Improving Few-Shot Generalization by Exploring and Exploiting Auxiliary Data

Anonymous Authors[1]

Abstract

Few-shot learning involves learning an effective model from only a few labeled datapoints. The use of a small training set makes it difficult to avoid overfitting but also makes few-shot learning applicable to many important real-world settings. In this work, we focus on **F**ew-shot **L**earning with **A**uxiliary **D**ata (FLAD), a training paradigm that assumes access to auxiliary data during few-shot learning in hopes of improving generalization. Introducing auxiliary data during few-shot learning leads to essential design choices where hand-designed heuristics can lead to sub-optimal performance. In this work, we focus on automated sampling strategies for FLAD and relate them to the explore-exploit dilemma that is central in multi-armed bandit settings. Based on this connection we propose two algorithms – EXP3-FLAD and UCB1-FLAD – and compare them with methods that either explore or exploit, finding that the combination of exploration *and* exploitation is crucial. Using our proposed algorithms to train T5 yields a 9% absolute improvement over the explicitly multi-task pre-trained T0 model across 11 datasets. All code will be publicly released.

1. Introduction

Few-shot learning is an attractive learning setting for many reasons: it promises efficiency in cost and time, and in some scenarios data is simply not available due to privacy concerns or the nature of the problem. However, few-shot learning is also a challenging setting that requires a delicate balance between learning the structure of the feature and label spaces while preventing overfitting to the limited training samples (Ravi & Larochelle, 2017; Wang et al., 2020; Parnami & Lee, 2022). One approach to improving the generalizability of models in the few-shot setting is **F**ew-shot **L**earning with **A**uxiliary **D**ata (FLAD), where auxiliary data is used to improve generalization on a target few-shot task (Wu & Dietterich, 2004; Esfandiarpoor et al., 2020).

However, FLAD methods can introduce their own challenges, including increased algorithmic and computational complexity. Specifically, incorporating auxiliary data during training introduces a large space of design choices for FLAD algorithms (e.g. how and when to train on auxiliary data). Manually designing the curriculum for training on large quantities of auxiliary data is not feasible due to the combinatorially large search space, and hand-picking which auxiliary data to use (e.g. in-domain or on-task) may lead to sub-optimal results (see section 6.2 from Albalak et al. (2022)). Delegating such choices to an algorithm can lead to better solutions, as demonstrated in the transfer learning (Vu et al., 2020; Pruksachatkun et al., 2020), meta-learning (Thrun & Pratt, 1998; Bansal et al., 2020), and multi-task literature (Wu et al., 2020; Aghajanyan et al., 2021). However, such algorithmic design choices require additional computation, motivating the search for efficient methods as the quantity of auxiliary data grows.

In this work, we consider the FLAD setting where auxiliary data is divided into separate datasets and training occurs simultaneously over both the target and auxiliary datasets. We desire a FLAD algorithm that

1. makes no assumptions on available auxiliary data a-priori (in-domain, on-task, quality, quantity, etc.),
2. continuously updates beliefs on importance of auxiliary data, and
3. adds minimal memory and computational overhead.

In this work, we design algorithms that satisfy our desiderata by drawing inspiration from the central problem in multi-armed bandit (MAB) settings: the exploration-exploitation trade-off (Macready & Wolpert, 1998; Simpkins et al., 2008). We relate the set of auxiliary datasets to the arms of a MAB and tailor the classic EXP3 (Auer et al., 2002b) and UCB1 (Auer et al., 2002a) algorithms to fit the FLAD framework by designing an efficient gradient-based reward signal. Figure 1 provides a basic illustration of how we formulate FLAD as an MAB problem.

Our study presents two efficient algorithms, EXP3-FLAD and UCB1-FLAD, which enhance the few-shot generaliza-

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

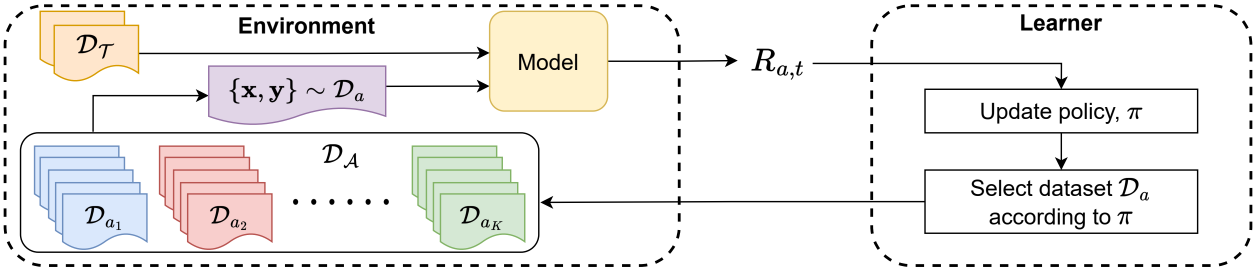

Figure 1. Overview of our formulation of few-shot learning with auxiliary data as a multi-armed bandit problem, as described in Section 3.

tion capabilities of a machine learning model. To empirically validate our approaches, we utilize readily available auxiliary datasets from P3 (Bach et al., 2022) which may be in- or out-of-domain and related or unrelated to the target task. We evaluate our methods on the same held-out tasks as T0 (Sanh et al., 2022) and show that, when using the same collection of auxiliary datasets, our algorithms outperform T0 (which simply concatenates the constituent datasets of P3) by 5.1% (EXP3-FLAD) and 5.6% (UCB1-FLAD) absolute. Furthermore, incorporating all available datasets in P3 (i.e. not just those used to train T0) increases the improvement to 7.6% and 9.1%. Finally, we compare models trained with our methods against strong few-shot models, finding that our methods improve performance by about 2%, even though one model utilizes 1,000 unlabeled target samples. Furthermore, to the best of our knowledge, our methods lead to the first 3 billion parameter model that improves over 175B GPT-3 using few-shot in-context learning.

In summary, our main contributions are:

- We connect FLAD to the MAB setting and focus on the exploration-exploitation trade-off.
- We design two algorithms, EXP3-FLAD and UCB1-FLAD that adapt existing MAB methods to the FLAD setting and propose a reward function that is simple and efficient (in both space and computational costs).
- We empirically validate that our methods improve few-shot performance of pretrained language models and show that strategies that employ only exploration or exploitation lead to sub-optimal performance.
- We perform two case studies to better understand why EXP3-FLAD and UCB1-FLAD outperform baselines.

2. Background

In this section we first informally describe the few-shot learning with auxiliary data (FLAD) problem. Then, we present the goals of the multi-armed bandit setting. Next, we present the adversarial bandit setting, connect it to FLAD, and review the EXP3 algorithm (Auer et al., 2002b) used to solve it. Finally, we present the UCB1 algorithm (Auer et al., 2002a) used to solve a more constrained MAB setting.

2.1. Few-shot Learning with Auxiliary Data

Few-shot learning with auxiliary data (**FLAD**) aims to improve generalization when training on a small quantity of target data $\mathcal{D}_\mathcal{T}$ by making use of a much larger quantity of (possibly) related auxiliary data $\mathcal{D}_\mathcal{A}$. The auxiliary data may be labeled or unlabeled, be cleanly differentiated as separate datasets/tasks, or have other properties. The goal of FLAD is to optimize a model for the distribution underlying $\mathcal{D}_\mathcal{T}$.

Under this definition, a great deal of prior work can be seen to focus on FLAD. Some past work sets up training in stages, such as STILTS (Phang et al., 2018), meta-learning (Bansal et al., 2020), or multi-task learning (Aghajanyan et al., 2021), where the first stage generally uses all of the auxiliary data equally and the second stage includes only the target data. Other approaches have utilized retrieval-augmented models that match auxiliary data to large quantities of unlabeled target data such as in ReCross (Lin et al., 2022) and DEFT (Ivison et al., 2022). Alternatively, some works scale the loss of auxiliary datasets (Verboven et al., 2022), or include unsupervised auxiliary data (Dery et al., 2022).

To narrow down the set of methods that we consider for satisfying our desiderata, we focus on the setting where – crucially – the final stage of training includes simultaneous training on both supervised auxiliary and target data. Henceforth, we use FLAD to refer specifically to this setting.

2.2. Multi-Armed Bandits

The Multi-Armed Bandit (**MAB**) setting is a problem from machine learning where a learner interacts with an environment over N rounds by following a policy π. At each round t the learner chooses one of the environment's K arms, $a \in \mathcal{A}$ where $K = |\mathcal{A}|$, after which the environment provides a reward R_t. Rewards for unplayed arms are not observed. The goal of the learner is to adopt a policy π that selects actions that lead to the largest cumulative reward over N rounds, $R = \sum_{t=1}^{N} R_t$. In this work we assume a finite K and that the underlying reward distribution of each arm may have a variety of properties (e.g. stochasticity or stationarity) depending on the exact scenario, leading to different optimal policies (Lattimore & Szepesvári, 2020).

Adversarial MAB The adversarial MAB setting assumes that the reward-generating process is controlled by an adversary. This assumption allows for modelling non-stationary and highly stochastic reward signals. We will later show why our FLAD formulation fits into this setting. Under this setting, it is assumed that an adversary is given access to the learner's policy π and determines the sequence of rewards, $(R_{a,t})_{t=1}^{N}$, for each arm prior to play (Auer et al., 1995). At each turn π determines a distribution over actions, $p(\mathcal{A})$, and an action is sampled from the distribution, $a \sim p(\mathcal{A})$. See Lattimore & Szepesvári (2020) for further details.

The EXP3 Algorithm The EXP3 algorithm ("Exponential-weight algorithm for Exploration and Exploitation") targets the adversarial multi-armed bandit problem Auer et al. (2002b) by choosing arms according to a Gibbs distribution based on the empirically determined importance-weighted rewards of arms. To allow for exploration, EXP3 mixes the Gibbs distribution with a uniform distribution.

