Learning to Obstruct Few-Shot Image Classification over Restricted Classes

Amber Yijia Zheng*, Chiao-An Yang*, and Raymond A. Yeh

Department of Computer Science, Purdue University
{zheng709,yang2300,rayyeh}@purdue.edu

Abstract. Advancements in open-source pre-trained backbones make it relatively easy to fine-tune a model for new tasks. However, this lowered entry barrier poses potential risks, e.g., bad actors developing models for harmful applications. A question arises: *Is possible to develop a pre-trained model that is difficult to fine-tune for certain downstream tasks?* To begin studying this, we focus on few-shot classification (FSC). Specifically, we investigate methods to make FSC more challenging for a set of restricted classes while maintaining the performance of other classes. We propose to meta-learn over the pre-trained backbone in a manner that renders it a "poor initialization". Our proposed Learning to Obstruct (LTO) algorithm successfully obstructs four FSC methods across three datasets, including ImageNet and CIFAR100 for image classification, as well as CelebA for attribute classification.

1 Introduction

Open-sourced and pre-trained models have helped to make tremendous progress in computer vision and machine learning research [6]. These open-source models improve the reproducibility of research and allow for fair comparisons across the models [48]. With the accessible pre-trained model such as image classifiers [19, 21, 42] trained on ImageNet [7], much research spurred out of these backbones building on top of them, e.g., detection [30, 41], segmentation [18, 32], and many other applications based on transfer learning [22, 26]. However, as computer vision and fine-tuning methods improve, open-sourced model weights may become a double-edged sword. With the ability to quickly fine-tune a model to a new task with few training samples, the entry barrier to developing a working computer vision system on a new task is greatly lowered; this includes bad actors developing potentially harmful applications, e.g., the ability to quickly train models on human subjects which may raise privacy concerns [35, 61]. In this work, we ask the following question:

Is possible to develop a pre-trained model that is difficult to fine-tune for certain downstream tasks?

If we succeed, the pre-trained models can be released to support scientific research while addressing safety concerns. To begin this endeavor, we focus on the task

*indicates equal contribution

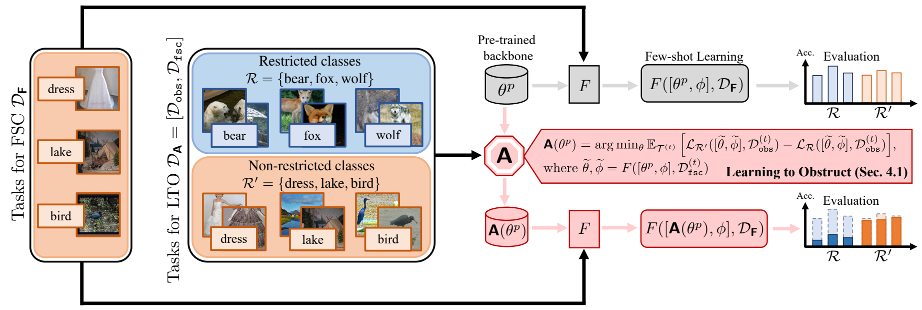
Tasks for FSC $\mathcal{D}_{\mathsf{F}}$
dress
lake
bird
Tasks for LTO $\mathcal{D}_{\mathsf{A}} = [\mathcal{D}_{\mathtt{obs}}, \mathcal{D}_{\mathtt{fsc}}]$
Restricted classes
$\mathcal{R}$ = {bear, fox, wolf}
bear
fox
wolf
Non-restricted classes
$\mathcal{R}'$ = {dress, lake, bird}
dress
lake
bird
Pre-trained backbone
θ^p
F
Few-shot Learning
$F([\theta^p, \phi], \mathcal{D}_{\mathsf{F}})$
Acc.
Evaluation
$\mathcal{R}$
$\mathcal{R}'$
A
$\mathbf{A}(\theta^p) = \arg\min_\theta \mathbb{E}_{\mathcal{T}^{(t)}} \left[\mathcal{L}_{\mathcal{R}'}([\widetilde{\theta}, \widetilde{\phi}], \mathcal{D}^{(t)}_{\mathtt{obs}}) - \mathcal{L}_{\mathcal{R}}([\widetilde{\theta}, \widetilde{\phi}], \mathcal{D}^{(t)}_{\mathtt{obs}})\right]$,
where $\widetilde{\theta}, \widetilde{\phi} = F([\theta^p, \phi], \mathcal{D}^{(t)}_{\mathtt{fsc}})$
Learning to Obstruct (Sec. 4.1)
$\mathbf{A}(\theta^p)$
F
$F([\mathbf{A}(\theta^p), \phi], \mathcal{D}_{\mathsf{F}})$
Acc.
Evaluation
$\mathcal{R}$
$\mathcal{R}'$

Fig. 1: Learning to Obstruct (LTO) few-shot learning paradigm. *Without LTO*: after the adaptation of few-shot learner F, the model can classify classes from $\mathcal{R}$ and $\mathcal{R}'$ correctly. *With LTO*: By modifying the pre-trained model parameters θ^p via our proposed method $\mathbf{A}$ before the adaptation of F, the model fails to generalize to restricted class set $\mathcal{R}$ while maintaining its performance in other class set $\mathcal{R}'$.

of few-shot classification (FSC). We investigate whether it is possible to have a pre-trained model that FSC performs poorly on a set of restricted classes while remaining competitive on the remaining classes. We selected FSC to study as it is a well-established area with proper benchmark and fine-tuning procedures [9, 14, 20, 46, 47, 60, 63] using pre-trained ImageNet backbones, e.g., ResNet [19], and more recently on large-scale language and vision backbone, e.g., CLIP [38].

To achieve this goal, we propose Learning To Obstruct (LTO), a MAML [10]-like algorithm, that learns a *poor initilization* w.r.t. an FSC algorithm for the set of restricted classes. We evaluate the proposed LTO algorithm by conducting experiments on two few-shot classification setups: (a) the classic N-way-K-shot setting using ProtoNet [46] and MetaOptNet [28]; (b) the more recent language-vision few-shot learning setting using CoOp [63] and Tip-Adapter [60]. On ImageNet [7] and CIFAR100 [27], we show that LTO successfully obstructs the learning of FSC methods, achieving lowered accuracy on restricted classes and maintained competition accuracy on other classes. Lastly, we also experimented with applying LTO for attribute learning on the CelebA dataset [31].

Our contributions are as follows:

- We propose the task of learning to obstruct FSC from learning in restricted classes.
- We present LTO, a meta-learning algorithm, that learns poor backbone initialization for obstructing FSC methods.
- We conduct extensive experiments validating the effectiveness of LTO on four different FSC algorithms on ImageNet, CIFAR100, and CelebA datasets.

2 Related Work

Few-shot learning. Learning from limited labeled data is a crucial task in computer vision, which is known as few-shot learning (FSL) [9]. The mainstream of FSL can be categorized into three main branches: metric-based [16, 34, 37, 46, 52, 59], optimization-based [10, 12, 23, 39, 40], and augmentation-based methods [4, 17, 43, 54]. In this work, we focus on few-shot image classification [2, 3, 24, 49, 55, 58], i.e., FSL but for image classification.

With the availability of pre-trained backbones, FSC methods now use pre-trained weights for initialization [36]. The choice of pre-trained weights varies between the methods. For example, a common choice is pre-training on large-scale image classification tasks [7, 19]. More recently, the use of large-scale language and vision backbones have also been explored. For example, CLIP [38] and DeCLIP [29] have demonstrated remarkable capabilities in learning zero-shot transferable features across diverse datasets and domains. Building on top of these pre-trained backbones, CoOp [63], CLIP-Adapter [14], and Tip-Adapter [60] show that the model performance can be further improved by prompt optimization [63], fine-tuning the introduced light-layer "Adapter" [14] or even training-free adaptation [60]. Without any restriction on the learnable classes, one can easily apply these few-shot methods on pre-trained backbones, even for classes with potential harm.

Machine unlearning. In machine unlearning, a model is trained to forget specific class(es) of data while retaining the memory of the rest without re-training from scratch. This concept is encapsulated in a data-forgetting algorithm proposed by Cao and Yang [1]. For example, Golatkar et al. [15] trained two separate networks: the core model and a mixed-linear model for unlearning purposes. Tarun et al. [51] introduced an error-maximization-based method to learn a noise matrix for the class to be forgotten. Recently, with the pre-trained language and vision backbone, machine unlearning is also used for de-biasing by forgetting certain attributes of the images. Wang et al. [53] removed the dimensions of CLIP embeddings that are highly correlated with the target attributes. Foster et al. [11] propose SSD, i.e. selective synaptic dampening, a swift and effective retrain-free method for machine unlearning, utilizing a two-step process to identify and diminish crucial parameters without the need for long-term storage of training data. We note that the proposed task of obstructing FSC is not machine unlearning. In machine unlearning, the goal is the "removal of" certain classes, whereas our goal is the "prevention of learning" certain classes.

