target task

$\mathcal{D}_T$

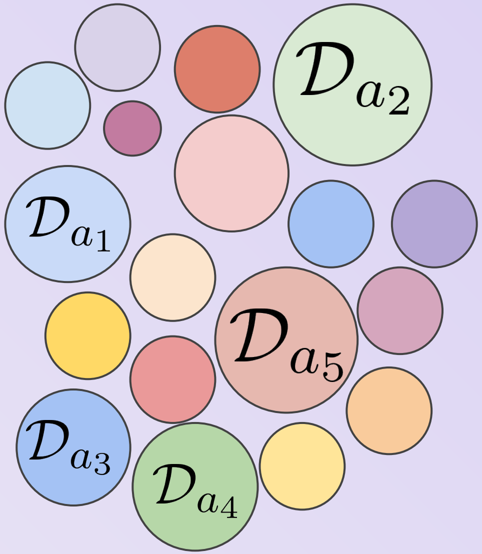
coreference
resolution

Few-Shot Learning with Auxiliary Data (FLAD)

Few-shot target task

coreference
resolution

Auxiliary Datasets



Failures of prior FLAD methods

- Assume all auxiliary data is relevant [1,2]
- Consider very small quantities of auxiliary data (1-3 datasets) [1,2,3,4]
- Determine auxiliary data based on pairwise interactions when using a single auxiliary dataset [3]
- Computation scales linearly (or worse) with number of auxiliary datasets

[1] Du et al. Adapting auxiliary losses using gradient similarity, 2020.
[2] Verboven et al. Hydalearn, 2022.
[3] Albalak et al. FETA: A benchmark for few-sample task transfer in open-domain dialogue, 2022.
[4] Chen et al. Weighted training for cross-task learning, 2022.

The Challenges of FLAD

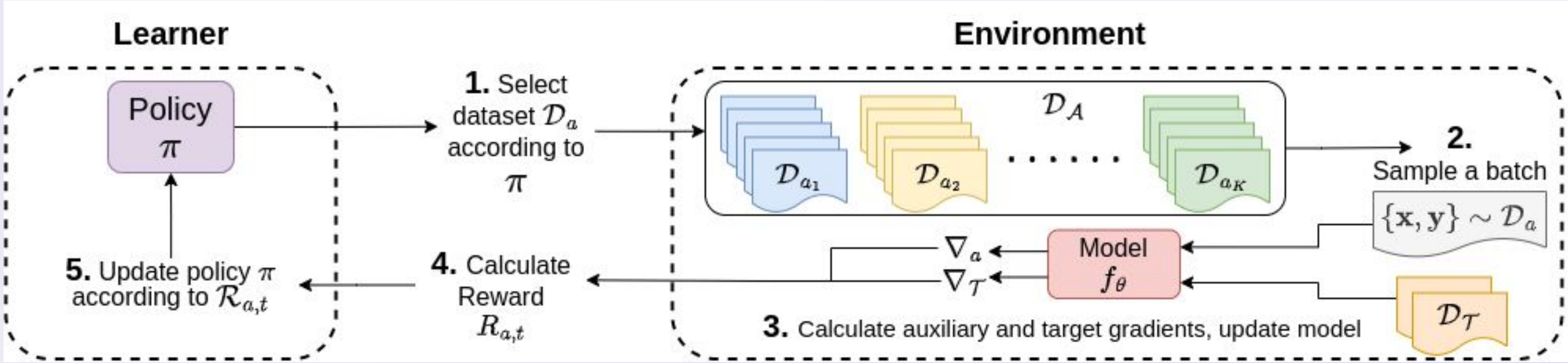
- How do we determine which auxiliary datasets will be helpful?
  - Manually comb through 100s of auxiliary datasets?
  - Train only on the most similar and discard the rest?
- How can we efficiently automate the decision-making process?

The Challenges of FLAD

- How do we determine which auxiliary datasets will be helpful?
  - Manually comb through 100s of auxiliary datasets?
  - Train only on the most similar and discard the rest?
- How can we efficiently automate the decision-making process?