Formally, let the exploration rate be $\gamma \in (0, 1]$. At round t, π defines the probability of selecting a given arm, $a \in \mathcal{A}$, as a linear combination of Gibbs and uniform distributions

$$p_t(a) = (1-\gamma)\frac{\exp(\gamma \hat{R}_{a,t-1}/K)}{\sum_{a\prime} \exp(\gamma \hat{R}_{a\prime,t-1}/K)} + \frac{\gamma}{K} \quad (1)$$

where the importance weighted reward $\hat{R}_{a,t}$ is calculated as

$$\hat{R}_{a,t} = \hat{R}_{a,t-1} + \frac{R_{a,t}}{p_{t-1}(a)} \quad (2)$$

and $R_{a,t}$ denotes the observed reward. All unplayed arms, $a\prime \neq a$ have unchanged importance weighted rewards; $\hat{R}_{a\prime,t} = \hat{R}_{a\prime,t-1}$.

Algorithmically, EXP3 takes the following steps at each round: First, calculate the sampling distribution p_t and sample an arm from the distribution. Then a reward $R_{a,t}$ is observed and the algorithm updates the importance weighted reward $\hat{R}_{a,t}$ for the played arm.

Informally, the use of an importance-weighted estimated reward compensates the rewards of actions that are less likely to be chosen, guaranteeing that the expected estimated reward is equal to the actual reward for each action. EXP3 is designed to be nearly optimal in the worst case, but due to the exploration rate it will select "bad" actions at a rate of γ/K. The exploration of EXP3 combined with importance-weighting allows the policy to handle non-stationary reward-generating processes.

The UCB1 Algorithm While the adversarial setting makes almost no assumptions about the reward-generating process and therefore maintains its performance guarantees under almost any circumstances, it can be outperformed in settings that *are* constrained. In this section we assume that the reward-generating processes are stationary Gaussian distributions. A common policy used to solve this MAB setting is the Upper Confidence Bound (UCB1) algorithm, which assigns each arm a value called the upper confidence bound based on Hoeffding's inequality (Auer et al., 2002a). The UCB1 algorithm is based on the principle of *optimism in the face of uncertainty*, meaning that with high probability the upper confidence bound assigned to each arm is an overestimate of the unknown mean reward.

Formally, let the estimated mean reward of arm a after being played n_a times be $\hat{R}_a$ and the true mean reward be R_a, then

$$\mathbb{P}\left(R_a \geq \hat{R}_a + \sqrt{\frac{2\ln(1/\delta)}{n_a}}\right) \leq \delta \quad \forall \delta \in (0,1)$$

derived from Hoeffding's inequality (following equation 7.1 of Lattimore & Szepesvári (2020)), where δ is the confidence level that quantifies the degree of certainty in the arm. In this work we let $\delta = 1/t$ where t is the number of rounds played, shrinking the confidence bound over rounds. Thus, we define the upper confidence bound for arm a at turn t as

$$UCB_{a,t} = \begin{cases} \infty, & \text{if } n_a = 0 \\ \hat{R}_a + \sqrt{\frac{2\ln t}{n_a}}, & \text{otherwise} \end{cases} \quad (3)$$

Algorithmically, UCB1 takes the following steps at each round. First, the UCB1 policy plays the arm with largest upper confidence bound, $a^* = \arg\max_{a\in\mathcal{A}} UCB_{a,t}$. Next, a reward $R_{a^*,t}$ is observed and the algorithm updates $\hat{R}_{a^*}$ (the estimated mean reward for a^*) and the upper confidence bounds for all a. Informally, this algorithm suggests that the learner should play arms more often if they either 1. have large expected reward, $\hat{R}$, or 2. n_a is small because the arm is not well explored.

3. From MAB to FLAD

In this work we formulate few-shot learning with auxiliary data (FLAD) as a multi-armed bandit (MAB) problem as follows. Assume auxiliary data $\mathcal{D}_{\mathcal{A}}$ is split into individual datasets $\mathcal{D}_a$ and at each round of training a learner updates it's policy π over the auxiliary datasets. Assume that at each round a batch of data is sampled from a single dataset $\mathcal{D}_a$ selected by the policy, i.e. $\{\mathbf{x}, \mathbf{y}\} \sim \mathcal{D}_a$. Then, the model being trained f_θ is updated through a gradient w.r.t. θ calculated over the sampled batch using a task-appropriate loss function as $\nabla_a = \nabla_\theta \mathcal{L}(f_\theta, \mathbf{x}, \mathbf{y})$. For practical purposes of optimizing large neural networks with stochastic gradient descent-based methods, we let G be the number of rounds between model updates to allow for varying batch sizes and for multiple auxiliary datasets to occur in a single model update. For simplicity, after every G rounds the model also computes the target gradient, $\nabla_{\mathcal{T}} = \nabla_\theta \mathcal{L}(f_\theta, \mathcal{D}_{\mathcal{T}})$, and updates model parameters w.r.t. $\nabla_{\mathcal{T}} + \sum_{a\in\mathcal{A}} \nabla_a$.

Designing the Reward Function We design the reward function with our desiderata in mind. To ensure that our algorithm adds minimal memory and computational overhead we consider rewards that utilize information intrinsic to the model and the losses being optimized, not an external metric (e.g. accuracy or BLEU).

In this work we adopt a gradient-based reward inspired by previous works (Du et al., 2018; Lin et al., 2019; Yu et al., 2020; Wang et al., 2021). Formally, at turn t let $\mathcal{D}_a$ be the auxiliary dataset selected by the learner, then we define our reward function as

$$R_{a,t} = \frac{\nabla_a \cdot \nabla_{\mathcal{T}}}{\|\nabla_a\| \|\nabla_{\mathcal{T}}\|} \tag{4}$$

i.e. the cosine similarity between the gradients of the sampled auxiliary dataset batch and the whole target dataset. This reward adds minimal computational complexity at each round (three vector dot products, two square roots, and two scalar multiplications). Additionally, while this reward adds space requirements, we show later that this can be reduced to at most $(G+1) * 2.3\%$ of the full model parameters.

3.1. Modelling Assumptions, Implications, and Consequences

FLAD as an Adversary First, the formulation we follow necessitates modelling the environment as an adversary. This is due to three factors: the highly non-convex loss landscape of deep neural networks, the use of stochastic gradient descent-based optimization, and our use of gradient alignment as a reward function. These factors imply that the rewards for each arm cannot be guaranteed to be stationary, independent, or gaussian. However, we will later empirically show that gradient alignments maintain a weak form of stationarity.

Parameter Sharing Next, our reward function requires that the model f shares parameters across datasets. We believe this is not overly restrictive due to the trend of unifying input formats for single models to handle many tasks (e.g. text-to-text (McCann et al., 2018; Radford et al., 2019; Raffel et al., 2020; Brown et al., 2020) and multi-modal settings (Wang et al., 2022; Alayrac et al., 2022)).

Mini-Batch Reward Calculation Additionally, to improve efficiency, we assume that the gradient of a single mini-batch from an auxiliary dataset is a reasonable approximation of the gradient w.r.t. the full dataset. We believe this is a reasonable assumption because approximation errors tend to balance each other out over the course of many sampling steps. Relatedly, mini-batch SGD has proven to improve training efficiency (LeCun et al., 2012).

Discrepancy Between Reward and Evaluation Metric Finally, the success of a policy in traditional bandit settings is directly measured by the reward being optimized. However, in our setting the reward we receive from playing an arm may not be the same metric that we aim to optimize (e.g. cross-entropy, accuracy, or ROUGE-L).

4. Adapting MAB algorithms for FLAD

In this section we describe our variations on the EXP3 and UCB1 algorithms based on how they do and do not fit the FLAD setting. We present pseudo-code for our methods in Algorithms 1 (EXP3-FLAD) and 2 (UCB1-FLAD).

4.1. EXP3 for FLAD

Recall that the EXP3 algorithm is designed for the adversarial MAB setting, where no assumptions are made on the reward-generation process and in Section 3.1 we argue that the adversarial MAB setting is an appropriate modelling choice for FLAD. However, in FLAD we can make some assumptions to weaken the power of the adversary and improve the quality of our learner.

First, we assume that the helpfulness or harmfulness of training on a particular auxiliary dataset (in terms of the impact on target dataset performance) will be relatively consistent throughout training. Of course, because we randomly sample batches during training, the reward-generating process is noisy and we cannot know for sure early on whether a dataset is actually helpful or harmful. To model this assumption, we use a decaying exploration rate, as proposed by Seldin et al. (2013). Recall that K is the number of auxiliary datasets. Then at the beginning of round t let the exploration rate be

$$\mathcal{E}_t = \min\Bigl\{\frac{1}{K}, \sqrt{\frac{\ln K}{K \cdot t}}\Bigr\} \tag{5}$$

Further, we adjust the sampling distribution from equation 1 to include the new exploration rate as

$$p(a) = (1 - K\mathcal{E}_t)\frac{\exp(\mathcal{E}_{t-1}\hat{R}_a)}{\sum_{a'} \exp(\mathcal{E}_{t-1}R_{a'})} + \mathcal{E}_t \tag{6}$$

Under our assumptions, this decaying rate now allows for an initial exploratory period after which the learner will select "bad" actions at a rate of $\mathcal{E}_t = \sqrt{\frac{\ln K}{K \cdot t}}$.