Data poisoning attacks. As the proposed LTO can be viewed as "ruining" the backbone for the restricted classes, another potential way to accomplish that is through data poisoning attacks. Specifically, data poisoning attacks make changes to the training data to corrupt models' test-time behavior [5, 25, 44, 45, 56]. In the context of few-shot learning, Oldewage et al. [33] poisoned the support data in meta-testing, and achieved misclassification for the query prediction. However, this poisoning approach falls short of offering genuine protection for specific restricted classes. Individuals could still collect a few clean images for FSC. In contrast to these approaches, our proposed method obstructs the pre-trained backbone against the restricted classes.

3 Preliminaries

As our proposed learning to obstruct can be thought of as learning an initialization for few-shot classification, we provide a brief review of how to learn a good initialization, specifically on MAML [10], followed by the background of few-shot classification [3, 20, 52].

Learning a good initialization. At a high level, MAML is an algorithm that aims to learn a "good initialization" for a model being trained on a new task using gradient-based methods. MAML assumes a distribution of tasks $P(\mathcal{T})$ where each task $\mathcal{T}^{(t)} = (\mathcal{S}^{(t)}, \mathcal{Q}^{(t)}) \sim P(\mathcal{T})$ is comprised of a support set $\mathcal{S}$ and a query set $\mathcal{Q}$.

The task of learning a good initialization is then formulated as the following optimization problem:

$$\vartheta^\star = \arg\min_\vartheta \mathbb{E}_{\mathcal{T}^{(t)} \sim P(\mathcal{T})} \left[\mathcal{L}^M \left(U(\vartheta, \mathcal{S}^{(t)}), \mathcal{Q}^{(t)} \right) \right], \tag{1}$$

where $\mathcal{L}^M$ denotes a loss for the task in MAML, and U denotes a learner function that updates the model parameter $\vartheta \in \mathbb{R}^d$, e.g., MAML chooses U to be a single gradient update. Intuitively, $\mathcal{L}$ is evaluating the quality of ϑ when used as initialization to the learner U. That is, MAML is searching for a good initialization such that after applying the learner U the model performs well.

To solve the optimization problem in Eq. (1), MAML uses gradient descent and approximates the gradient only using first-order terms. In summary, the optimization problem in Eq. (1) can be decomposed into two parts: **(a)** a learner function U that modifies the model parameters, and **(b)** an outer optimization that minimizes $\mathcal{L}$ usually solved using gradient descent for deep nets. This general framework serves as the basis for our proposed LTO algorithm.

Few-shot classification (FSC). The goal of few-shot classification is to train a model such that it generalizes well to novel tasks $\mathcal{T}^{(T+1:T+M)} \sim P(\mathcal{T})$ given a dataset of training tasks $\mathcal{T}^{(1:T)} \sim P(\mathcal{T})$. For few-shot classification, the support set $\mathcal{S} = \{(\boldsymbol{x}_s, \boldsymbol{y}_s)\}$ and query set $\mathcal{Q} = \{(\boldsymbol{x}_q, \boldsymbol{y}_q)\}$ contains input images $\boldsymbol{x}$ with the corresponding groundtruth $\boldsymbol{y}$ in the class space $\mathcal{Y}^{(t)}$ of each task. Typically, the task follows a N-way-K-shot setup, i.e., each support set $\mathcal{S}$ contains N classes and K examples per class.

For most FSC methods [46, 49, 52], a prediction $\hat{\boldsymbol{y}}_q$ given $\boldsymbol{x}_q$ is made using a predictor $\hat{F}$, i.e., $\hat{\boldsymbol{y}}_q = \hat{F}(\boldsymbol{x}_q, \vartheta, \mathcal{S}^{(t)})$, where the prediction depends on the examples in the support set $\mathcal{S}^{(t)}$. Note, $\hat{\boldsymbol{y}}_q$ is a vector of the predicted probability of each class. To train this model, FSC methods train the model using all the training tasks $\mathcal{T}^{(1:T)} = \{\mathcal{T}^{(1)}, \ldots, \mathcal{T}^{(T)}\}$, i.e.,

$$\mathcal{L}^F(\vartheta, \mathcal{T}^{(1:T)}) = \sum_{\mathcal{Q}^{(t)} \in \mathcal{T}^{(1:T)}} \sum_{(\boldsymbol{x}_q, \boldsymbol{y}_q) \in \mathcal{Q}^{(t)}} \ell(\boldsymbol{y}_q, \hat{\boldsymbol{y}}_q) \tag{2}$$

with ℓ denoting the cross-entropy loss on a single sample. Optimizing Eq. (2) gives rise to a learner function

$$F(\vartheta, \mathcal{T}^{(1:T)}) = \arg\min_\vartheta \mathcal{L}^F(\vartheta, \mathcal{T}^{(1:T)}), \tag{3}$$

which outputs the optimal model parameters $\widetilde{\vartheta}$ given the set of training tasks $\mathcal{T}^{(1:T)}$. In summary, a FSC algorithm $\mathbf{F} = (\hat{F}, F)$ is composed of a $\hat{F}$ yielding the prediction for each query sample and F learner function that updates the model parameters towards the solution that minimizes the loss Eq. (2).

For FSC method $\mathbf{F}$ using deep-nets, the model parameter $\vartheta = [\theta, \phi] \in \mathbb{R}^d$ is further decomposed into two parts: θ for the parameters of the backbone g_θ and ϕ for the parameters of the classifier f_ϕ used for constructing the prediction function $\hat{F}$. The choice of g_θ and f_ϕ depends on the FSC method. For example, ProtoNet [46] chooses a classifier f_ϕ to be the normalized distance of an input query's feature to the prototypes, i.e., $\hat{F}(\boldsymbol{x}_q, [\theta, \phi], \mathcal{S})[k] = \frac{\exp(-d(g_\theta(\boldsymbol{x}_q), \boldsymbol{r}_k))}{\sum_{k'} \exp(-d(g_\theta(\boldsymbol{x}_q), \boldsymbol{r}'_k))}$ where d corresponds to a distance function with prototypes $\boldsymbol{r}_k = \frac{1}{\mathcal{S}_k} \sum_{(\boldsymbol{x}_s, \boldsymbol{y}_s) \in \mathcal{S}} g_\theta(\boldsymbol{x}_s)$ defined as the average of samples in the support set with the label k denoted as $\mathcal{S}_k$.

Next, to further improve model performance, instead of training θ from scratch, recent works [2, 3, 20] introduce pre-trained backbones θ^p as the initialization to θ when the training a few-shot classifier.

Language and vision based FSC. CLIP [38] is a powerful foundation model that encodes rich language and vision information. It serves as a strong pre-trained backbone θ^p for few-shot or even zero-shot learning. CLIP consists of a text encoder and a visual encoder, i.e., $\theta^p = [\theta^p_{\texttt{text}}, \theta^p_{\texttt{img}}]$. To build a few-shot classifier [14, 50, 60, 62, 63] over classes $\mathcal{Y}$ with CLIP, the predictor $\hat{F}$ is defined as follows: $\hat{F}(\boldsymbol{x}_q, [\theta, \phi], \mathcal{S})[k] = \frac{\exp(\boldsymbol{v}_k^\intercal \boldsymbol{v}_{\boldsymbol{x}_q})}{\sum_{k'} \exp(\boldsymbol{v}_{k' \in \mathcal{Y}}^\intercal \boldsymbol{v}_{\boldsymbol{x}_q})}$, where $\boldsymbol{v}_{\boldsymbol{x}} = g_{[\theta_{\texttt{img}}, \phi]}(\boldsymbol{x})$ and $\boldsymbol{v}_k = g_{[\theta_{\texttt{text}}, \phi]}(k)$ corresponds to image and class features extracted from the encoders g. We note that there are "implicit" dependencies on the support set $\mathcal{S}$, as the encoders g are fine-tuned on $\mathcal{S}$ in this language and vision-based FSC methods.

4 Approach

In this paper, our goal is to *obstruct* the learning of specific classes in a restricted class set $\mathcal{R}$, when utilizing few-shot classification (FSC) methods. At the same time, we aim to ensure that the model's performance in the other class set $\mathcal{R}'$ remains unaffected. We consider the scenario where the FSC algorithms $\mathbf{F} = (\hat{F}, F)$ using a pre-trained backbone and are known to the obstructor. To achieve this, we introduce the Learning To Obstruct (LTO) algorithm $\mathbf{A}$ that modifies in the pre-trained backbone's parameters θ^p *to create a poor initialization* $\mathbf{A}(\theta^p)$. When the FSC algorithm is applied to $\mathbf{A}(\theta^p)$, the model will perform poorly on restricted classes but not the other classes.