With multi-armed bandits!

Our MAB-based FLAD Method



Designing Reward Functions

Gradient alignment	$\mathcal{R}^{GA} = \frac{\nabla_a \cdot \nabla_\mathcal{T}}{\|\nabla_a\|_2 \|\nabla_\mathcal{T}\|_2}$
Gradient magnitude similarity	$\mathcal{R}^{GMS} = \frac{2\|\nabla_a\|_2 \|\nabla_\mathcal{T}\|_2}{\|\nabla_a\|_2^2 + \|\nabla_\mathcal{T}\|_2^2}$
Aggregated reward	$\mathcal{R}^{AGG} = \frac{1+\mathcal{R}^{GA}}{2} + \mathcal{R}^{GMS}$

MAB Algorithms

EXP3

- **Exp**onential-weight algorithm for **Exp**loration and **Exp**loitation
- Defines a stochastic policy as linear combination of Gibbs and uniform distributions

$$\pi_t(a) = (1 - K\mathcal{E}_t)\frac{\exp(\mathcal{E}_{t-1}\hat{R}_a)}{\sum_{a'}\exp(\mathcal{E}_{t-1}\hat{R}_{a'})} + \mathcal{E}_t$$

$\mathcal{E}_t$ = exploration rate at turn t

UCB1

- Upper Confidence Bound algorithm
- Assigns each arm an upper confidence bound for the reward

$$UCB_{a,t} = \hat{\mathcal{R}}_a + \sqrt{\frac{2\ln t}{n_a}}$$

- Samples greedily

$\sqrt{\frac{2\ln t}{n_a}}$ = exploration rate at turn t

Experiments

Models: T5-LM and T0 (both 3B)

Experiments

Models: T5-LM and T0 (both 3B)

Target Datasets: T0Mix-eval (train a separate model for each)

- 11 datasets
  - **sentence completion** (COPA, HellaSwag, Story Cloze)
  - **NLI** (ANLI, CB, RTE)
  - **coreference resolution** (WSC, Winogrande)
  - **word sense disambiguation** (WiC)
- Only 20-70 samples for training, full evaluation

Experiments

Models: T5-LM and T0 (both 3B)

Target Datasets: T0Mix-eval (train a separate model for each)

- 11 datasets
    - **sentence completion** (COPA, HellaSwag, Story Cloze)
    - **NLI** (ANLI, CB, RTE)
    - **coreference resolution** (WSC, Winogrande)
    - **word sense disambiguation** (WiC)
- Only 20-70 samples for training, full evaluation