In our setting, negative rewards are possible due to the use of cosine similarity to measure gradient alignment in our reward function. Along with our first assumption (harmful auxiliary data will be harmful throughout training), we believe this leads to reasonable actions in the learner. In the original EXP3 algorithm, the use of an importance-weighted estimated reward compensates the rewards of actions that are less likely to be chosen if their empirical reward is unexpectedly high. In our variation, we allow for a similar phenomenon to occur when an empirical reward is expectedly low. Specifically, if the probability of the selected

Algorithm 1 EXP3-FLAD

Require: $\mathcal{D}_{\mathcal{A}}, \mathcal{D}_{\mathcal{T}}$: Auxiliary and target datasets
Require: f_θ: Parameterized model
Require: G: Gradient accumulation steps
1: **Initialize**: $K = |\mathcal{A}|$; $\mathcal{E}_0 = \frac{1}{K}$;
$\forall a \in \mathcal{A} : \nabla_a = 0, \hat{R}_a = 1$
2: **for** $t = 1, 2, \ldots, N$ **do**
3: $\mathcal{E}_t = \min\left\{\frac{1}{K}, \sqrt{\frac{\ln K}{K \cdot t}}\right\}$
4: $\forall a \in \mathcal{A} : p(a) \leftarrow (1 - K\mathcal{E}_t)\frac{\exp(\mathcal{E}_{t-1}\hat{R}_a)}{\sum_{a'}\exp(\mathcal{E}_{t-1}\hat{R}_{a'})} + \mathcal{E}_t$
5: Sample $a \sim p(\mathcal{A})$ and batch $\{\mathbf{x}, \mathbf{y}\} \sim \mathcal{D}_a$
6: $\nabla_a \leftarrow \nabla_a + \nabla_\theta \mathcal{L}(f_\theta, \mathbf{x}, \mathbf{y})$
7: **if** $t \pmod{G} \equiv 0$ **then**
8: $\nabla_{\mathcal{T}} \leftarrow \nabla_\theta \mathcal{L}(f_\theta, \mathcal{D}_{\mathcal{T}})$
9: Update model parameters w.r.t.$\nabla_{\mathcal{T}} + \sum_a \nabla_a$
10: **for all** $\{a \in \mathcal{A} | \nabla_a \neq 0\}$ **do**
11: $\hat{R}_a \leftarrow \hat{R}_a + \frac{R_{a,t}}{p(a)}$
12: $\nabla_a \leftarrow 0$
13: **end for**
14: **end if**
15: **end for**

Algorithm 2 UCB1-FLAD

Require: $\mathcal{D}_{\mathcal{A}}, \mathcal{D}_{\mathcal{T}}$: Auxiliary and target datasets
Require: f_θ: Parameterized model
Require: G: Gradient accumulation steps
Require: β: Smoothing factor
1: **Initialize**:
$\forall a \in \mathcal{A} : n_a = 1,$
$\hat{R}_a = \cos(\nabla_\theta \mathcal{L}(f_\theta, \mathcal{D}_{\mathcal{T}}), \nabla_\theta \mathcal{L}(f_\theta, \mathcal{D}_a))$
2: **for** $t = 1, 2, \ldots, N$ **do**
3: $a^* = \underset{a \in \mathcal{A}}{\text{argmax}}\ \hat{R}_a + \sqrt{\frac{2 \ln t}{n_a}}$
4: Sample batch $\{\mathbf{x}, \mathbf{y}\} \sim \mathcal{D}_{a^*}$
5: $\nabla_{a^*} \leftarrow \nabla_{a^*} + \nabla_\theta \mathcal{L}(f_\theta, \mathbf{x}, \mathbf{y})$
6: $n_{a^*} \leftarrow n_{a^*} + 1$
7: **if** $t \pmod{G} \equiv 0$ **then**
8: $\nabla_{\mathcal{T}} \leftarrow \nabla_\theta \mathcal{L}(f_\theta, \mathcal{D}_{\mathcal{T}})$
9: Update model parameters w.r.t. $\nabla_{\mathcal{T}} + \sum_a \nabla_a$
10: **for all** $\{a \in \mathcal{A} | \nabla_a \neq 0\}$ **do**
11: $\hat{R}_a \leftarrow (1 - \beta)\hat{R}_a + \beta R_{a,t}$
12: $\nabla_a \leftarrow 0$
13: **end for**
14: **end if**
15: **end for**

dataset a is low and the learner receives a negative reward, then our learner was correct in its estimation of a low reward, further decreasing the importance of seeing dataset a in the future. In the original EXP3 the learner can push up on the probability of "good" arms, and in our variation we give it the ability to push down on the probability of "bad" datasets as well.

EXP3-FLAD Algorithm On each turn, the learner first computes the current exploration rate as in Equation 5. Then, the learner samples an auxiliary dataset from the distribution defined by Equation 6. Next, the learner samples a batch from the chosen auxiliary dataset and calculates the gradient. This process repeats G times, at which point the learner calculates the gradient of the target dataset and updates the model with all auxiliary and target gradients that have accumulated. Finally, the importance-weighted reward for each auxiliary batch is calculated as in Equation 2, using our cosine reward defined in equation 4. See Algorithm 1 for pseudo-code.

4.2. UCB1 for FLAD

While the EXP3 algorithm accounts for the stochastic, non-stationary reward-generating process in FLAD, UCB1 does not. To address this in UCB1-FLAD, we include an exponential moving average when estimating the mean reward for a given arm. Formally, if at turn t the learner selects auxiliary dataset $\mathcal{D}_a$, then we update the estimated mean reward $\hat{R}_a$ as

$$\hat{R}_a = (1 - \beta)\hat{R}_a + \beta R_{a,t} \tag{7}$$

where β is the smoothing factor and $R_{a,t}$ is the observed reward.

Furthermore, in the original MAB setting all interactions with the environment occur online, but FLAD is a unique situation where the learner can interact with the auxiliary data prior to training. To take advantage of this, rather than initializing estimated rewards with a single mini-batch, we propose to initialize them with larger data quantities to improve the approximation of the true dataset gradients (line 1 of Algorithm 2). This is done for each auxiliary dataset by calculating the gradient $\nabla_a = \nabla_\theta \mathcal{L}(f_\theta, \mathbf{x}, \mathbf{y})$, where the number of samples in $\{\mathbf{x}, \mathbf{y}\}$ is significantly larger than a mini-batch, and can be up to the size of the full dataset.

UCB1-FLAD Algorithm First, the estimated rewards $\hat{R}_a$ are initialized by approximating the gradient from each dataset as described in the previous paragraph. Then at each turn the learner computes the upper confidence bounds for each auxiliary dataset as in Equation 3. Next, the learner samples a batch of data from the dataset with highest upper confidence bound and calculates the gradient. This process repeats G times, at which point the learner calculates the gradient of the target dataset and updates the model with all auxiliary and target gradients that have accumulated. Finally, the smoothed estimated mean reward is calculated using Equation 7. See Algorithm 2 for pseudo-code.

5. Experimental Setup

Models For our experiments, we utilize encoder-decoder models from the T5 family of pre-trained models (Raffel et al., 2020). Specifically, we experiment with LM-adapted T5 (T5-LM) and T0. The T5-LM model, from Lester et al. (2021), further trains the T5.1.1 model for 100,000 steps (corresponding to 100B tokens) from the C4 dataset on prefix language modelling objective (i.e. predicting the following text based on a prefix). The T0 model was initialized from T5-LM and further trained on a multitask mixture of prompted datasets as described by Sanh et al. (2022). We repeat each experiment with the T5-LM XL (hereafter T5-XL) and T0-3B models (both using the same architecture with 2.85 billion parameters) as starting checkpoints (from Hugging Face Transformers (Wolf et al., 2020)).

Target Datasets We obtain all datasets from Hugging Face Datasets[1], and cast them to the text-to-text format by applying prompt templates from the Public Pool of Prompts (P3) (Bach et al., 2022) that was used to train T0. To evaluate our few-shot methods, we utilize the same held-out datasets as T0, which cover four distinct tasks: **sentence completion** (COPA (Gordon et al., 2012), HellaSwag (Zellers et al., 2019), Story Cloze (Sharma et al., 2018)), **natural language inference** (ANLI (Nie et al., 2020), CB (de Marneffe et al., 2019), RTE (Dagan et al., 2006)), **coreference resolution** (WSC (Levesque et al., 2012), Winogrande (Sakaguchi et al., 2020)), and **word sense disambiguation** (WiC (Pilehvar & Camacho-Collados, 2019)). For each dataset, we randomly sample five few-shot splits from their training data, containing the same number of training examples as previous works, between 20 to 70 (Brown et al., 2020; Liu et al., 2022). We further divide each split into equal training and validation partitions. Only ANLI datasets have a publicly available test set, so for all other datasets we evaluate models on the validation set (not utilized for few-shot training or validation). For all datasets, we report the mean and standard deviation of accuracy across splits.

Auxiliary Datasets We compare the performance of our methods using two sets of auxiliary data and never include any of the target datasets as part of auxiliary data. First, we use the collection of datasets used to train T0 (henceforth referred to as T0Mix), including 35 unique datasets covering the tasks of question answering, sentiment analysis, topic classification, summarization, paraphrase detection and structure-to-text. Second, we utilize all datasets in P3 (which forms a superset of T0Mix) and, to prevent data leakage, filter out those datasets that overlap with any target dataset, leading to 260 available datasets. For each auxiliary dataset, we use at most 10,000 of the dataset's examples.

Training Details For target-only fine-tuning, we use learning rates in $\{1e\text{-}4, 3e\text{-}4\}$. For all other methods, we always use a learning rate of $1e$-4. For target-only fine-tuning and FLAD baselines, we use batch sizes in $\{32, 128\}$. For EXP3-FLAD and UCB1-FLAD we use mini-batches of 8 samples, and let G be in $\{4, 16\}$ to match the batch size of other methods. For all experiments we use the Adafactor optimizer (Shazeer & Stern, 2018) and validation-based early stopping.

[1]https://huggingface.co/datasets

Table 1. Results of target-only fine-tuning and FLAD-based methods on our target datasets (i.e. the held out datasets from T0). Reported scores are mean accuracy and standard deviation over the full evaluation set.