4.1 Learning to obstruct

Problem formulation. As the name suggests, LTO is formulated as a learning problem. In this learning problem, we are given a distribution of tasks $P(\mathcal{T})$ for which we can sample tasks $\mathcal{T}^{(t)}$ containing the support and query set $(\mathcal{S}^{(t)}, \mathcal{Q}^{(t)})$, and a set of restricted classes $\mathcal{R}$. We further define the set of "other classes" as

$\mathcal{R}' = \{k \in \mathcal{Y} : k \notin \mathcal{R}\}$, where $\mathcal{Y} = \bigcup_t \mathcal{Y}^{(t)}$ denotes the set of all possible classes across all tasks.

For our algorithm, we further split the data into two parts,

$$\mathcal{D}^{(t)}_{\texttt{obs}} = (\mathcal{S}^{(t)}_{\texttt{obs}}, \mathcal{Q}^{(t)}_{\texttt{obs}}) \quad \text{and} \quad \mathcal{D}^{(t)}_{\texttt{fsc}} = (\mathcal{S}^{(t)}_{\texttt{fsc}}, \mathcal{Q}^{(t)}_{\texttt{fsc}}). \tag{4}$$

The $\mathcal{D}^{(t)}_{\texttt{obs}}$ split is for evaluating the quality of the obstruction, and $\mathcal{D}^{(t)}_{\texttt{fsc}}$ is to be used for training by the FSC learner function F. Using these splits, we formulate the Learning To Obstruct (LTO) algorithm $\mathbf{A}$ as an optimization problem:

$$\min_\theta \mathbb{E}_{\mathcal{T}^{(t)}} \left[\mathcal{L}_{\mathcal{R}'}\left([\widetilde{\theta}, \widetilde{\phi}], \mathcal{D}^{(t)}_{\texttt{obs}}\right) - \mathcal{L}_{\mathcal{R}}\left([\widetilde{\theta}, \widetilde{\phi}], \mathcal{D}^{(t)}_{\texttt{obs}}\right)\right] \text{ s.t. } \widetilde{\theta}, \widetilde{\phi} = F([\theta, \phi], \mathcal{D}^{(t)}_{\texttt{fsc}}). \tag{5}$$

The objective in Eq. (5) consists of two terms $\mathcal{L}_{\mathcal{R}'}$ and $\mathcal{L}_{\mathcal{R}}$ which corresponds to the few-shot learning loss $\mathcal{L}^F$ in Eq. (2) but evaluated only on *other classes* $\mathcal{R}'$ and restricted classes $\mathcal{R}$ respectively for the query set. Formally,

$$\mathcal{L}_{\mathcal{R}}([\widetilde{\theta}, \widetilde{\phi}], \mathcal{D}^{(t)}_{obs}) = \mathcal{L}^F\left([\widetilde{\theta}, \widetilde{\phi}], (\mathcal{S}^{(t)}_{\texttt{obs}}, \{(\boldsymbol{x}_q, \boldsymbol{y}_q) \in \mathcal{Q}^{(t)} \wedge \boldsymbol{y}_q \in \mathcal{R}\})\right)$$

and vice versa for $\mathcal{R}'$. Intuitively Eq. (5), inspired by MAML, aims to learn a poor initialization of model weights for classes in $\mathcal{R}$, we first let θ explore the landscape using the FSC's learner F, then we collect the gradients by maximizing $\mathcal{L}_{\mathcal{R}}$ and at the same time minimize $\mathcal{L}_{\mathcal{R}'}$.

Algorithm 1 Learning to Obstruct (Our method)

```
Input: pre-trained backbone: $\theta^p$, task distribution: $P(\mathcal{T})$,
    epoch: $I$, learning rate: $\alpha$, restricted classes: $\mathcal{R}$, FSC loss:
    $\mathcal{L}^F$, few-shot learner: $F$.
Output: Obstructive backbone: $\theta$
 1: Intialize $\theta = \theta^p$, and $\phi$ following the FSC method.
 2: for $i = 1$ to $I$ do
 3:     Sample batch $\mathcal{B} : \{\mathcal{T}^{(t)}\}_{t=1}^{|\mathcal{B}|} \sim P(\mathcal{T})$,
        where $\mathcal{T}^{(t)} = (\mathcal{D}^{(t)}_{\texttt{fsc}}, \mathcal{D}^{(t)}_{\texttt{obs}})$
 4:     for $t = 1, \ldots, |\mathcal{B}|$ do
 5:         $\widetilde{\theta}, \widetilde{\phi} = F([\theta, \phi], \mathcal{D}^{(t)}_{\texttt{fsc}})$
 6:         $\Delta\theta^{(t)} = \nabla_\theta \left[\mathcal{L}_{\mathcal{R}'}\left([\widetilde{\theta}, \widetilde{\phi}], \mathcal{D}^{(t)}_{\texttt{obs}}\right) - \mathcal{L}_{\mathcal{R}}\left([\widetilde{\theta}, \widetilde{\phi}], \mathcal{D}^{(t)}_{\texttt{obs}}\right)\right]$
 7:     end for
 8:     $\theta \leftarrow \theta - \alpha \sum_{t=1}^{|\mathcal{B}|} \Delta\theta^{(t)}$
 9: end for
10: return $\theta$
```

Optimization. To solve the optimization problem in Eq. (5), we use a gradient-based method. We illustrate the overall algorithm in Alg. 1 using mini-batch gradient descent. Given a randomly sampled batch of tasks $\mathcal{B}$, the FSC learner F updates the model parameters based on $\mathcal{D}^{(t)}_{\texttt{fsc}}$ separately for each task to produce updated parameters $\widetilde{\theta}$ and $\widetilde{\phi}$. To backpropagate through, the learner defined in Eq. (3), we approximate the arg min with unrolled gradients.

Next, based on whether the example is of a class belonging to $\mathcal{R}$, we compute $\mathcal{L}_{\mathcal{R}}$ and $\mathcal{L}_{\mathcal{R}'}$ over $\mathcal{D}^{(t)}_{\texttt{obs}}$ with updated parameters $\widetilde{\theta}$ and $\widetilde{\phi}$. We then compute the gradient $\Delta\theta^{(t)}$ with bacpropgation on $\mathcal{L}_{\mathcal{R}'} - \mathcal{L}_{\mathcal{R}}$ w.r.t. θ. After collecting all the $\Delta\theta^{(t)}$ within one batch, we update θ with the aggregated gradient. This training procedure allows the model parameters to be steered to an unfavored spot conditioned on $\mathcal{R}$ while maintaining or even enhancing its performance of generalizing to unrestricted classes after applying a downstream few-shot learner.

From classification to multi-label classification. Beyond the classification formulation, LTO can also be formulated to obstruct attribute learning. We treat attribute learning as a multi-label classification problem, i.e., each input $\boldsymbol{x}$ is labeled with a vector label $\boldsymbol{k} \in \mathbb{N}^{|\mathcal{A}|}$, where $\mathcal{A}$ denotes the set of all attributes. For each attribute $a \in \mathcal{A}$, an algorithm F builds a designated predictor and introduces additional parameters ϕ_a to build a model with parameters $\vartheta = [\theta, \phi_{1:|\mathcal{A}|}]$.

In attribute learning, LTO aims to obstruct a set of restricted attributes $\mathcal{R} \subset \mathcal{A}$. The algorithm follows Alg. 1, except we change the objective in Eq. (5) to the following:

$$\mathcal{L}_{\mathcal{R}}([\widetilde{\theta}, \widetilde{\phi}_{1:|\mathcal{A}|}], \mathcal{D}_{obs}^{(t)}) = \sum_{a \in \mathcal{R}} \mathcal{L}^F\left([\widetilde{\theta}, \widetilde{\phi}_a], \mathcal{D}_{obs}^{(t)}\right), \tag{6}$$

and vice versa for $L_{\mathcal{R}'}$.

4.2 Additional Details

Gradient computation for θ. Implementation-wise, to collect all $\Delta\theta^{(t)}$ for each task $\mathcal{T}^{(t)}$ within one batch with the same initial parameters, we restore $[\theta, \phi]$ to the values at the beginning of each epoch. That is, after collecting all $\Delta\theta^{(t)}$s and updating $[\theta, \phi]$ at the end of the epoch, we cache the updated parameters for future restoration.