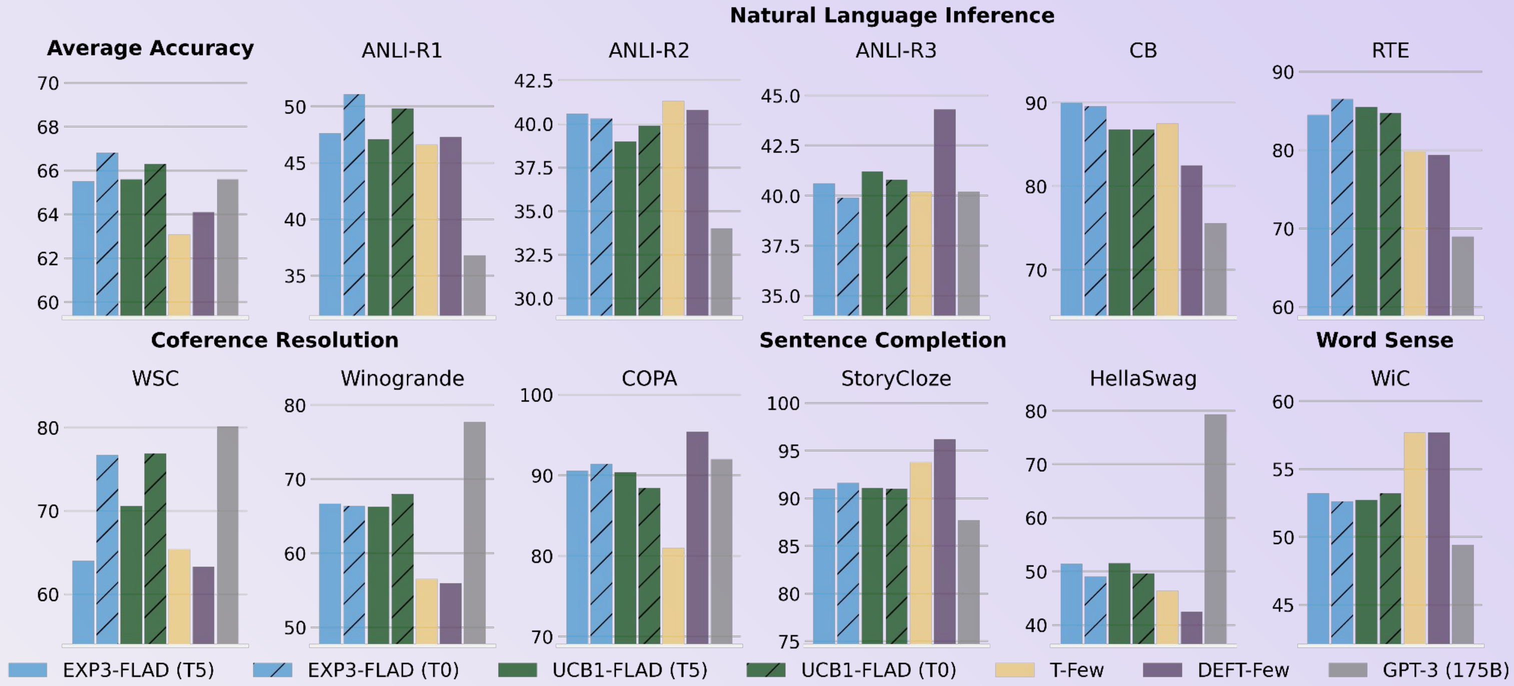
Auxiliary Datasets:

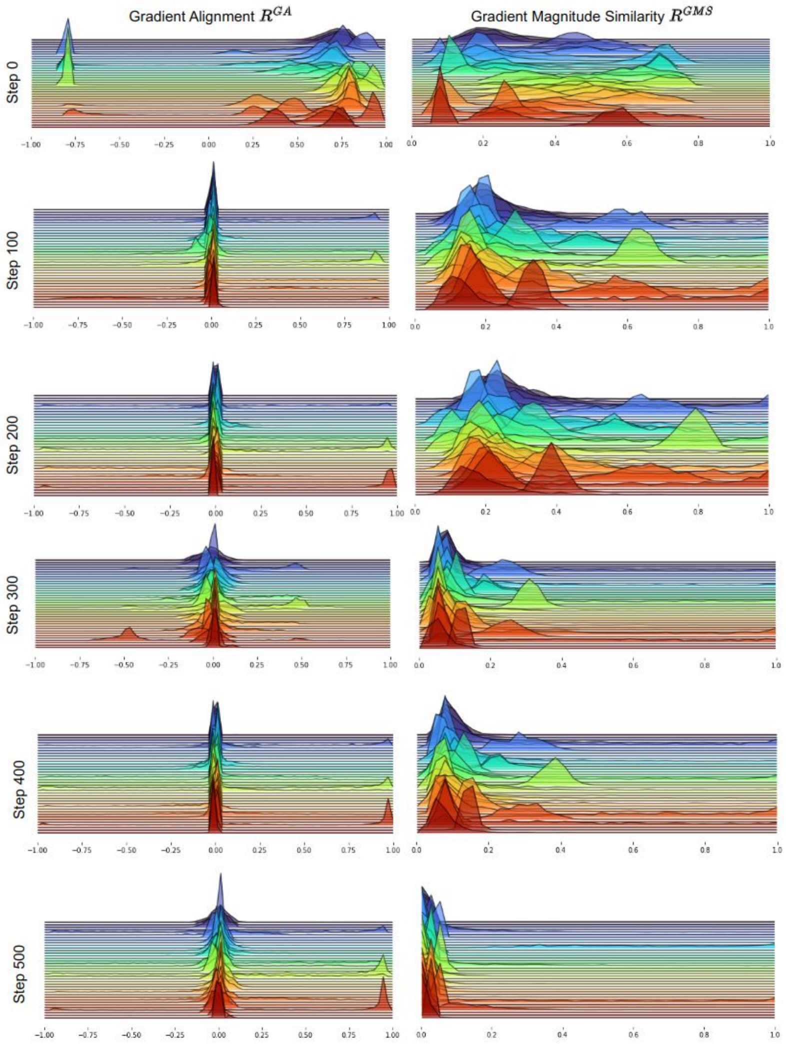
- T0Mix-train (35 datasets)
- P3 (260 datasets)