Base Model	T5-XL		T0-3B	
Aux. Data	T0Mix	P3	T0Mix	P3
Target-Only	$52.82_{3.34}$		$56.44_{4.70}$	
Explore-Only	$59.18_{5.52}$	$60.64_{4.92}$	$61.17_{3.30}$	$62.77_{4.83}$
Exploit-Only	$59.79_{5.63}$	$60.49_{5.01}$	$60.87_{3.35}$	$62.87_{3.69}$
EXP3-FLAD	$61.50_{4.25}$	$64.07_{4.81}$	$62.87_{3.85}$	$\underline{65.98}_{3.20}$
UCB1-FLAD	$62.01_{3.89}$	$\underline{65.52}_{3.86}$	$62.89_{3.68}$	$\mathbf{66.29}_{3.29}$

In preliminary experiments we consider rewards using gradients from various model partitions: the full model weights, encoder-only weights, decoder-only weights, and the weights of the output vocabulary matrix (language modelling head). We find that using the parameters from the language modelling head provides the best performance and contains only 2.3% of the full model parameters, significantly reducing memory consumption. For UCB1-FLAD we found the smoothing factor $\beta = 0.9$ to work well in preliminary experiments and initialize auxiliary dataset gradient alignment using 1,000 samples.

More implementation details, including hyperparameters, can be found in Appendix A

6. Findings and Analyses

Now that we have designed and described the EXP3-FLAD and UCB1-FLAD algorithms, we first compare their performance with other FLAD methods. Aggregate scores are shown in Table 1 with detailed results in Table B of the Appendix. Then, we compare our methods with strong few-shot methods and show the full results in Figure 3.

FLAD Helps Models Generalize First, we compare our proposed methods with target-only fine-tuning (i.e. without using auxiliary data) as well as two baseline FLAD settings (explore-only and exploit-only). *Explore-only* is equivalent to multi-task training where auxiliary data is uniformly sampled, i.e. continuously exploring auxiliary data and never exploiting knowledge of its relation to the target data. *Exploit-only* computes gradient alignment prior to training, as in UCB1, followed by multi-task training where the sampling probabilities are defined by a Gibbs distribution over similarities (similar to that in EXP3) which results in always exploiting and never exploring. For both FLAD baselines we utilize mixed-dataset batches and a target dataset mixing ratio with the possible values of $\{1, 5, 10\}$ times the greatest auxiliary sampling probability.

Table 1 shows that *all* FLAD methods provide significant

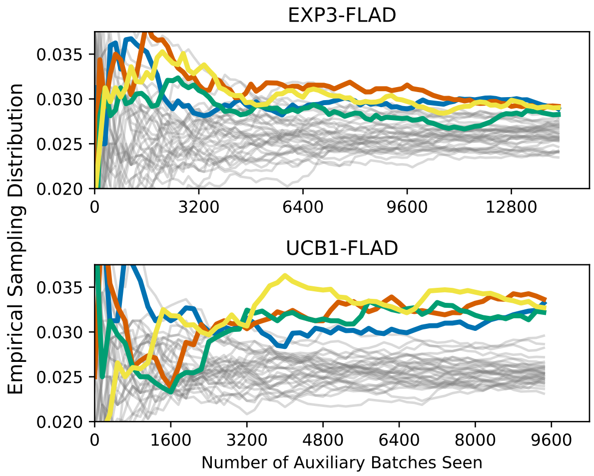

Figure 2. **Empirical Sampling Distributions of FLAD Methods**. A case study on RTE as the target dataset and T0Mixture as auxiliary data. Top-4 most sampled auxiliary datasets are highlighted with color. UCB1-FLAD forms a distribution with two clear peaks, while EXP3-FLAD forms a more flat distribution.

improvement in few-shot generalization over target-only fine-tuning, proving the utility of FLAD methods. We find that even the naive mixing approach, *explore-only*, improves performance by 4.8 - 8%, with the remaining approaches improving accuracy by up to 12.7%. Furthermore we find that, given identical training data, all single-stage FLAD methods on T5-XL + T0Mix (left column of Table 1) improve over the multi-task trained T0 with target-only fine-tuning (between 2.7 - 5.6% improvement), demonstrating how simultaneous training on auxiliary and target data, as in FLAD methods, can help models generalize in few-shot settings.

The Importance of Exploration *and* Exploitation Our experiments show that EXP3-FLAD and UCB1-FLAD consistently outperform the explore-only and exploit-only FLAD methods across both models and auxiliary datasets. Additionally, Table B of the appendix shows that our methods surpass the baselines in *every* evaluation task.

Furthermore, when comparing the ability of FLAD methods to leverage additional auxiliary data (i.e. going from T0Mix to all of P3), we find that the improvement for explore- and exploit-only methods is minimal with only 0.7-2% improvement. On the other hand, EXP3-FLAD and UCB1-FLAD show a notable improvement of 2.6-3.5%, emphasizing the importance of both exploration *and* exploitation, particularly when dealing with large collections of auxiliary data.

Comparing EXP3-FLAD and UCB1-FLAD In Section 3.1 we argue that adversarial MAB is an appropriate model of the FLAD setting, but our results empirically show that EXP3-FLAD is slightly outperformed by UCB1-FLAD. In this section we analyze each method's performance and training dynamics to understand this phenomenon.

To better understand the training dynamics, we perform a case study on T5-XL with T0Mix as the auxiliary data, with full details and figures in Appendix C. First we look at RTE, where UCB1-FLAD outperforms EXP3-FLAD. We calculate the empirical distribution of samples seen from each auxiliary dataset, shown in Figure 2, and find that EXP3-FLAD samples nearly uniformly from all datasets while UCB1-FLAD forms a bimodal sampling distribution with 2 very clear peaks. The lack of peakiness in the EXP3-FLAD distribution is counterintuitive, as we do find that it achieves separation between auxiliary tasks in the cumulative estimated reward (as shown in Figure 5), but this does not lead to separation in the sampling probability space. Additionally we find that even on COPA, where EXP3-FLAD outperforms UCB1-FLAD, EXP3-FLAD still achieves good separation between cumulative estimated rewards, but has a monomodal sampling distribution, while UCB1-FLAD does not have as clear of a bimodal distribution as in RTE.

The difference in empirical sampling distributions is likely due to the difference between the greedy policy of UCB1-FLAD and the stochastic policy of EXP3-FLAD. Empirically, we find that EXP3-FLAD very rarely assigns an auxiliary dataset a probability $< 1\%$, leading to many "bad" batches over the course of thousands of turns. On the other hand, the optimistic policy of UCB1-FLAD spends much less time exploring and will sample "bad" batches much less frequently.

Empirical Findings on Gradient Alignment as a Reward We find gradient alignment to be a very noisy signal (see Appendix C). For EXP3-FLAD, this may harm it's policy because the adversary is frequently changing which auxiliary dataset is best, forcing it to make large changes to the sampling distribution. On the other hand, UCB1-FLAD smooths the reward signal with an exponential moving average, leading to the model sampling multiple batches from an auxiliary dataset before adjusting the confidence bounds. By sampling multiple batches, UCB1-FLAD may better approximate the auxiliary dataset reward.

Furthermore, we find that the rewards are mostly stationary (although highly stochastic) after an initial break-in period of $\sim$100 gradient updates, tending towards 0.

FLAD Provides Robustness Compared with Non-FLAD Methods In this section, we compare the performance of our methods trained on P3 with competitive few-shot methods: T-Few, DEFT-Few, and GPT-3. T-Few (Liu et al., 2022) is a variant of the T0-3B model that multi-task pretrains parameter-efficient $(IA)^3$ modules followed by target-only fine-tuning the $(IA)^3$ modules. DEFT-Few (Ivison et al., 2022) is a variant of the T5-XL model that has been multi-task trained using retrieved auxiliary data. Using

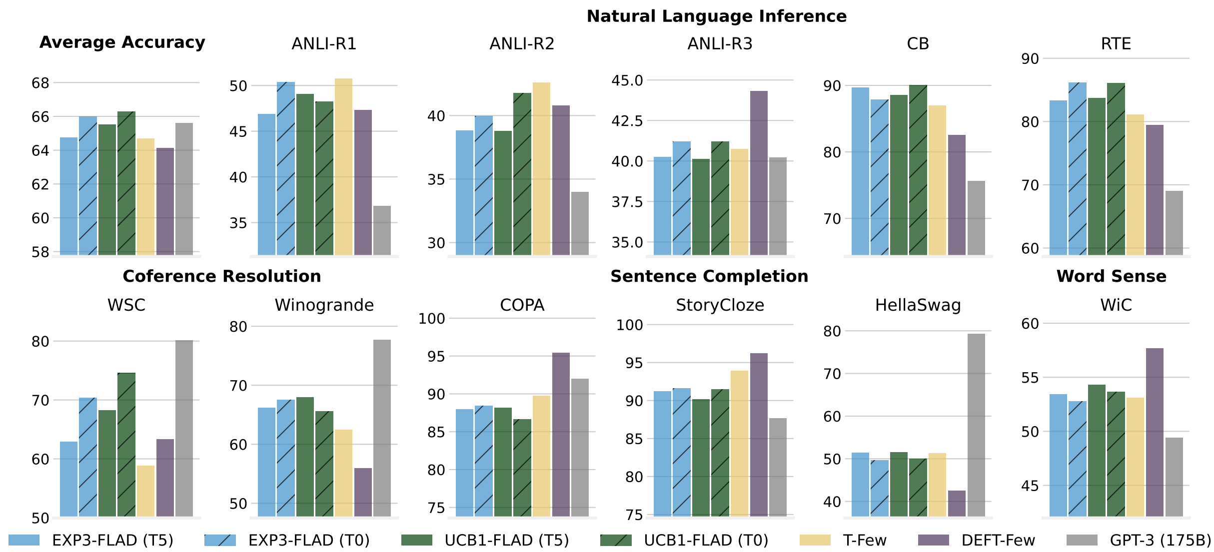

Figure 3. **Comparison of FLAD methods trained on P3 with previous few-shot methods**. We calculate T-Few scores on our data splits using code from Liu et al. (2022). DEFT-Few scores are from Ivison et al. (2022). GPT-3 scores are from Brown et al. (2020) and utilize few-shot in-context learning. All models utilize the same number of few-shot examples and (other than GPT-3) have 3B parameters.