Resampling prompts and texts for CLIP-based FSC. In CLIP-baed FSC, a prompt set $\mathcal{P}$ contains text templates to guide the output of the text encoder. Each prompt $\boldsymbol{p} \in \mathcal{P}$ usually takes the form of "a photo of {k}" in image classification, where $k \in \mathcal{Y}$ is a class label. The classifiers in CLIP-based FSC are usually built on the text features of $\mathcal{P}$ and $\mathcal{Y}$. That is, for each class label $k \in \mathcal{Y}$, the class feature

$$\boldsymbol{v}_k = g_{[\theta_{\texttt{text}}, \phi]}(k) = \texttt{Agg}_{\boldsymbol{p} \in \mathcal{P}} g_{[\theta_{\texttt{text}}]}(\boldsymbol{p}(k)), \tag{7}$$

where $\texttt{Agg}(\cdot)$ is an aggregaion function.

Due to the limitation of GPU memory, we do not compute the gradients of text features generated from all prompts $\mathcal{P}$ and all classes $\mathcal{Y}$. Instead, for every few steps, we use an unbiased estimate by re-sampling a subset of prompts $\mathcal{P}' \subset \mathcal{P}$ and classes $\mathcal{Y}' \subset \mathcal{Y}$. For each $k \in \mathcal{Y}'$, its correspondent $\boldsymbol{v}_k$ is defined as $\texttt{Agg}_{\boldsymbol{p} \in \mathcal{P}'} g_{[\theta_{\texttt{text}}]}(\boldsymbol{p}(k))$, and only the gradients from these tensors $\{\boldsymbol{v}_k \mid k \in \mathcal{Y}'\}$ are backpropagated.

5 Experiments

We first describe how we construct the restricted class set followed by the dataset preparation in our experiments. Dataset-specific details are described subsequently.

Selection of restricted classes $\mathcal{R}$. To resemble real-world scenarios, choose the set of restricted classes including individual classes that are related to each

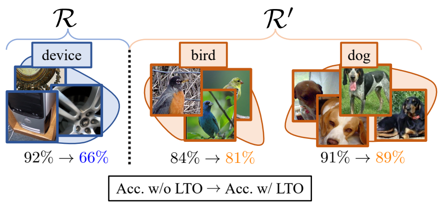

Fig. 2: Selection of $\mathcal{R}$ on image classification. The objective of LTO on image classification is to minimize the top-1 Acc. on $\mathcal{R}$ while maintaining the top-1 Acc. on $\mathcal{R}'$. In this example, $\mathcal{R} = \mathcal{Y}_{\text{device}}$ while $\mathcal{R}' = \mathcal{Y}_{\text{bird}} \bigcup \mathcal{Y}_{\text{dog}}$. LTO decreases the top-1 Acc. on $\mathcal{R}$ from 92% to 66%, while the drops on $\mathcal{R}'$ are no more than 3%.

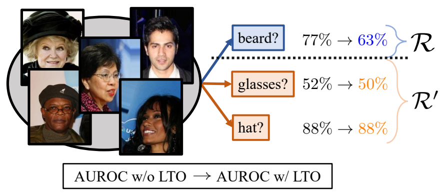

Fig. 3: Selection of $\mathcal{R}$ on attribute learning. The objective of LTO on attribute learning is to minimize the AUROC on $\mathcal{R}$ while maintaining the performance on $\mathcal{R}'$. In this example, $\mathcal{R} = \{\text{bald}\}$ while $\mathcal{R}' = \{\text{hat}, \text{glasses}\}$. LTO decreases the AUROC on $\mathcal{R}$ from 77% to 63%, while the drops on $\mathcal{R}'$ are no more than 2%.

other. Specifically, we divide $\mathcal{Y}$ into N' superclasses $\{\mathcal{Y}_1, \cdots, \mathcal{Y}_{N'}\}$, where each $\mathcal{Y}_n$ consists of k with similar semantics following existing work [8, 27]. Once superclass $\mathcal{Y}_n$ is picked then all subsuming classes $k \in \mathcal{Y}_i$ are categorized into $\mathcal{R}$ while the remaining classes are put into $\mathcal{R}'$. Fig. 2 provides an illustration where we extract 4 superclasses for image classification. Similarly, in Fig. 3, we illustrate how the restricted classes are chosen for attribute learning.

Training and evaluation data split. To show the obstruction effect of LTO, we use FSC algorithm $\mathbf{F}$ with backbone parameters initialized as $\mathbf{A}(\theta^p)$. To avoid data/class information leakage in the experiment setup, carefully consider three disjoint dataset splits $\{\mathcal{D}_{\mathbf{A}}, \mathcal{D}_{\mathbf{F}}, \mathcal{D}_{\texttt{eval}}\}$ each corresponds to data used by LTO, by FSC in meta-training, and the evaluation set for meta-testing. We note that $\mathcal{D}_{\mathbf{A}}$ is used for sampling $\mathcal{D}_{\texttt{fsc}}$ and $\mathcal{D}_{\texttt{obs}}$ in Eq. (4), i.e., $\mathcal{D}_{\mathbf{A}} = \bigcup_{t=1}^{|\mathcal{B}|} \mathcal{T}^{(t)} = \bigcup_{t=1}^{|\mathcal{B}|} (\mathcal{D}_{\texttt{fsc}}^{(t)} \bigcup \mathcal{D}_{\texttt{obs}}^{(t)})$.

Baselines. We compare LTO to two baselines:

- *Only$\mathcal{R}$* aims to "ruin" the backbone by directly maximizing the loss for the restricted set $\mathcal{R}$, i.e.,

$$\max_{\theta} \mathbb{E}_{\mathcal{T}^{(t)}} \left[\mathcal{L}_{\mathcal{R}} \left([\theta, \phi], \mathcal{T}^{(t)} \right) \right]. \tag{8}$$

- *NoF* chooses to minimize $\mathcal{L}_{\mathcal{R}'} - \mathcal{L}_{\mathcal{R}}$ without the consideration of the FSC algorithm $\mathbf{F}$. This is equivalent to removing the computation of $\widetilde{\theta}$ in LTO, i.e.,

$$\min_{\theta} \mathbb{E}_{\mathcal{T}^{(t)}} \left[\mathcal{L}_{\mathcal{R}'} \left([\theta, \phi], \mathcal{T}^{(t)} \right) - \mathcal{L}_{\mathcal{R}} \left([\theta, \phi], \mathcal{T}^{(t)} \right) \right]. \tag{9}$$

For both baselines, for each FSC method, we use the same amount of data and hyperparameters as in LTO for all superclasses.

5.1 Obstruct Classical Few-shot Classification

We perform experiments by choosing $\mathbf{F}$ to be classical FSC methods, including, ProtoNet [46] and MetaOptNet [28]. We demonstrate that by incorporating our

LTO algorithm $\mathbf{A}$, the performance of $\mathbf{F}(\mathbf{A}(\theta^p))$ in the restricted class set $\mathcal{R}$ declines significantly while the performance in $\mathcal{R}'$ is mostly maintained.

Experiment setup. Following the setting of Hu et al. [20], we choose a pre-trained ResNet18 [19] as θ^p. For $\mathcal{R}$ selection on ImageNet, we choose to group classes based on the superclasses provided by Engstrom et al. [8]. In total, there are 10 superclasses and each superclass subsumes 38 of the original ImageNet classes. For each experiment, we select one out of ten superclasses as $\mathcal{R}$. We use the split of the first superclass (id = 0) as the validation task for tuning hyperparameters. We report the result for the rest of the superclasses from id = 1 to 9. Additional details of the superclass are provided in the appendix. For data split, we visualize it in Fig. 4. Since we are evaluating the performance of $\mathbf{F}$ in novel classes, both $\mathcal{R}$ and $\mathcal{R}'$ in $\mathcal{D}_{\texttt{eval}}$ should not show up in $\mathcal{D}_{\mathbf{F}}$. Thus, We further split $\mathcal{R}'$, with 70% classes for $\mathcal{D}_{\mathbf{F}}$ and 30% for $\mathcal{D}_{\texttt{eval}}$.

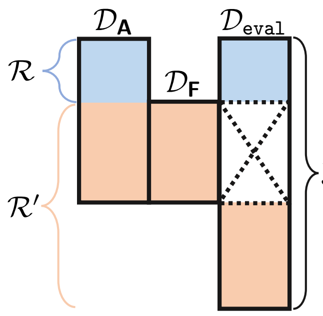
$\mathcal{D}_{\mathbf{A}}$
$\mathcal{D}_{\texttt{eval}}$
$\mathcal{R}$
$\mathcal{D}_{\mathbf{F}}$
$\mathcal{R}'$
$\mathcal{Y}$

Fig. 4: Data split for LTO on classical FSC. For LTO on classical FSC, we split the dataset into three disjoint sets $\mathcal{D}_{\mathbf{A}}$, $\mathcal{D}_{\mathbf{F}}$, and $\mathcal{D}_{\texttt{eval}}$. The set $\mathcal{D}_{\mathbf{A}}$ is for LTO; the set $\mathcal{D}_{\mathbf{F}}$ is for training $\mathbf{F}$ after LTO; $\mathcal{D}_{\texttt{eval}}$ is for evaluation.