Results

Training Method \	**Base Model** / *Auxiliary Data*	**T5-XL** *T0Mix*	*P3*	**T0-3B** *T0Mix*	*P3*
Target-Only		52.82		56.44	
Explore-Only [8]		59.18	60.64	61.17	62.77
Exploit-Only [8]		59.79	60.49	60.87	62.87
EXP3-FLAD ($\mathcal{R}^{AGG}$)		<u>62.05</u>	<u>65.47</u>	<u>62.84</u>	**66.84**
UCB1-FLAD ($\mathcal{R}^{AGG}$)		**62.08**	**65.63**	**62.93**	66.29

Results


Natural Language Inference
Average Accuracy
ANLI-R1
ANLI-R2
ANLI-R3
CB
RTE
Coference Resolution
Sentence Completion
Word Sense
WSC
Winogrande
COPA
StoryCloze
HellaSwag
WiC
EXP3-FLAD (T5)
EXP3-FLAD (T0)
UCB1-FLAD (T5)
UCB1-FLAD (T0)
T-Few
DEFT-Few
GPT-3 (175B)


Gradient Alignment R^{GA}
Gradient Magnitude Similarity R^{GMS}
Step 0
Step 100
Step 200
Step 300
Step 400
Step 500

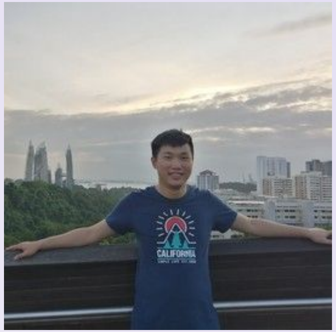
Takeaways

1. Our methods lead to 3**B models that outperform GPT-3 175B** and SOTA few-shot methods
2. Our MAB-based FLAD methods demonstrate **improved auxiliary dataset scaling** over existing FLAD methods
3. The **combination of exploration and exploitation is crucial** in determining the sampling policy
4. Our **gradient-based rewards add minimal computational overhead**, leading to very efficient algorithms that can scale to 100x more auxiliary datasets than previous FLAD methods