1000 unlabeled target dataset samples, DEFT first multi-task trains a T5-XL model on the 500 nearest neighbor samples from P3. DEFT then involves target-only fine-tuning with the $(IA)^3$ modules from Liu et al. (2022). Finally, we also compare against the 175 billion parameter variant of GPT-3 (Brown et al., 2020), which utilizes in-context learning.

We find that, on average, T0 models trained using our FLAD-based methods outperform all other methods and, to the best of our knowledge, our methods lead to the first 3 billion parameter model that outperforms GPT-3 on this dataset mixture (previous smallest models have 11 billion parameters), with 62.5 times fewer parameters. Additionally, we find that our FLAD-based methods provide robust performance across datasets, achieving best or second-best performance on $9/11$ datasets, only performing worst on COPA.

7. Related Work

A similar line of work to ours is data selection, where specific datapoints are selected based on criteria relative to the model being trained (Mindermann et al., 2021; Siddiqui et al., 2022; Sorscher et al., 2022). Yet other areas which have a similar goal to FLAD are transfer-learning (Pruksachatkun et al., 2020), meta-learning (Thrun & Pratt, 1998), and multi-task learning (Sanh et al., 2022).

Prior works have utilized MAB to improve supervised learning. For example, Pasunuru et al. (2020) use MAB to optimize multiple rewards for language generation, Guo et al. (2018) use it to select auxiliary tasks to improve sentence simplification, and Graves et al. (2017) use it to automate curriculum learning. Even more methods have utilized MAB in reinforcement learning, such as (Sharma & Ravindran, 2017; Lin et al., 2019).

Some early works found success when combining auxiliary data into target-aware training, such as Wu & Dietterich (2004). Dery et al. (2021) argue for target-aware training as an alternative to pre-training and fine-tuning. Chen et al. (2022) perform target-aware training, but assume access to 10,000 target samples. Other works (Lin et al., 2022; Ivison et al., 2022) incorporate auxiliary data by retrieving samples, but assume access to 1,000 unlabeled target examples to create a quality representation for retrieval. More similar to our work are Du et al. (2018); Verboven et al. (2022), which adaptively weight auxiliary losses based on gradient alignment, but evaluate with only 2 auxiliary datasets.

8. Conclusion

The methods proposed in this work demonstrate the effectiveness of: simultaneous training on auxiliary and target datasets in few-shot settings, continuously updating beliefs by exploring *and* exploiting auxiliary data, and framing FLAD as an MAB problem. In this work, we show that FLAD is not a strong adversary, that gradient alignments maintain a weak form of stationarity, and although gradient alignments tend towards 0 there is useful information in the noisy signal that models can utilize. While the presented algorithms satisfy our desiderata, the findings from this study can inform future work to further improve upon these methods in a number of ways, such as improving the reward function and reducing the space complexity.

References

Aghajanyan, A., Gupta, A., Shrivastava, A., Chen, X., Zettlemoyer, L., and Gupta, S. Muppet: Massive multi-task representations with pre-finetuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5799–5811, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.468. URL `https://aclanthology.org/2021.emnlp-main.468`.

Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., Ring, R., Rutherford, E., Cabi, S., Han, T., Gong, Z., Samangooei, S., Monteiro, M., Menick, J., Borgeaud, S., Brock, A., Nematzadeh, A., Sharifzadeh, S., Binkowski, M., Barreira, R., Vinyals, O., Zisserman, A., and Simonyan, K. Flamingo: a visual language model for few-shot learning. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=EbMuimAbPbs`.

Albalak, A., Tuan, Y.-L., Jandaghi, P., Pryor, C., Yoffe, L., Ramachandran, D., Getoor, L., Pujara, J., and Wang, W. Y. Feta: A benchmark for few-sample task transfer in open-domain dialogue, 2022. URL `https://arxiv.org/abs/2205.06262`.

Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. E. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Proceedings of IEEE 36th annual foundations of computer science*, pp. 322–331. IEEE, 1995.

Auer, P., Cesa-Bianchi, N., and Fischer, P. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256, 2002a.

Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. E. The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, 32(1):48–77, 2002b.

Bach, S., Sanh, V., Yong, Z. X., Webson, A., Raffel, C., Nayak, N. V., Sharma, A., Kim, T., Bari, M. S., Fevry, T., Alyafeai, Z., Dey, M., Santilli, A., Sun, Z., Ben-david, S., Xu, C., Chhablani, G., Wang, H., Fries, J., Al-shaibani, M., Sharma, S., Thakker, U., Almubarak, K., Tang, X., Radev, D., Jiang, M. T.-j., and Rush, A. 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*, pp. 93–104, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-demo.9. URL `https://aclanthology.org/2022.acl-demo.9`.

Bansal, T., Jha, R., Munkhdalai, T., and McCallum, A. Self-supervised meta-learning for few-shot natural language classification tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 522–534, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.38. URL `https://aclanthology.org/2020.emnlp-main.38`.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf`.

Chen, S., Crammer, K., He, H., Roth, D., and Su, W. J. Weighted training for cross-task learning. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=ltM1RMZntpu`.

Dagan, I., Glickman, O., and Magnini, B. The pascal recognising textual entailment challenge. In Quiñonero-Candela, J., Dagan, I., Magnini, B., and d'Alché Buc, F. (eds.), *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pp. 177–190, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-33428-6.

de Marneffe, M.-C., Simons, M., and Tonhauser, J. The commitmentbank: Investigating projection in naturally occurring discourse. In *Sinn und Bedeutung*, 2019.

Dery, L. M., Michel, P., Talwalkar, A. S., and Neubig, G. Should we be pre-training? an argument for end-task aware training as an alternative. *ArXiv*, abs/2109.07437, 2021.

Dery, L. M., Michel, P., Khodak, M., Neubig, G., and Talwalkar, A. Aang: Automating auxiliary learning, 2022. URL `https://arxiv.org/abs/2205.14082`.

Du, Y., Czarnecki, W. M., Jayakumar, S. M., Farajtabar, M., Pascanu, R., and Lakshminarayanan, B. Adapting auxiliary losses using gradient similarity, 2018. URL `https://arxiv.org/abs/1812.02224`.

Esfandiarpoor, R., Pu, A., Hajabdollahi, M., and Bach, S. H. Extended few-shot learning: Exploiting existing resources for novel tasks, 2020. URL https://arxiv.org/abs/2012.07176.

Gordon, A., Kozareva, Z., and Roemmele, M. SemEval-2012 task 7: Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pp. 394–398, Montréal, Canada, 7-8 June 2012. Association for Computational Linguistics. URL https://aclanthology.org/S12-1052.

Graves, A., Bellemare, M. G., Menick, J., Munos, R., and Kavukcuoglu, K. Automated curriculum learning for neural networks. In *international conference on machine learning*, pp. 1311–1320. PMLR, 2017.

Guo, H., Pasunuru, R., and Bansal, M. Dynamic multi-level multi-task learning for sentence simplification. In *International Conference on Computational Linguistics*, 2018.

Ivison, H., Smith, N. A., Hajishirzi, H., and Dasigi, P. Data-efficient finetuning using cross-task nearest neighbors, 2022. URL https://arxiv.org/abs/2212.00196.

Lattimore, T. and Szepesvári, C. *Bandit Algorithms*. Cambridge University Press, 2020. doi: 10.1017/9781108571401.

LeCun, Y. A., Bottou, L., Orr, G. B., and Müller, K.-R. *Efficient BackProp*, pp. 9–48. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-35289-8. doi: 10.1007/978-3-642-35289-8_3. URL https://doi.org/10.1007/978-3-642-35289-8_3.

Lester, B., Al-Rfou, R., and Constant, N. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.243. URL https://aclanthology.org/2021.emnlp-main.243.

Levesque, H., Davis, E., and Morgenstern, L. The winograd schema challenge. In *Thirteenth international conference on the principles of knowledge representation and reasoning*, 2012.

Lin, B. Y., Tan, K., Miller, C., Tian, B., and Ren, X. Unsupervised cross-task generalization via retrieval augmentation. *ArXiv*, abs/2204.07937, 2022.

Lin, X., Baweja, H., Kantor, G., and Held, D. Adaptive auxiliary task weighting for reinforcement learning. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/0e900ad84f63618452210ab8baae0218-Paper.pdf.

Liu, H., Tam, D., Muqeeth, M., Mohta, J., Huang, T., Bansal, M., and Raffel, C. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *arXiv preprint arXiv:2205.05638*, 2022.

Macready, W. G. and Wolpert, D. H. Bandit problems and the exploration/exploitation tradeoff. *IEEE Transactions on evolutionary computation*, 2(1):2–22, 1998.

McCann, B., Keskar, N. S., Xiong, C., and Socher, R. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*, 2018.

Mindermann, S., Razzak, M., Xu, W., Kirsch, A., Sharma, M., Morisot, A., Gomez, A. N., Farquhar, S., Brauner, J., and Gal, Y. Prioritized training on points that are learnable, worth learning, and not yet learned. In *International Conference on Machine Learning*, 2021.

Nie, Y., Williams, A., Dinan, E., Bansal, M., Weston, J., and Kiela, D. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4885–4901, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.441. URL https://aclanthology.org/2020.acl-main.441.

Parnami, A. and Lee, M. Learning from few examples: A summary of approaches to few-shot learning, 2022. URL https://arxiv.org/abs/2203.04291.

Pasunuru, R., Guo, H., and Bansal, M. Dorb: Dynamically optimizing multiple rewards with bandits. In *EMNLP*, 2020.