Training details. For all of the experiments, we ran 200 steps of obstructive learning with a batch size of 20. Within each obstructive step, we ran 20 gradient steps for the FSC learner F.

Evaluation metric. We assess the quality of obstruction by measuring the *gap in* top-1 classification accuracy with and without LTO. We denote the decrease in top-1 Acc. of FSC when using LTO for the restricted class $\mathcal{R}$ and the other classes $\mathcal{R}'$ as $\delta_{\mathcal{R}}$ and $\delta_{\mathcal{R}'}$ respectively.

For comparison, we propose the evaluation metric $\texttt{DropRatio}@\beta$ denoted as $\Delta@\beta = \frac{\delta_{\mathcal{R}}}{\delta_{\mathcal{R}'}}$, which corresponds to the ratio of the accuracy drop of the restricted classes over the other classes. We select the model when the accuracy drop on the other classes $\delta_{\mathcal{R}'}$ is closest to β%. The higher the drop ratio, the better the obstruction.

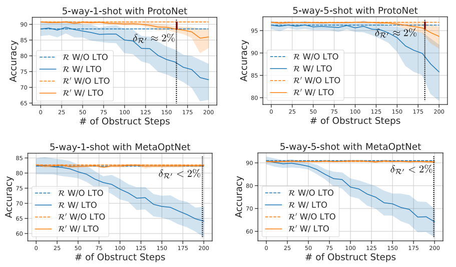

Fig. 5: Accuracy (%) of classical few-shot learning on ImageNet. Gaps between different line types show the effect of LTO. Gaps between different colors show the effect on $\mathcal{R}$ and $\mathcal{R}'$.

Results. In Fig. 5, we visualize the average accuracy vs. the obstructive learning steps on all nine superclasses of ImageNet with and without LTO on the selected classical FSC methods. The gap between the solid and dashed lines illustrates the accuracy drop due to LTO, while the blue and orange lines highlight the restricted and other classes respectively. As can be observed, the gaps are consistently observed across all the plots. For MetaOptNet, we do not observe 2% drop in $\mathcal{R}'$ within 200 steps, hence we selected the model of the last step.

Next, we report the proposed evaluation metric of `DropRatio` with $\beta = 2$, i.e., $\Delta@2$ when obstructed using the baselines and our LTO for comparison. Intuitively, this corresponds to the ratio between the gap (between orange and blue lines) in accuracy at the vertical line shown in Fig. 5. Results are shown in Tab. 1. Higher the `DropRatio` indicates stronger obstruction on the FSC method. For the column of LTO *highlighted in pink*, we observe that on average $\Delta@2$ is much larger than 1 under all settings, which means that our method can obstruct $\mathcal{R}$ effectively without ruining the accuracy in $\mathcal{R}'$. Overall, our method worsens the model's performance in the restricted classes while maintaining the accuracy of other classes.

Table 1: `DropRatio` of LTO on classical few-shot learning. We report $\Delta@2$ on ImageNet over 9 selected superclasses. All experiments are 5-way classification.

F	$\mathcal{R}$	1-shot			5-shot		
		Only$\mathcal{R}$	*NoF*	Ours	*Only*$\mathcal{R}$	*NoF*	Ours
ProtoNet [20, 46]	1	1.07	2.21	**7.85**	0.93	0.96	**1.69**
	2	1.22	3.38	**3.93**	1.15	2.31	**2.63**
	3	1.05	2.15	**3.21**	0.94	**1.98**	0.05
	4	1.09	**6.91**	6.60	1.36	2.78	**2.79**
	5	1.37	**2.54**	1.89	1.25	1.39	3.19
	6	1.05	4.82	**6.62**	1.33	2.59	**2.90**
	7	1.05	2.96	**4.30**	1.08	2.33	**4.51**
	8	1.00	**4.62**	2.59	0.84	0.89	**3.16**
	9	1.03	**4.28**	2.80	1.02	**2.82**	0.69
	Avg.	1.10	3.77	**4.42**	1.10	2.00	**2.40**
MetaOptNet [20, 28]	1	2.35	**12.06**	8.71	2.64	**16.45**	14.08
	2	2.21	**7.14**	6.41	2.02	6.99	**11.32**
	3	2.08	3.53	**5.22**	1.74	5.31	**7.84**
	4	2.52	**12.05**	9.42	2.39	14.10	**14.98**
	5	1.56	4.94	**6.84**	1.53	7.34	**10.07**
	6	1.57	14.54	**15.11**	2.06	13.38	**22.44**
	7	1.60	8.88	**9.61**	1.91	9.77	**13.94**
	8	2.27	13.53	**17.01**	1.51	12.87	**21.57**
	9	**1.37**	1.16	1.34	1.63	**4.79**	4.34
	Avg.	1.95	8.65	**8.85**	1.94	10.11	**13.40**

5.2 Obstruct CLIP-based Few-shot Classification.

We perform experiments using CLIP-based few-shot algorithms, including CoOp [63], TipAdapter [60], and a simple baseline method, named Cross-Entropy Fine-Tune(CE), which optimizes the CLIP's parameters by minimizing cross-entropy loss. We demonstrate that LTO can also obstruct foundation models with rich knowledge of both language and vision.

Experiment setup. We select CLIP-ResNet50 [38] as the pre-trained backbone. We use ImageNet [7] and CIFAR100 [27] to evaluate the performance of our LTO on clip-based image classification. For restricted classes $\mathcal{R}$ selection on CIFAR100, we follow the superclass groups provided by Krizhevsky et al. [27]. The 100 classes in CIFAR100 are categorized into 20 superclasses, each containing five classes. For each experiment, we select one of the 20 superclasses as $\mathcal{R}$ while the rest as $\mathcal{R}'$. The first superclass (id = 0) is reserved as the validation task for hyperparameter tuning. For $\mathcal{R}$ selection on ImageNet is the same as in Sec. 5.1.

Unlike classical FSC methods, CLIP-based FSC methods [60, 63] focus on learning for one single task, i.e., the $|\mathcal{Y}|$-way K-shot support set with the entire testing split as the query set. Hence, $\mathcal{D}_{\mathsf{A}}$, $\mathcal{D}_{\mathsf{F}}$, and $\mathcal{D}_{\mathtt{eval}}$ all have access to every classes in $\mathcal{R}$ and $\mathcal{R}'$. Additionally, since few-shot learning is motivated by the scarcity of high-quality labeled data, affecting both LTO and FSC for evaluation, we set $\mathcal{D}_{\mathsf{A}}$ to have the same few-shot setting as $\mathcal{D}_{\mathsf{F}}$. Specifically, for both ImageNet and CIFAR100, we sample a $|\mathcal{Y}|$-way 5-shot set from the training split as $\mathcal{D}_{\mathsf{A}}$ and another $|\mathcal{Y}|$-way 5-shot set as $\mathcal{D}_{\mathsf{F}}$. For each batch task $\mathcal{T}^{(t)}$, we sample a $|\mathcal{Y}|$-way 1-shot set from $\mathcal{D}_{\mathsf{A}}$ as $\mathcal{D}^{(t)}_{\mathtt{fsc}}$ and a $|\mathcal{Y}|$-way 4-shot set as $\mathcal{D}^{(t)}_{\mathtt{obs}}$.

Training details. For CoOp [63] and TipAdapter [60], we follow their training hyperparameters when used for evaluating LTO. Additional training details are in the appendix.

Results. We report on combinations of LTO methods and few-shot algorithms $\mathbf{F}$. Each $\mathbf{A}(\theta^p)$ under the same adaptation algorithm $\mathbf{F}$ for evaluation. All $\mathbf{F}$ are trained with 5-shot data. In Tab. 2, we report Δ@2 on CIFAR100 and ImageNet over the first 9 superclasses. On both datasets, introducing F for an intermediate $\widetilde{\theta}$ generally brings a consistent improvement on Δ@2. This suggests LTO is effective at obstructing the restricted classes without hurting other classes. Overall, on ImageNet we can obstruct the performance of CE in $\mathcal{R}$ by 7.65%, CoOp by 4.58%, and Tip-Adapter by 5.86%.

Table 2: `DropRatio` of Clip-based LTO on CIFAR100 and ImageNet. We report Δ@2 on CIFAR100 and ImageNet over 9 selected superclasses. Note, superclasses are not the same across datasets.