Efficient Online Data Mixing For Language Model Pre-Training

Alon Albalak

Liangming Pan



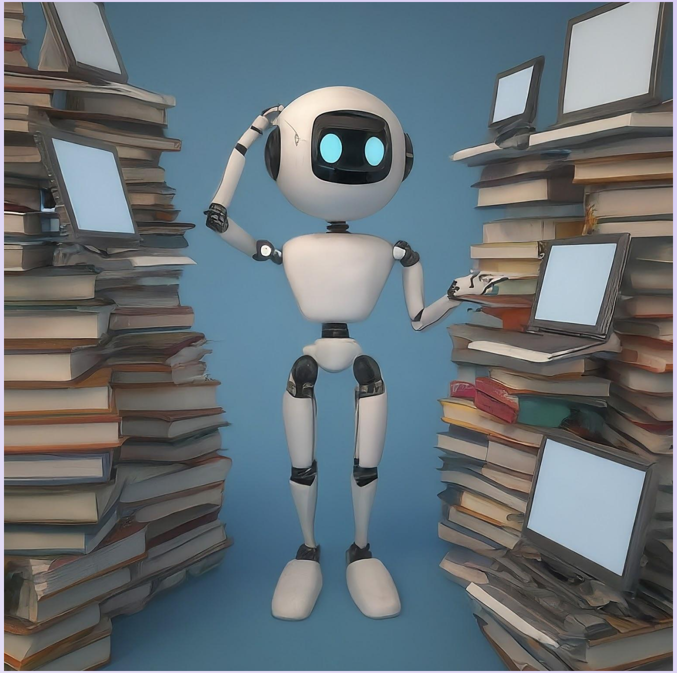
Colin Raffel

William Wang

Data Mixing for LLM Pretraining

- LLMs are often pretrained with data from multiple domains (wikipedia, Github, mathematics, etc.)

- Data mixing proportions are typically fixed prior to training, and do not adapt to training dynamics



Motivation for Online Data Mixing (ODM)

- Best prior method, DoReMi, optimizes for the best worst-case domain performance, but requires training 3 models minimum

- **Motivation**: The goal of pretraining is for a model to absorb large quantities of information.

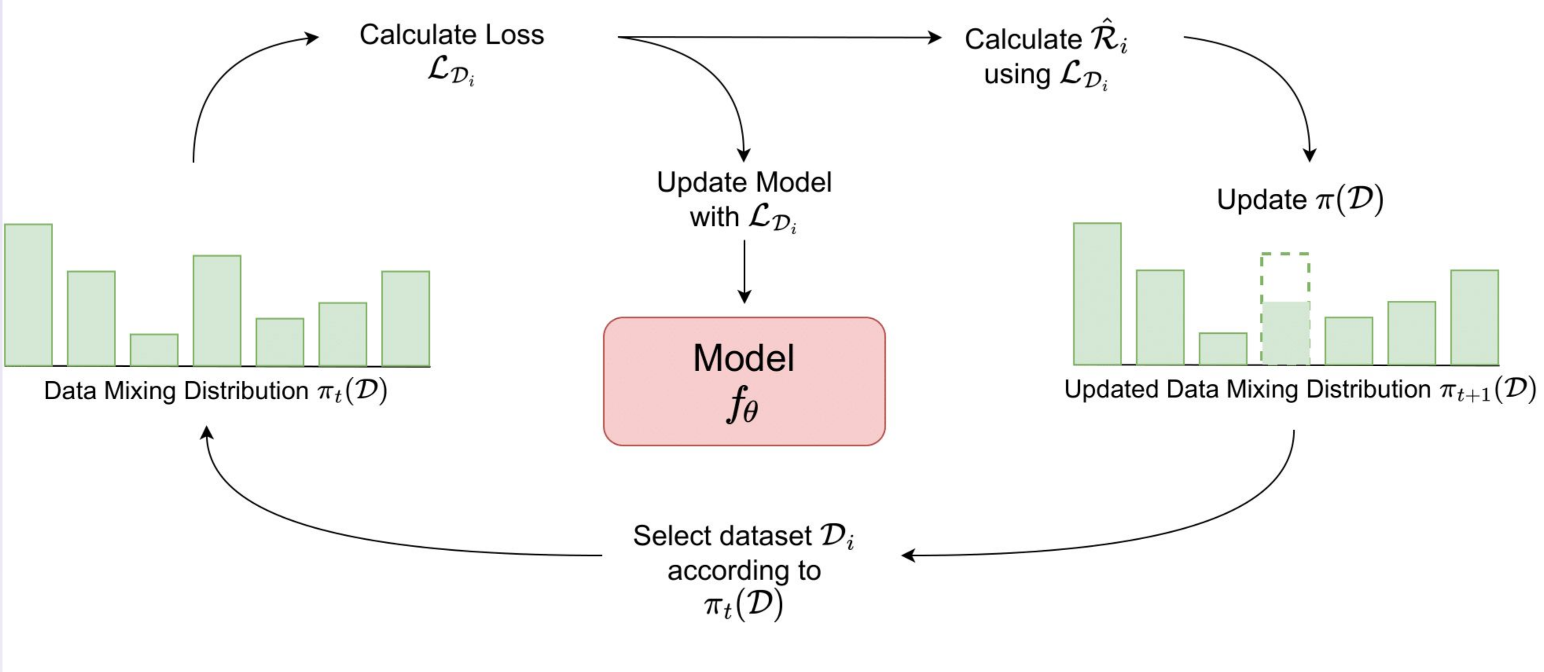
- **Goal**: Can we develop an efficient online data mixing (ODM) algorithm that maximizes the information a model contains?