Phang, J., Févry, T., and Bowman, S. R. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*, 2018.

Pilehvar, M. T. and Camacho-Collados, J. WiC: the word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* *Volume 1 (Long and Short Papers)*, pp. 1267–1273, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1128. URL `https://aclanthology.org/N19-1128`.

Pruksachatkun, Y., Phang, J., Liu, H., Htut, P. M., Zhang, X., Pang, R. Y., Vania, C., Kann, K., and Bowman, S. R. Intermediate-task transfer learning with pretrained language models: When and why does it work? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 5231–5247, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.467. URL `https://aclanthology.org/2020.acl-main.467`.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2019.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1), jun 2020. ISSN 1532-4435.

Ravi, S. and Larochelle, H. Optimization as a model for few-shot learning. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=rJY0-Kcll`.

Sakaguchi, K., Bras, R., Bhagavatula, C., and Yejin, C. Winogrande: An adversarial winograd schema challenge at scale. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:8732–8740, 04 2020. doi: 10.1609/aaai.v34i05.6399.

Sanh, V., Webson, A., Raffel, C., Bach, S., Sutawika, L., Alyafeai, Z., Chaffin, A., Stiegler, A., Raja, A., Dey, M., Bari, M. S., Xu, C., Thakker, U., Sharma, S. S., Szczechla, E., Kim, T., Chhablani, G., Nayak, N., Datta, D., Chang, J., Jiang, M. T.-J., Wang, H., Manica, M., Shen, S., Yong, Z. X., Pandey, H., Bawden, R., Wang, T., Neeraj, T., Rozen, J., Sharma, A., Santilli, A., Fevry, T., Fries, J. A., Teehan, R., Scao, T. L., Biderman, S., Gao, L., Wolf, T., and Rush, A. M. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=9Vrb9D0WI4`.

Seldin, Y., Szepesvári, C., Auer, P., and Abbasi-Yadkori, Y. Evaluation and analysis of the performance of the exp3 algorithm in stochastic environments. In Deisenroth, M. P., Szepesvári, C., and Peters, J. (eds.), *Proceedings of the Tenth European Workshop on Reinforcement Learning*, volume 24 of *Proceedings of Machine Learning Research*, pp. 103–116, Edinburgh, Scotland, 30 Jun–01 Jul 2013. PMLR. URL `https://proceedings.mlr.press/v24/seldin12a.html`.

Sharma, R., Allen, J., Bakhshandeh, O., and Mostafazadeh, N. Tackling the story ending biases in the story cloze test. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 752–757, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2119. URL `https://aclanthology.org/P18-2119`.

Sharma, S. and Ravindran, B. Online multi-task learning using active sampling. *ArXiv*, abs/1702.06053, 2017.

Shazeer, N. and Stern, M. Adafactor: Adaptive learning rates with sublinear memory cost. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4596–4604. PMLR, 10–15 Jul 2018. URL `https://proceedings.mlr.press/v80/shazeer18a.html`.

Siddiqui, S. A., Rajkumar, N., Maharaj, T., Krueger, D., and Hooker, S. Metadata archaeology: Unearthing data subsets by leveraging training dynamics. *ArXiv*, abs/2209.10015, 2022.

Simpkins, A., De Callafon, R., and Todorov, E. Optimal trade-off between exploration and exploitation. In *2008 American Control Conference*, pp. 33–38. IEEE, 2008.

Sorscher, B., Geirhos, R., Shekhar, S., Ganguli, S., and Morcos, A. S. Beyond neural scaling laws: beating power law scaling via data pruning. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=UmvSlP-PyV`.

Thrun, S. and Pratt, L. (eds.). *Learning to Learn*. Kluwer Academic Publishers, USA, 1998. ISBN 0792380479.

Verboven, S., Chaudhary, M. H., Berrevoets, J., Ginis, V., and Verbeke, W. Hydalearn. *Applied Intelligence*, Jul 2022. ISSN 1573-7497. doi: 10.1007/s10489-022-03695-x. URL `https://doi.org/10.1007/s10489-022-03695-x`.

Vu, T., Wang, T., Munkhdalai, T., Sordoni, A., Trischler, A., Mattarella-Micke, A., Maji, S., and Iyyer, M. Exploring and predicting transferability across NLP tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7882–7926, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.635. URL `https://aclanthology.org/2020.emnlp-main.635`.

Wang, P., Yang, A., Men, R., Lin, J., Bai, S., Li, Z., Ma, J., Zhou, C., Zhou, J., and Yang, H. Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. *CoRR*, abs/2202.03052, 2022.

Wang, Y., Yao, Q., Kwok, J. T., and Ni, L. M. Generalizing from a few examples: A survey on few-shot learning. *ACM Comput. Surv.*, 53(3), jun 2020. ISSN 0360-0300. doi: 10.1145/3386252. URL https://doi.org/10.1145/3386252.

Wang, Z., Tsvetkov, Y., Firat, O., and Cao, Y. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=F1vEjWK-lH_.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6. URL https://aclanthology.org/2020.emnlp-demos.6.

Wu, P. and Dietterich, T. G. Improving svm accuracy by training on auxiliary data sources. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04, pp. 110, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 1581138385. doi: 10.1145/1015330.1015436. URL https://doi.org/10.1145/1015330.1015436.

Wu, S., Zhang, H. R., and Ré, C. Understanding and improving information transfer in multi-task learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SylzhkBtDB.

Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., and Finn, C. Gradient surgery for multi-task learning. *arXiv preprint arXiv:2001.06782*, 2020.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472.

A. Training Details

We train all models (FLAD and non-FLAD) on 40Gb A100s.

For all experiments, we use validation-based early stopping, and train for a maximum of 10,000 gradient update steps. In practice, we find that early-stopping leads to significantly fewer than 10,000 updates, usually between 50-150 for direct fine-tuning, and 1-2,000 for other methods.

For the smoothing factor, β, in UCB1-FLAD we ran preliminary experiments using values of $\{0.99, 0.9, 0.75, 0.5\}$ and found 0.9 to work well across datasets. All reported scores use $\beta = 0.9$.

In preliminary experiments we consider rewards using gradients from multiple model partitions: the full model, encoder-only, decoder-only, and language modelling head (token classifier). We find that using the parameters from the LM head provides best performance, followed by the decoder-only, encoder-only, and full model gradients. The differential from best to worst method was $\sim 3\%$ relative performance. Recall that with a gradient accumulation factor of G, our algorithms need to store at most $G + 1$ gradients at any time. So not only does using the LM head provide performance improvements, but also saves memory. For the models we use, the LM head contains only 2.3% of the full model parameters.

B. Full Results

The full results of experiments on target-only fine-tuning, explore-only, exploit-only, EXP3-FLAD, and UCB1-FLAD are found on the next page.

Table 2. Detailed results from the main experiment including direct fine-tuning, exploration-only, exploitation-only baselines and our proposed methods, EXP3-FLAD and UCB1-FLAD.

		Dataset	Anli-r1	Anli-r2	Anli-r3	CB	COPA	HellaSwag	RTE	Story Cloze	WiC	Winogrande	WSC	Average
T5-3B	T0Mix	Direct Fine-Tuning	37.6	36.2	35.0	83.2	53.8	51.0	54.2	75.9	51.6	49.6	53.1	52.8
		Exploration-Only	38.1	40.3	36.7	88.6	85.6	51.2	67.6	88.8	51.0	55.5	47.7	59.2
		Exploitation-Only	38.8	40.5	38.0	86.1	86.0	51.1	69.4	89.5	52.8	59.2	46.3	59.8
		Exp3-FLAD	40.6	39.9	36.9	86.1	**89.8**	**52.0**	76.7	90.8	50.5	60.3	52.9	61.5
		UCB1-FLAD	41.8	39.0	38.0	85.4	87.0	**52.0**	79.1	91.4	49.7	62.7	56.2	62.0
	P3	Exploration-Only	40.1	37.7	36.0	85.4	83.6	**52.1**	77.3	89.1	51.5	57.2	57.1	60.6
		Exploitation-Only	40.4	37.2	37.3	87.1	84.4	51.0	78.6	90.3	51.3	56.2	51.5	60.5
		Exp3-FLAD	46.9	38.8	40.2	89.6	88.0	51.5	76.9	91.2	53.4	66.2	61.9	64.1
		UCB1-FLAD	49.1	38.8	40.1	88.6	88.2	51.6	83.7	90.2	**54.3**	**68.0**	68.3	65.5
T0-3B	T0Mix	Direct Fine-Tuning	40.9	39.1	37.1	79.6	66.4	43.5	67.1	83.2	52.5	54.6	56.7	56.4
		Exploration-Only	44.4	40.3	37.0	82.5	85.6	47.9	77.6	90.1	52.1	58.6	56.9	61.2
		Exploitation-Only	42.5	39.3	37.2	84.3	82.8	48.1	79.7	88.8	52.8	57.8	56.3	60.9
		Exp3-FLAD	46.2	41.5	37.7	83.9	87.6	49.4	80.0	90.1	52.6	63.4	59.0	62.9
		UCB1-FLAD	43.7	40.8	37.6	86.1	85.4	48.6	80.5	91.3	53.4	63.5	61.0	62.9
	P3	Exploration-Only	45.4	40.3	38.0	82.5	87.8	50.6	82.2	88.8	52.4	61.8	60.6	62.8
		Exploitation-Only	45.5	40.0	38.8	87.5	82.2	49.9	79.6	90.9	52.2	60.1	64.8	62.9
		Exp3-FLAD	**50.4**	40.0	**41.2**	87.9	88.4	49.7	**86.1**	**91.6**	52.8	67.5	70.4	66.0
		UCB1-FLAD	48.2	**41.8**	**41.2**	**90.0**	86.6	50.0	**86.1**	**91.5**	53.6	65.6	**74.6**	66.3

C. EXP3-FLAD and UCB1-FLAD Training Dynamics

The following 4 pages include a case study on the training dynamics of EXP3-FLAD and UCB1-FLAD when training T5-XL using T0Mix as the auxiliary data. First, we find datasets where EXP3-FLAD and UCB1-FLAD improve significantly over the baseline FLAD methods, but also where either EXP3-FLAD or UCB1-FLAD clearly outperforms the other. The two datasets that fulfill our interests are RTE and COPA.