$\mathbf{F}$	$\mathcal{R}$	CIFAR100			ImageNet		
		Only$\mathcal{R}$	*NoF*	Ours	*Only*$\mathcal{R}$	*NoF*	Ours
CE	1	3.68	**17.66**	14.55	2.79	4.82	**9.93**
	2	0.69	9.87	**15.22**	2.09	0.88	**4.84**
	3	1.71	3.88	**9.96**	1.08	1.18	**6.89**
	4	0.35	2.71	**15.99**	2.01	1.11	**9.15**
	5	1.66	4.09	**4.56**	1.20	0.96	**2.72**
	6	0.31	13.26	**18.00**	7.64	6.06	**15.86**
	7	0.84	3.05	**15.64**	5.89	2.77	**8.84**
	8	0.60	1.09	**10.82**	0.97	0.78	**6.59**
	9	3.45	**3.52**	3.17	1.53	2.30	**4.02**
	Avg.	1.48	6.57	**11.99**	2.80	2.32	**7.65**
CoOp [63]	1	6.56	**8.44**	5.91	1.15	2.23	**6.15**
	2	0.56	**3.49**	3.03	1.02	2.47	**8.49**
	3	0.83	6.56	**8.96**	1.51	**4.54**	3.29
	4	0.73	**7.95**	6.99	0.57	2.97	**5.55**
	5	1.21	**3.89**	1.58	1.30	**2.70**	1.93
	6	0.31	5.94	**6.80**	0.84	1.91	**5.02**
	7	1.82	5.78	**11.40**	1.13	2.84	**3.91**
	8	1.35	4.85	**12.02**	1.08	1.44	**3.01**
	9	2.83	**8.44**	3.87	2.14	3.47	**3.95**
	Avg.	1.80	6.15	**6.73**	1.19	2.73	**4.58**
Tip-Adapter [60]	1	3.68	9.57	**15.09**	1.37	2.78	**5.92**
	2	0.85	3.41	**17.12**	2.04	3.26	**4.89**
	3	1.93	**10.21**	6.39	1.76	2.83	**5.59**
	4	0.62	1.07	**7.48**	1.76	4.00	**7.49**
	5	4.63	3.04	**6.78**	1.27	2.25	**3.25**
	6	3.45	3.66	**8.64**	4.74	3.99	**8.57**
	7	0.92	4.47	**11.17**	3.10	2.36	**7.41**
	8	1.28	0.82	**6.39**	1.81	1.88	**4.97**
	9	1.35	3.65	**12.39**	1.62	1.41	**4.64**
	Avg.	2.08	4.43	**10.16**	2.16	2.75	**5.86**

Data efficiency. It is intuitive that after applying LTO, if more data is available to the FSC $\mathcal{T}$, then can overcome the obstruction. In Tab. 3, we test out this scenario by applying more data in the FSC methods for evaluation, denoted as $\mathbf{F}'$, instead of using the same amount of data for both $\mathbf{F}$ and $\mathbf{F}'$. We study the performance on superclass id = 1. We note that $\mathcal{D}_{\mathbf{A}}$ is still a 5-shot set. A data multiplier of value $m_{\texttt{data}}\times$ denotes that the training data of $\mathbf{F}'$ is $(5m_{\texttt{data}})$-shot. We observe that for CE and CoOp [63], it is impactful for them to use more training data, yet it cannot fully overcome the obstruction. In the case for Tip-Adapter [60], the difference between using FSC with 20-shot and 5-shot is 2.24% on Δ@2. We can conclude that the $\mathbf{F}'$ for evaluation must leverage a lot more data than that $\mathbf{F}$ in LTO had used to recover the degraded accuracy. Even then, only a partial of that performance is recovered.

Time Efficiency. Next, we study whether training longer in the FCS methods can overcome the obstruction caused by LTO. In Tab. 4, we report the Δ@2 when FSG uses more training epochs. An epoch multiplier of value $m_{\texttt{epoch}}\times$ denotes that the training epoch of FSG is m times the one it originally was. We observe that although increasing the training epoch decreases Δ@2, the gap is not fully recoverable. Especially for `Tip-Adapter` [60], by training for 4× longer for, the improvement is a 2.76% on Δ@2. However, in the case of `CoOp` [63], it is impactful for them to use more training epochs. It is shown that the $\mathbf{F}'$ for

Table 3: Data efficiency. We report the Δ@2 of our LTO on ImageNet obstructing superclass id=1 when $|\mathcal{D}_{\mathsf{F}}|$ is $m_{\texttt{data}}$, i.e. the data multiplier, times of $|\mathcal{D}_{\mathsf{A}}|$. TA: Tip-Adapter [60]. CO: CoOp [63].

F	$m_{\texttt{data}}$			
	1×	2×	3×	4×
CE	9.93	6.34	2.76	2.82
CO	6.15	2.46	2.26	2.32
TA	5.92	3.09	3.06	3.76

Table 4: Time efficiency. We report the Δ@2 of our LTO on ImageNet obstructing superclass id=1 when the training epoch of FCS methods is $m_{\texttt{time}}$, i.e. the epoch multiplier, times of its original.

F	$m_{\texttt{time}}$			
	1×	2×	3×	4×
CE	9.93	3.06	4.66	3.64
CO	6.15	4.66	2.52	1.99
TA	5.92	2.68	3.41	3.16

Table 5: Cross-(F, F'). We report the Δ@2 of our LTO on ImageNet obstructing superclass id=1 when the FSC methods F and F' can mismatch. TA: Tip-Adapter [60], CO: CoOp [63].

F	F'			
	CE	CO	TA	Avg.
CE	9.93	4.71	7.33	7.32
CO	4.79	6.15	4.34	5.09
TA	4.16	7.75	5.92	5.94

evaluation must have access to a lot more computation resources than what F had used to overcome some of the degraded performance.

Mismatched FSC. All previous experiments assume, we use the same FSG method in our LTO and during evaluation. We now experiment with a mismatch in FSG, i.e., we use F in LTO and another FSC method F' for evaluation. In Tab. 5, we report the Δ@2 on different combinations of $(\mathsf{F}, \mathsf{F}')$. We observe that the drop ratio is indifferent to which F was used during the obstruction. When the LTO uses `Tip-adapter`, the difference of Δ@2 between $\mathsf{F}' = \texttt{CE}$ and $\mathsf{F}' = \texttt{Tip-adapter}$ is only 1.76% while $\mathsf{F}' = \texttt{CoOp}$ is actually more vulnerable and gain 1.83% on Δ@2.

5.3 Obstruct CLIP-based Attribute Learning

Experiment setup. We select CLIP-ResNet50 [38] as the pre-trained backbone. We conduct experiments on the CelebA [31] which contains 40 annotated attributes annotated for face images. We follow the setup by Gannamaneni et al. [13] and select 12 appearance attributes out of 40, including "EyeClasses", "Wearing_Hat", "Bald", etc. In each experiment, 1 out of the 12 selected appearance attributes is selected to be the restricted class$\mathcal{R}$, while the rest 11 attributes belong to $\mathcal{R}'$. We use the first attribute (id $= 0$) as the validation task for tuning hyperparameters. We report the result for the rest of the attributes from id $= 1 \sim 11$. To evaluate the performance of attribute learning we use AUROC (Area Under the Receiver Operating Characteristic curve) as the main metric.

FSC method. We extend the CE learner from Sec. 5.2 to conduct attribute learning. Following the naive prompt settings [13], for each attribute $a \in \mathcal{A}$, we build a binary classifier. The classifier consists of two features, a positive and a negative prompt. The classifier makes a prediction $\hat{a}$ by determining which feature has a higher product with the visual feature of input image $\boldsymbol{v}_{\boldsymbol{x}}$.

Results. In Tab. 6, we report the Δ@2 of LTO with $\mathsf{F} = \texttt{CE}$ on CelebA attribute learning. We observe consistent drops of AUROC on all attributes, while some attributes are more vulnerable than others to LTO. Specifically, for ours,

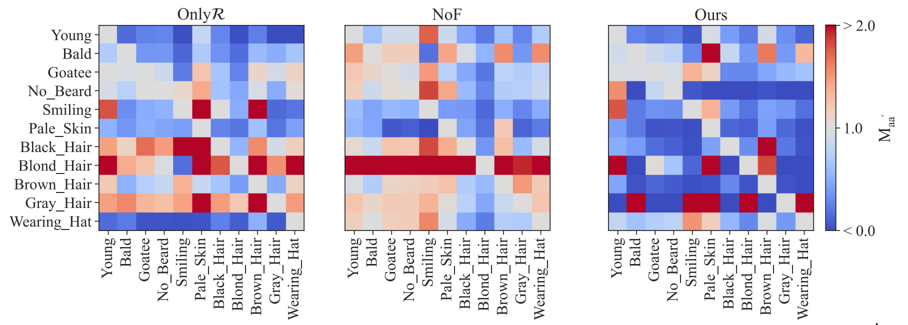

Fig. 6: Confusion matrix on CelebA attribute learning. Each cell $M_{aa'} = \delta_a^{a'}/\delta_a^a$ where δ_j^i denotes the decrease of AUROC in attribute i when LTO obstruct $\mathcal{R} = \{j\}$ for all $i, j \in \mathcal{A}$. Ideally, all non-diagonal values should be as small (blue) as possible.

attribute "Pale_Skin"(id= 6) is the easiest to obstruct with $\Delta@2 = 31.64\%$ while attribute 'Gray_Hair"(id= 10) is the hardest to obstruct with $\Delta@2\% = 0.08$. In comparison to the baselines, ours also achieves the most significant $\Delta@2$ on most attributes. Overall, our LTO can decrease the average of AUROC by $\Delta@2 = 8.04\%$.