Online Data Mixing (ODM) Method

- Formulate ODM as a multi-armed bandit (**MAB**) using a variation of the Exp3 algorithm
  - At each turn, the data mixing policy (π) is defined as a Gibbs distribution of importance-weighted rewards mixed with a uniform distribution (to allow exploration)
  - Different from FLAD, ODM updates the expected reward as an exponential moving average (instead of a cumulative reward)

- **Reward - Information Gain**:
  - Directly proportional to the entropy/perplexity
  - Reward for each domain is calculated directly as the loss

Online Data Mixing Method



Online Data Mixing Method

Algorithm 1 Online Data Mixing (ODM)

Require: $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_K\}$: Grouped dataset
Require: f_θ: Parameterized model
Require: $\mathcal{L}$: Loss function
Require: G: Gradient accumulation steps

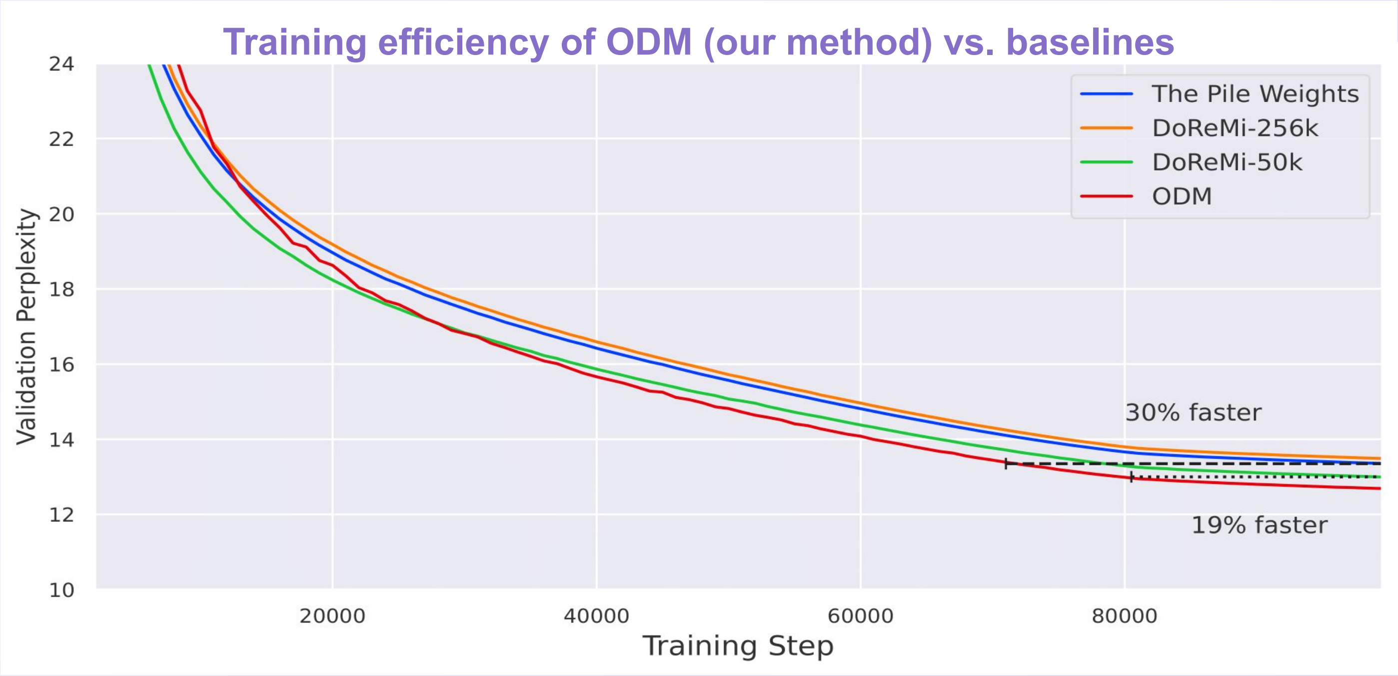
Initialize: $K = |\mathcal{D}|; \quad \mathcal{E}_0 = \frac{1}{K}; \quad \forall i \in \{1, \ldots, K\} : \hat{R}_i = 0$
for $t = 1, 2, \ldots, N$ **do**
  $\mathcal{E}_t = \min\left\{\frac{1}{K}, \sqrt{\frac{\ln K}{K \cdot t}}\right\}$ ▷ Update the exploration rate
  $\pi(\mathcal{D}) : \pi(\mathcal{D}_i) \leftarrow (1 - K\mathcal{E}_t)\frac{\exp(\mathcal{E}_{t-1}\hat{R}_i)}{\sum_j \exp(\mathcal{E}_{t-1}\hat{R}_j)} + \mathcal{E}_t$ ▷ Calculate the mixing distribution
  $\forall i = 1, 2, \ldots, K : \mathcal{L}_{\mathcal{D}_i} = 0$ ▷ Reset group losses
  for $g = 1, 2, \ldots, G$ **do**
    Sample $\mathcal{D}_i \sim \pi(\mathcal{D})$ and sample a batch $\{\mathbf{x}, \mathbf{y}\}$ from $\mathcal{D}_i$
    $\mathcal{L}_{\mathcal{D}_i} \leftarrow \mathcal{L}_{\mathcal{D}_i} + \mathcal{L}(f_\theta, \mathbf{x}, \mathbf{y})$ ▷ Record group losses for reward updates
  end for
  Update model parameters w.r.t $\sum_i \nabla_\theta \mathcal{L}_{\mathcal{D}_i}$
  for $i \in \{1, \ldots, K\}$ where $\mathcal{L}_{\mathcal{D}_i} \neq 0$ **do**
    $\hat{R}_i \leftarrow \alpha\hat{R}_i + (1 - \alpha)\frac{\mathcal{L}_{\mathcal{D}_i}}{\pi(\mathcal{D}_i)}$ ▷ Update estimated rewards
  end for
end for