We find that UCB1-FLAD outperforms EXP3-FLAD on RTE, and show their respective training dynamics in Figure 4 (UCB1) and Figure 5 (EXP3).

We find that EXP3-FLAD outperforms UCB1-FLAD on COPA, and show their respective training dynamics in Figure 6 (UCB1) and Figure 7 (EXP3).

We include details and takeaways in the caption for each figure. For EXP3-FLAD figures we include charts of the cumulative estimated reward, empirical gradient alignment, instantaneous sampling distribution determined by the policy, and the empirical sampling distribution determined by the total number of samples seen per dataset as a fraction of the total samples seen. For UCB1-FLAD figures, we include charts of the upper confidence index, estimated gradient alignment, and the empirical sampling distribution.

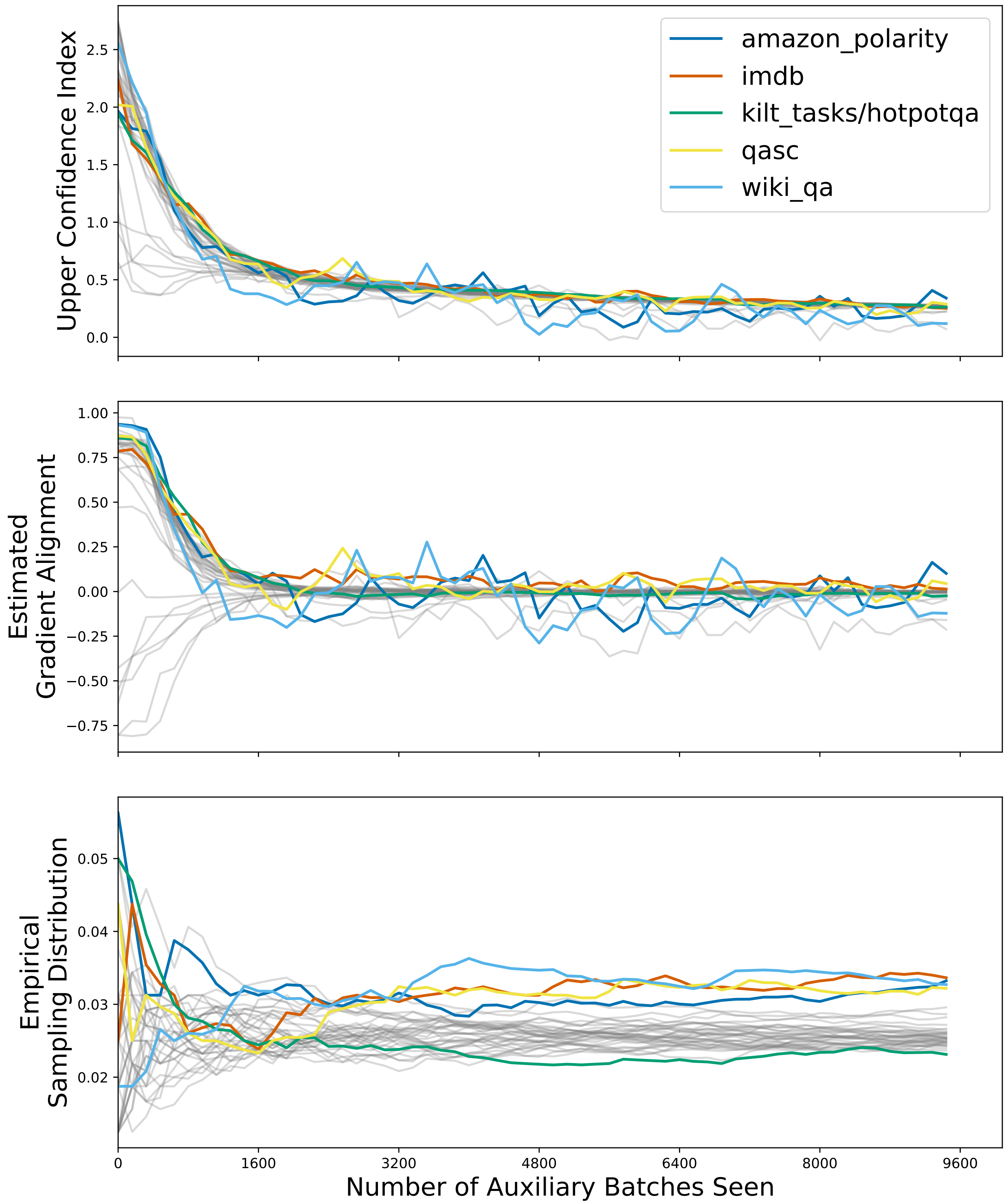

Figure 4. Training dynamics of UCB1-FLAD, a case study using RTE as target dataset and T0Mix as auxiliary data, where UCB1-FLAD outperforms EXP3-FLAD. Colored lines are a sample of auxiliary datasets with interesting properties, the remaining datasets are shown in grey. We find that even though wiki_qa's estimated gradient alignment falls to below 0 (middle), UCB1 does not abandon sampling from it in the future, finding that between 3200 and 4800 batches, it becomes the dataset with largest upper confidence bound (top). Similarly, we see that UCB1 alternates between wiki_qa, amazon_polarity, and qasc as the datasets with higher gradient alignment and upper confidence bounds. kilt_tasks/hotpotqa has a very high gradient alignment prior to training, but UCB1 samples very infrequently from it, due to it'ls lower upper confidence bound. This is a failure case for transfer learning-based methods. Interestingly, UCB1 never estimates imdb to have a negative gradient, and gradually samples from it more and more frequently over the course of training.

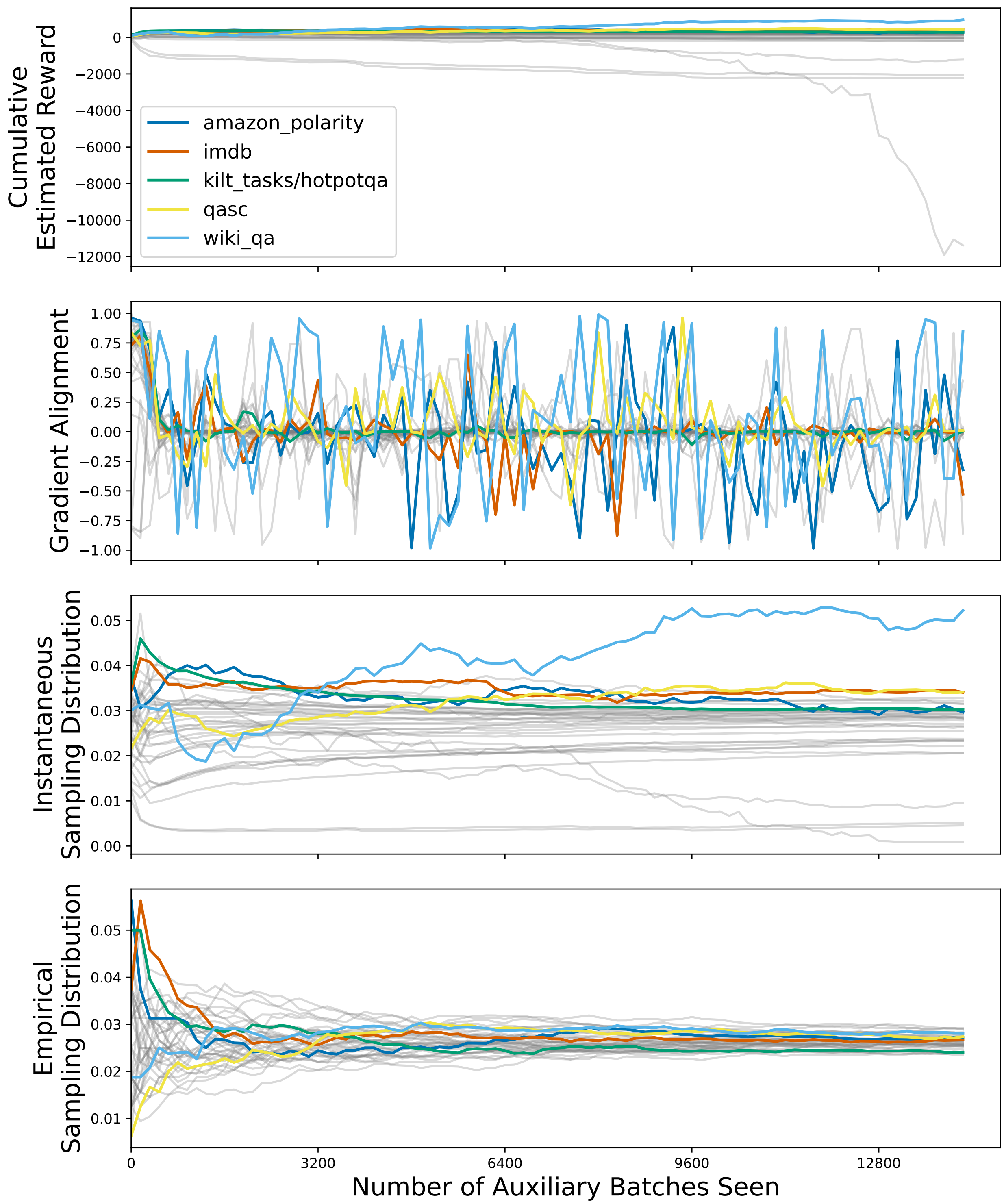

Figure 5. Training dynamics of EXP3-FLAD, a case study using RTE as target dataset and T0Mix as auxiliary data, where UCB1-FLAD outperforms EXP3-FLAD. Colored lines are a sample of auxiliary datasets with interesting properties, the remaining datasets are shown in grey. We find that the gradient alignment signal is particularly noisy for EXP3-FLAD, possibly leading to it's slightly worse performance on RTE. All five highlighted auxiliary datasets have high instantaneous sampling probability, but over the course of training, the empirical sampling distribution is very condensed across the full set of auxiliary datasets, unlike UCB1 which is able to find better separation.