Table 6: CLIP-based LTO on CelebA attribute learning. We report the $\Delta@5$ for each selected attributes.

$\mathcal{R}$	*Only$\mathcal{R}$*	*NoF*	Ours	$\mathcal{R}$	*Only$\mathcal{R}$*	*NoF*	Ours
1	**5.56**	2.96	4.04	7	1.69	**2.48**	2.38
2	1.01	3.58	**5.20**	8	0.77	0.48	**1.62**
3	0.79	0.54	**1.11**	9	7.26	4.13	**10.01**
4	0.29	0.57	**9.48**	10	**0.87**	0.72	0.08
5	0.75	0.96	**1.96**	11	**7.67**	1.07	4.94
6	5.40	10.38	**31.64**	Avg.	3.56	3.10	**8.04**

Confusion among attributes. We define the confusion matrix $\boldsymbol{M} \in \mathbb{R}^{|\mathcal{A}|\times|\mathcal{A}|}$ as follows, each cell $M_{aa'}$ is the ratio $\delta_a^{a'}/\delta_a^a$ where δ_j^i denotes the decrease of AUROC in attribute i when LTO obstruct $\mathcal{R} = \{j\}$ for all $i, j \in \mathcal{A}$. If $M_{aa'} < 1$, then the performance drop on attribute a' is smaller than the performance drop on attribute a, which is a success case for our LTO. On the other hand, if $M_{aa'} > 1$, then the performance drop on attribute a' is larger than the performance drop on attribute a, which is considered a failure of obstruction. This visualization also shows whether two attributes a and a' are related during obstruction, i.e., did LTO obstruct a while unintentionally also obstructing a'?

In Fig. 6, we visualize the confusion matrices of *Only$\mathcal{R}$*, *NoF*, and ours on the 12 selected attributes. Ideally, any non-diagonal $M_{aa'}$, where $a \neq a'$, should be as small (blue) as possible. Visually speaking, our LTO method outperforms the two baselines. LTO successfully lowers the performance of restricted attributes during obstruction learning. We also observed that certain attribute pairs with a strong correlation in the confusion matrix of our LTO. For example, it is hard to obstruct "Blond_hair" without obstructing "Brown_hair", which is understandable since the colors are similar. There are also high correlation between ("Gray_Hair", "Bald") or ("No_beard", "Young").

Additional comparison with unlearning for obstruction. While machine unlearning is not designed for obstruction, the unlearning algorithms can be used to modify a pre-trained model. In this section, we benchmark against machine unlearning and show that LTO is not just unlearning the classes, but also *making them more difficult to learn back*. We choose SSD [11], a state-of-the-art machine unlearning algorithm on CIFAR100 for comparison. The experiment setup follows the setting in Sec. 5.2. We apply SSD on the pre-trained CLIP-RN50 weights by setting the forget set to be the set of restricted classes $\mathcal{R}$, while the retain set is set to $\mathcal{R}'$. Note that we are considering, i.e., if $\mathcal{D}_A$ is 5-shot then forget and retrain set are also 5-shot.

With the pre-trained model unlearned, we then apply few-shot algorithms $\mathbf{F} \in \{\texttt{CE}, \texttt{CoOp}, \texttt{TipAdapter}\}$ with 100-way 5-shot to learn over both $\mathcal{R}$ and $\mathcal{R}'$. We report the drop ratio Δ@2 as in Tab. 2. As shown in Tab. 7, the results on the first 9 superclasses in CIFAR100 show the averages Δ@2 of SSD are 1.09/1.17/1.07 for each $\mathbf{F}$ while ours are 11.99/6.73/10.16. We observe that SSD is not able to unlearn the forget set while maintaining the performance on the retain set; We suspect that this is because SSD is not designed for the few-shot setting. Hence, we then increase $\mathcal{D}_A$ for both methods to 20-shot to study how the size of the training set affects their performance. Although SSD benefits from utilizing more training data, 20-shot SSD is still not comparable to 5-shot ours on most superclasses. Moreover, 20-shot LTO outperforms 20-shot SSD significantly under all scenarios.

Table 7: `DropRatio` of unlearning method SSD [11] versus our LTO on CIFAR100. We report Δ@2 on CIFAR100 over 9 selected superclasses. Each method is trained on $\mathcal{D}_A$ under two few-shot settings, i.e. 5-shot and 20-shot, but evaluated on 5-shot $\mathbf{F}$.

F	$\mathcal{R}$	5-shot $\mathcal{D}_A$ SSD	5-shot $\mathcal{D}_A$ Ours	20-shot $\mathcal{D}_A$ SSD	20-shot $\mathcal{D}_A$ Ours
CE	1	0.67	**14.55**	2.22	**21.87**
CE	2	1.28	**15.22**	7.62	**34.41**
CE	3	1.17	**9.96**	0.36	**15.23**
CE	4	1.31	**15.99**	1.69	**12.25**
CE	5	1.23	**4.56**	1.23	**15.29**
CE	6	1.01	**18.00**	7.57	**24.34**
CE	7	1.03	**15.64**	1.14	**18.62**
CE	8	0.64	**10.82**	2.76	**18.14**
CE	9	1.42	**3.17**	2.23	**20.28**
	Avg.	1.09	**11.99**	2.98	**20.04**
CoOp [63]	1	0.74	**5.91**	3.72	**21.25**
CoOp [63]	2	1.23	**3.03**	6.68	**16.81**
CoOp [63]	3	1.18	**8.96**	3.16	**9.33**
CoOp [63]	4	1.30	**6.99**	7.60	**14.28**
CoOp [63]	5	1.31	**1.58**	1.33	**11.57**
CoOp [63]	6	1.11	**6.80**	10.03	**20.00**
CoOp [63]	7	0.88	**11.40**	2.62	**11.81**
CoOp [63]	8	0.68	**12.02**	3.12	**26.32**
CoOp [63]	9	1.35	**3.87**	2.89	**16.77**
	Avg.	1.17	**6.73**	4.57	**16.46**
Tip-Adapter [60]	1	0.69	**15.09**	3.30	**11.62**
Tip-Adapter [60]	2	1.09	**17.12**	4.57	**9.31**
Tip-Adapter [60]	3	1.17	**6.39**	3.38	**9.30**
Tip-Adapter [60]	4	1.27	**7.48**	4.00	**6.67**
Tip-Adapter [60]	5	1.31	**6.78**	1.32	**9.82**
Tip-Adapter [60]	6	1.11	**8.64**	7.63	**14.89**
Tip-Adapter [60]	7	0.85	**11.17**	1.83	**12.89**
Tip-Adapter [60]	8	0.76	**6.39**	3.18	**14.17**
Tip-Adapter [60]	9	1.34	**12.39**	2.73	**7.58**
	Avg.	1.07	**10.16**	3.55	**10.69**

6 Conclusion

We propose LTO that learns a *poor initialization* of the backbone to obstruct FSC on restricted classes. Empirically, LTO successfully obstructs four FSC methods over image classification and attribute classification tasks. Please note that we study the obstruction of FSC with a more significant objective in mind. We aim to ensure the safer release of open-source models. By developing pre-trained models that are difficult to fine-tune for certain downstream tasks, we can potentially prevent the misuse of released models on known harmful applications, while maintaining the open-source culture of the computer vision community. We believe that LTO on FSC represents a promising step towards this goal.

Acknowledgements

This project is supported in part by an NSF Award #2420724.