Experimental Setup

- **Data**: The Pile with 22 domains (50 billion tokens)
- **Models**: 1 billion parameter decoder-only LM
- **Evaluation** (metrics):
  - The Pile test set (perplexity)
  - 5-shot MMLU (accuracy)
- **Baselines**:
  - The Pile Weights (TPW)
  - DoReMi-256k calculated on a 256k vocabulary tokenizer
  - DoReMi-50k calculated on same tokenizer as our model

Results

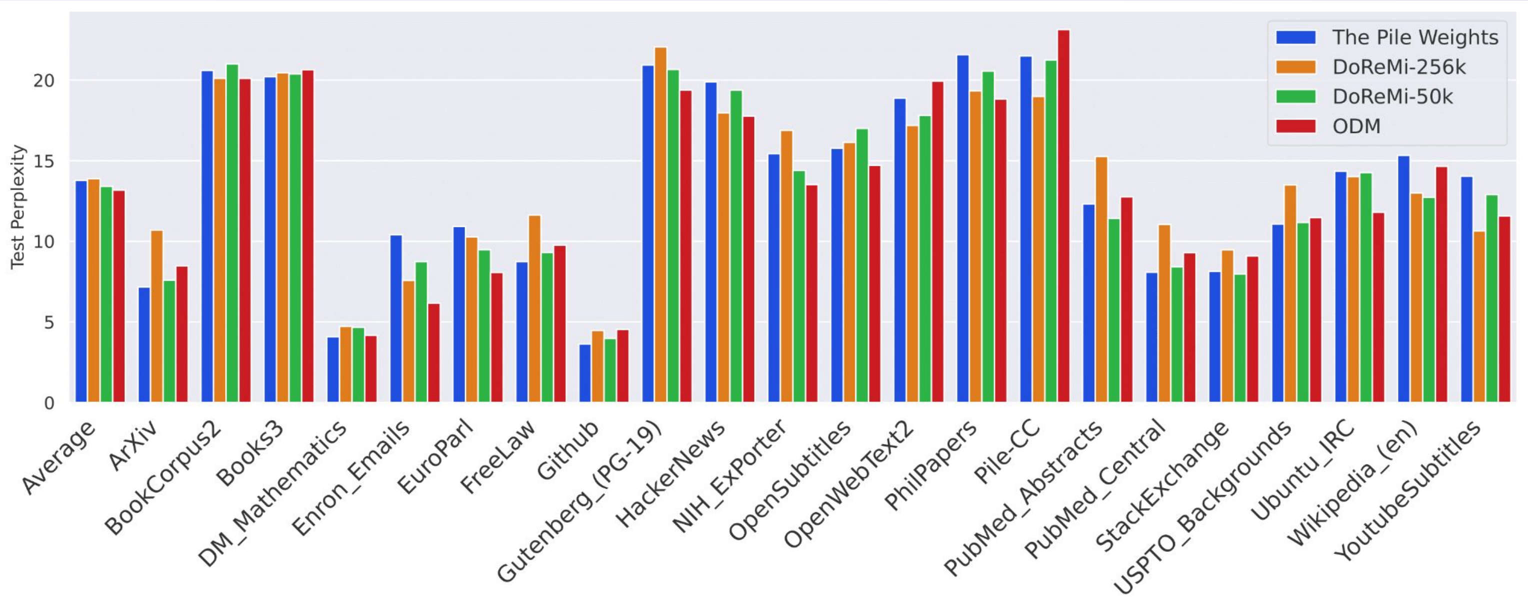
Online Data Mixing (ODM) improves training efficiency, requires 19% fewer iterations to reach the final validation perplexity of the next best method



Domain-wise comparison

ODM performs best on 9/22 domains, worst on Books, Github, and web text

DoReMi performs best on only 3 domains, worst on 2



Results

ODM leads to better downstream performance on MMLU, improves over TPW by 3% and DoReMi-50k by 1.9%

Method	Accuracy
The Pile Weights	0.27469
DoReMi-256k	0.27596
DoReMi-50k	0.27887
ODM	0.28416

Table 1: **Average 5-shot accuracy on MMLU**

ODM adds only ~0.00007% overhead!

Takeaways

- This version of ODM is not the ultimate solution, but it is a **proof of concept that data mixing can be done online, efficiently**

Future Research Directions

1. Data-Centric
2. Bigger Picture (moving beyond data)

Data-centric Research Directions

- Make data research more accessible (lower the barrier to entry)
  - Methods of directly measuring data
  - Scaling down
- Extending data mixing methods to individual data points
- Extending data selection methods to low-resource languages
- Improving data diversity in areas where it’s most needed (e.g. alignment)
- Understanding models from a data-centric perspective
  - How can we maintain good memorization (e.g. facts), but remove bad memorization (e.g. PII)

Bigger Picture Directions

- Move beyond siloed data research
  - Combine the 3 components (architecture, optimization, data)
  - Systems research (multiple models, multiple optimization objectives, data for multiple purposes)
- Combining data-centric and neuro-symbolic directions
  - Multi-component systems can potentially solve more abstract problems
  - Multi-component systems are also more interpretable
- Humans + Machines
  - Understanding how/when models fail allows us to give them “feedback” through their training data
  - Optimization objectives don’t care about societal impacts or unexpected side-effects, so humans need to

Questions?

- ➢ Improving Few-Shot Generalization by Exploring and Exploiting Auxiliary Data
- ➢ Efficient Online Data Mixing For Language Model Pretraining