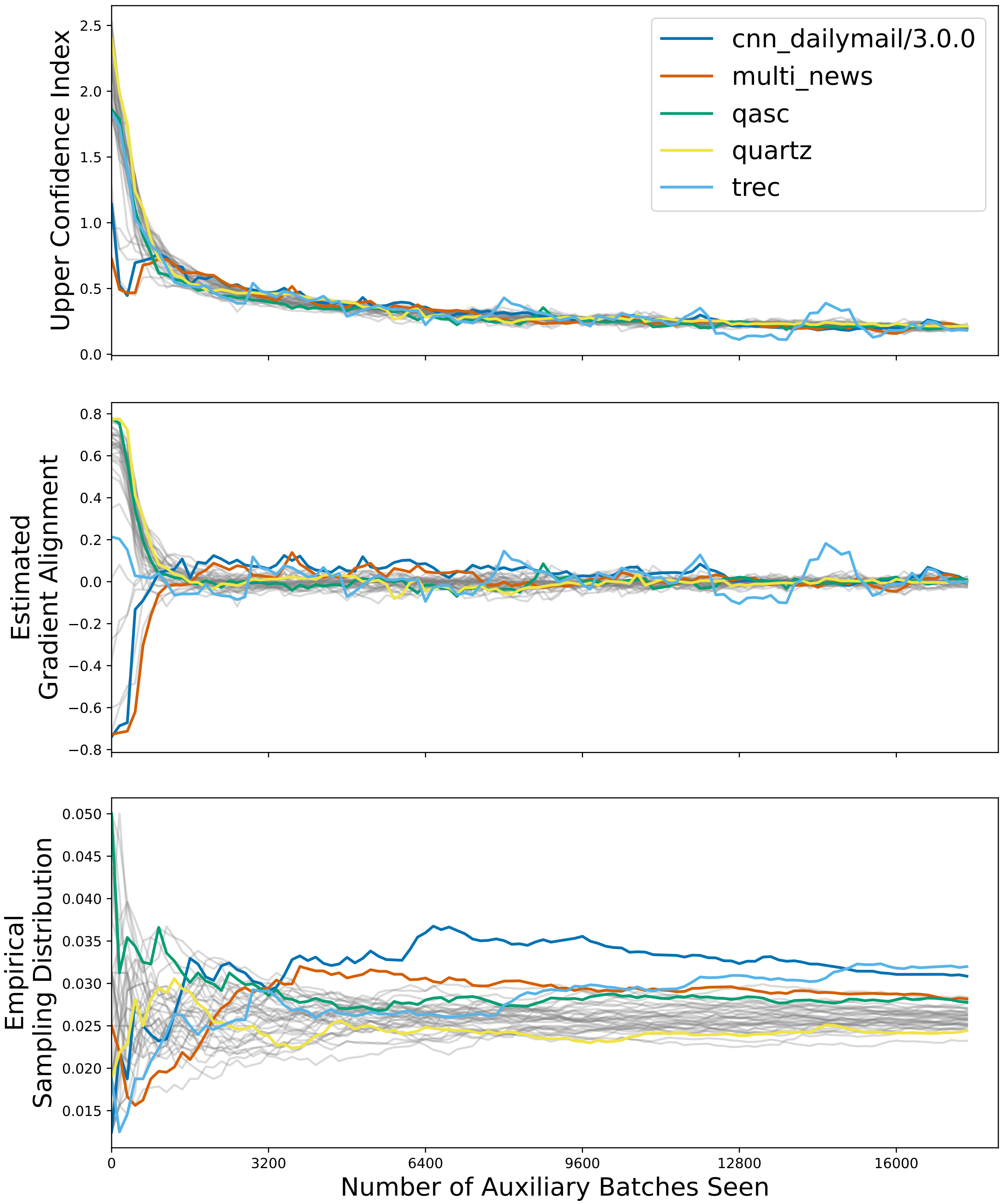

Figure 6. Training dynamics of UCB1-FLAD, a case study using COPA as target dataset and T0Mix as auxiliary data, where EXP3-FLAD outperforms UCB1-FLAD. Colored lines are a sample of auxiliary datasets with interesting properties, the remaining datasets are shown in grey. We find that although qasc and quartz start with very high gradient alignment, they very quickly fall to negative alignment (middle figure, green and yellow). In the end, we find that the algorithm samples much more from qasc than from quartz (bottom figure). Interestingly, we find that although both cnn_dailymail and multi_news start off with very negative gradient alignment, they quickly become the most aligned with the target task (middle figure, blue and red). We find that the three auxiliary datasets with highest upper confidence index (top figure) and largest sampling percent (bottom figure) are cnn_dailymail, multi_news, and trec even though these all considered dissimilar to the target prior to training.

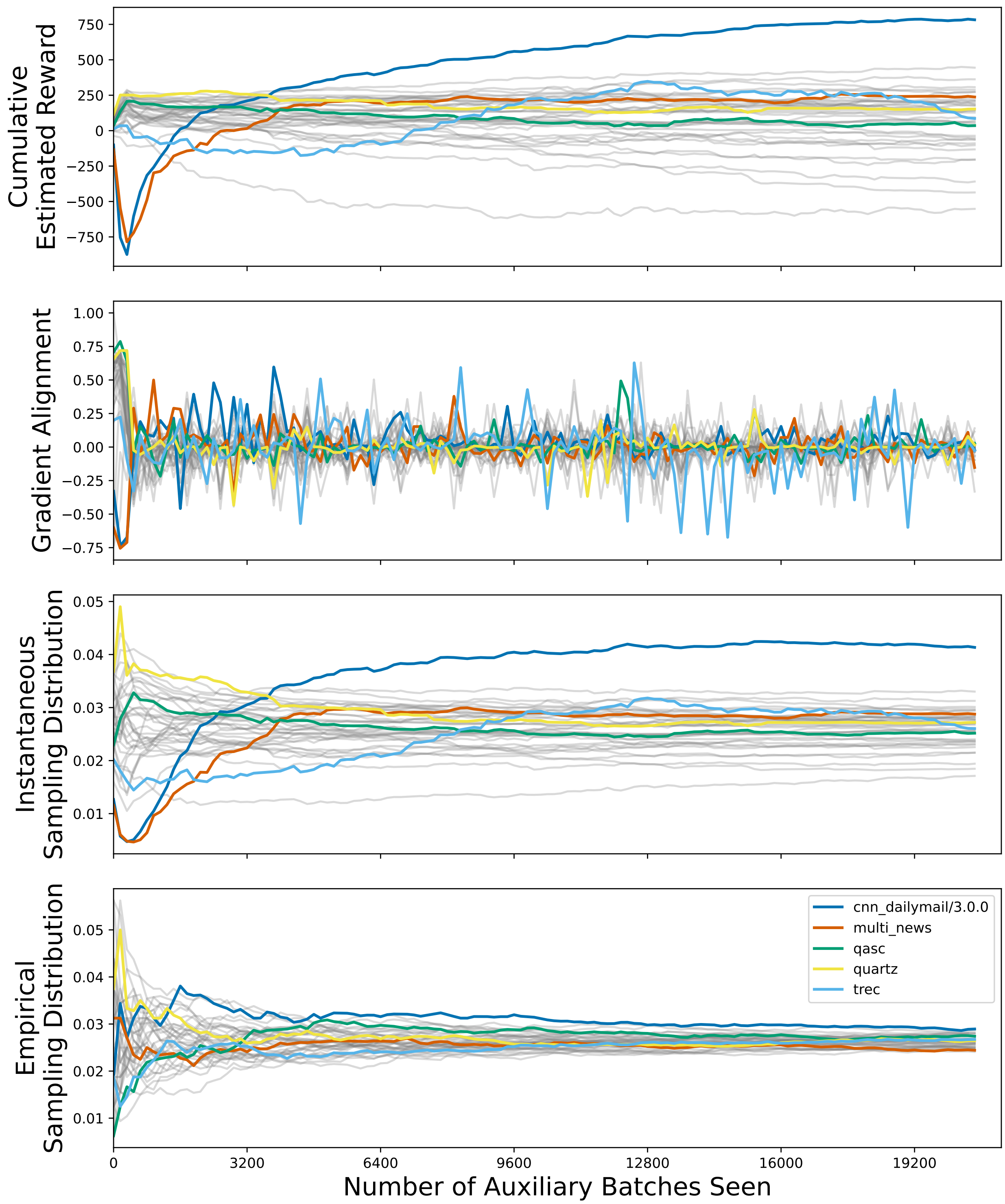

Figure 7. Training dynamics of EXP3-FLAD, a case study using COPA as target dataset and T0Mix as auxiliary data, where EXP3-FLAD outperforms UCB1-FLAD. Colored lines are a sample of auxiliary datasets with interesting properties, the remaining datasets are shown in grey. This is an impressive example of the importance-weighted estimated reward. We see that cnn_dailymail and multi_news both start with very negative alignment, but EXP3 quickly updates it's estimated reward once their alignment becomes positive. Similar to RTE, we see that EXP3 never makes large separations in the empirical sampling distribution, possibly a reason why UCB1 outperforms EXP3 overall. Compared to RTE, we find that gradient alignments are much less variable, with a maximum alignment close to 0.5 and minimum alignment close to -0.5. Whereas in RTE, alignments regularly reach close to 1.0 and -1.0.