References

1. Cao, Y., Yang, J.: Towards making systems forget with machine unlearning. In: IEEE Symposium on Security and Privacy. (2015)
2. Chen, W.Y., Liu, Y.C., Kira, Z., Wang, Y.C., Huang, J.B.: A closer look at few-shot classification. In: Proc. ICLR (2019)
3. Chen, Y., Liu, Z., Xu, H., Darrell, T., Wang, X.: Meta-baseline: Exploring simple meta-learning for few-shot learning. In: Proc. ICCV (2021)
4. Chen, Z., Fu, Y., Wang, Y.X., Ma, L., Liu, W., Hebert, M.: Image deformation meta-networks for one-shot learning. In: Proc. CVPR (2019)
5. Dai, J., Chen, C., Li, Y.: A backdoor attack against lstm-based text classification systems. IEEE Access (2019)
6. Dean, J.: A Golden Decade of Deep Learning: Computing Systems & Applications. Daedalus (2022)
7. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: A large-scale hierarchical image database. In: Proc. CVPR (2009)
8. Engstrom, L., Ilyas, A., Salman, H., Santurkar, S., Tsipras, D.: Robustness (Python Library) (2019), URL https://github.com/MadryLab/robustness
9. Fei-Fei, L., Fergus, R., Perona, P.: One-shot learning of object categories. IEEE TPAMI (2006)
10. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: Proc. ICML (2017)
11. Foster, J., Schoepf, S., Brintrup, A.: Fast machine unlearning without retraining through selective synaptic dampening. In: Proc. AAAI (2023)
12. Frikha, A., Krompaß, D., Köpken, H.G., Tresp, V.: Few-shot one-class classification via meta-learning. In: Proc. AAAI (2021)
13. Gannamaneni, S.S., Sadaghiani, A., Rao, R.P., Mock, M., Akila, M.: Investigating CLIP performance for meta-data generation in ad datasets. In: Proc. CVPR (2023)
14. Gao, P., Geng, S., Zhang, R., Ma, T., Fang, R., Zhang, Y., Li, H., Qiao, Y.: CLIP-Adapter: Better vision-language models with feature adapters. IJCV (2023)
15. Golatkar, A., Achille, A., Ravichandran, A., Polito, M., Soatto, S.: Mixed-privacy forgetting in deep networks. In: Proc. CVPR (2021)
16. Hao, F., He, F., Cheng, J., Wang, L., Cao, J., Tao, D.: Collect and select: Semantic alignment metric learning for few-shot learning. In: Proc. ICCV (2019)
17. Hariharan, B., Girshick, R.: Low-shot visual recognition by shrinking and hallucinating features. In: Proc. ICCV (2017)
18. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: Proc. CVPR (2017)

19. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proc. CVPR (2016)
20. Hu, S.X., Li, D., Stühmer, J., Kim, M., Hospedales, T.M.: Pushing the limits of simple pipelines for few-shot learning: External data and fine-tuning make a difference. In: Proc. CVPR (2022)
21. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proc. CVPR (2017)
22. Huh, M., Agrawal, P., Efros, A.A.: What makes imagenet good for transfer learning? arXiv preprint arXiv:1608.08614 (2016)
23. Jamal, M.A., Qi, G.J.: Task agnostic meta-learning for few-shot learning. In: Proc. CVPR (2019)
24. Kang, D., Kwon, H., Min, J., Cho, M.: Relational embedding for few-shot classification. In: Proc. ICCV (2021)
25. Koh, P.W., Liang, P.: Understanding black-box predictions via influence functions. In: Proc. ICML (2017)
26. Kornblith, S., Shlens, J., Le, Q.V.: Do better ImageNet models transfer better? In: Proc. CVPR (2019)
27. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
28. Lee, K., Maji, S., Ravichandran, A., Soatto, S.: Meta-learning with differentiable convex optimization. In: Proc. CVPR (2019)
29. Li, Y., Liang, F., Zhao, L., Cui, Y., Ouyang, W., Shao, J., Yu, F., Yan, J.: Supervision exists everywhere: A data efficient contrastive language-image pre-training paradigm. arXiv preprint arXiv:2110.05208 (2021)
30. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proc. CVPR (2017)
31. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: Proc. ICCV (December 2015)
32. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proc. CVPR (2015)
33. Oldewage, E.T., Bronskill, J.F., Turner, R.E.: Adversarial attacks are a surprisingly strong baseline for poisoning few-shot meta-learners. In: I Can't Believe It's Not Better Workshop: Understanding Deep Learning Through Empirical Falsification (2022)
34. Oreshkin, B., Rodríguez López, P., Lacoste, A.: Tadam: Task dependent adaptive metric for improved few-shot learning. In: Proc. NeurIPS (2018)
35. Payton, T., Claypoole, T.: Privacy in the age of Big data: Recognizing threats, defending your rights, and protecting your family. Rowman & Littlefield (2023)
36. Phan, N., Wang, Y., Wu, X., Dou, D.: Differential privacy preservation for deep auto-encoders: an application of human behavior prediction. In: Proc. AAAI (2016)
37. Qiao, S., Liu, C., Shen, W., Yuille, A.L.: Few-shot image recognition by predicting parameters from activations. In: Proc. CVPR (2018)
38. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: Proc. ICML (2021)

39. Rajeswaran, A., Finn, C., Kakade, S.M., Levine, S.: Meta-learning with implicit gradients. In: Proc. NeurIPS (2019)
40. Ravi, S., Larochelle, H.: Optimization as a model for few-shot learning. In: Proc. ICLR (2016)
41. Redmon, J., Farhadi, A.: YOLO9000: better, faster, stronger. In: Proc. CVPR (2017)
42. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: MobileNetV2: Inverted residuals and linear bottlenecks. In: Proc. CVPR (2018)
43. Schwartz, E., Karlinsky, L., Shtok, J., Harary, S., Marder, M., Kumar, A., Feris, R., Giryes, R., Bronstein, A.: Delta-encoder: an effective sample synthesis method for few-shot object recognition. In: Proc. NeurIPS (2018)
44. Schwarzschild, A., Goldblum, M., Gupta, A., Dickerson, J.P., Goldstein, T.: Just how toxic is data poisoning? a unified benchmark for backdoor and data poisoning attacks. In: Proc. ICML (2021)
45. Shafahi, A., Huang, W.R., Najibi, M., Suciu, O., Studer, C., Dumitras, T., Goldstein, T.: Poison frogs! targeted clean-label poisoning attacks on neural networks. In: Proc. NeurIPS (2018)
46. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. In: Proc. NeurIPS (2017)
47. Song, H., Dong, L., Zhang, W.N., Liu, T., Wei, F.: CLIP models are few-shot learners: Empirical studies on VQA and visual entailment. In: Proc. ACL (2022)
48. Sonnenburg, S., Braun, M.L., Ong, C.S., Bengio, S., Bottou, L., Holmes, G., LeCunn, Y., Muller, K.R., Pereira, F., Rasmussen, C.E., et al.: The need for open source software in machine learning. Journal of Machine Learning Research (2007)
49. Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P.H., Hospedales, T.M.: Learning to compare: Relation network for few-shot learning. In: Proc. CVPR (2018)
50. Sung, Y.L., Cho, J., Bansal, M.: Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks. In: Proc. CVPR (2022)
51. Tarun, A.K., Chundawat, V.S., Mandal, M., Kankanhalli, M.: Fast yet effective machine unlearning. IEEE Transactions on Neural Networks and Learning Systems (2023)
52. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. In: Proc. NeurIPS (2016)
53. Wang, J., Liu, Y., Wang, X.E.: Are gender-neutral queries really gender-neutral? mitigating gender bias in image search. In: Proc. EMNLP (2021)
54. Wang, Y.X., Girshick, R., Hebert, M., Hariharan, B.: Low-shot learning from imaginary data. In: Proc. CVPR (2018)
55. Wertheimer, D., Tang, L., Hariharan, B.: Few-shot classification with feature map reconstruction networks. In: Proc. CVPR (2021)
56. Xiao, H., Biggio, B., Brown, G., Fumera, G., Eckert, C., Roli, F.: Is feature selection secure against training data poisoning? In: Proc. ICML (2015)
57. Xiao, J., Hays, J., Ehinger, K.A., Oliva, A., Torralba, A.: Sun database: Large-scale scene recognition from abbey to zoo. In: Proc. CVPR (2010)

58. Yang, Z., Wang, J., Zhu, Y.: Few-shot classification with contrastive learning. In: Proc. ECCV (2022)
59. Yoon, S.W., Seo, J., Moon, J.: Tapnet: Neural network augmented with task-adaptive projection for few-shot learning. In: Proc. ICML (2019)
60. Zhang, R., Zhang, W., Fang, R., Gao, P., Li, K., Dai, J., Qiao, Y., Li, H.: Tip-adapter: Training-free adaption of CLIP for few-shot classification. In: Proc. ECCV (2022)
61. Zheng, A.Y., Yeh, R.A.: Imma: Immunizing text-to-image models against malicious adaptation. In: Proc. ECCV (2024)
62. Zhou, K., Yang, J., Loy, C.C., Liu, Z.: Conditional prompt learning for vision-language models. In: Proc. CVPR (2022)
63. Zhou, K., Yang, J., Loy, C.C., Liu, Z.: Learning to prompt for vision-language models. IJCV (2022)