Argmax Centroids

Anonymous Author(s)
Affiliation
Address
email

Abstract

We propose a general method to construct centroid approximation for the distribution of maximum points of a random function (a.k.a. argmax distribution), which finds broad applications in machine learning. Our method optimizes a set of centroid points to compactly approximate the argmax distribution with a simple objective function, without explicitly drawing exact samples from the argmax distribution. Theoretically, the argmax centroid method can be shown to minimize a bound in term of Wasserstein distance with the ground-truth argmax distribution under proper conditions. We demonstrate the applicability and effectiveness of our method on a variety of real-world multi-task learning applications, including few-shot image classification, personalized dialogue systems and multi-target domain adaptation.

1 Introduction

Many problems in machine learning and statistics involve optimizing a random function such as the empirical loss. This work focuses on approximating the distribution of the optimum points of a random function of interest. Specifically, we consider the following problem:

The Argmax Centroid Problem *Assume we are given a random function $f_\xi(\theta)$ where ξ is a random variable and θ is a variable of interest. We are interested in estimating the distribution ρ^* of the minimum points $\theta_\xi := \arg\min_\theta f_\xi(\theta)$ as we draw ξ randomly. In particular, we want to find a set of "centroid points" $\{\theta_i\}_{i=1}^n$, whose empirical measure is close to ρ^* in terms of Wasserstein distance.*

This problem can find applications in a variety of machine learning and statistics techniques, including bootstrap (Efron & Tibshirani, 1994), random MAP (Hazan et al., 2013), Thompson sampling (Russo et al., 2017), as well as multi-task learning and meta learning. One naive method to approximate ρ^*, which is widely used in the methods above, is to draw a *i.i.d.* sample of ξ and calculate the corresponding argmax point. However, this approach is computationally expensive because calculating each argmax point requires to solve an independent optimization problem and the quality of Monte Carlo approximation is poor unless n is very large, especially for large-scale optimization problems in deep learning.

In this work, we propose a more efficient centroid approximation to replace the Monte Carlo sampling, in which we explicitly optimize a set of points $\boldsymbol{\theta} = \{\theta_i\}_{i=1}^n$ to approximate target distribution ρ^*. By carefully choosing the location of each point θ_i (rather than drawing them randomly), our method allows us to obtain a set of points $\boldsymbol{\theta}$ that are well aligned to approximate the overall argmax distribution ρ^*. Theoretically, under proper condition, our method can be viewed as minimizing a bound of Wasserstein distance between the empirical distribution of the centroids and the ground-truth distribution. Therefore, given a small budget on sample size, we can estimate the argmax distribution more accurately than using Monte Carlo estimation.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

Argmax centroids can find applications in a variety of machine learning methods and problems. In our empirical studies in this work, we mainy focus on meta learning and multi-task learning tasks. For these problems, we learn from a distribution of datasets/tasks/domains and ξ denotes a random dataset drawn from some population. Using argmax centroids allows us to obtain an ensemble of centroid models that can capture the uncertainty and variation in different datasets and domains.

We test our method on multiple settings, e.g., few-shot image classification, personalized dialogue systems, and multi-target domain adaptation. For all the above settings and tasks, our method can be easily implemented and can enhance the performance of a number of baseline methods, including but not limited to some of the recent state-of-the-art (SOTA) methods, e.g. IFSL (Yue et al., 2020).

2 Methodology

We first introduce our problem description and propose our algorithm to approximate the distribution of maxima over a random function. We analyze our algorithm theoretically and show its property.

Problem description Many problems in machine learning can be framed as minimizing or maximizing random functions $f_\xi(\theta)$ where ξ is some random variable drawn from a distribution π on space Ξ and $\theta \in \Theta \subseteq \mathbb{R}^d$. That is,

$$\theta_\xi = \arg\min_{\theta \in \Theta} f_\xi(\theta). \tag{1}$$

We assume the minimum of $f_\xi(\theta)$ is unique. Denote by ρ^* the distribution of θ_ξ as $\xi \sim \pi$, which we call the argmin distribution of f. We are interested in approximating ρ^*. The problem in (1) finds wide applications in machine learning in various different ways, including random MAP (Hazan et al., 2013), bootstrapping (Efron & Tibshirani, 1994), and multi-domain/meta learning as we elaborate in Section 3.

As mentioned in Section 1, a naive method to approximate ρ^* is to draw *i.i.d.* samples $\{\xi_i\}_{i=1}^n$ from π and calculate the corresponding argmin points θ_{ξ_i} for each particle respectively. However, the quality of Monte Carlo approximation is poor when n is small (Kuo & Sloan, 2005). In practice, it is not affordable to draw a large number of argmax points for large-scale optimization problems in deep learning, both because it requires to solve an optimization for calculating each θ_{ξ_i}, and it is memory-hungry to store a large number of θ_{ξ_i} when the parameter dimension is high.

Argmax Centroids We propose a more efficient centroid-like approximation to replace Monte Carlo sampling, where we explicitly optimize a set of points $\boldsymbol{\theta} = \{\theta_i\}_{i=1}^n$ to approximate target distribution ρ^*. By carefully choosing the location of of each point θ_i (rather than drawing them randomly), our method allows us to obtain a set of diverse points $\boldsymbol{\theta}$ that are well aligned to the overall distribution ρ^* in Wasserstein distance. This is achieved by minimizing the following objective function:

$$\min_{\{\theta_i\}_{i=1}^n} \left\{ L(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \mathbb{E}_{\xi\sim\pi}\left[\min_{i\in[n]} f_\xi(\theta_i)\right]\right\}. \tag{2}$$

Let $\boldsymbol{\theta}^* = \{\theta_i^*\}_{i=1}^n$ the optimal solution and $[n]$ denotes $\{1, 2, \cdots, n\}$. We assign each θ_i^* an important weight via

$$\nu_i^* = \mathbb{E}_{\xi\sim\pi}[i \in \arg\min_{j\in[n]} f_\xi(\theta_j^*)]. \tag{3}$$

Then we approximate ρ^* by the weighted empirical measure $\hat{\rho}_{\boldsymbol{\theta}^*,\boldsymbol{\nu}^*} \propto \sum_{i=1}^n \nu_i^* \delta_{\theta_i^*}$. Note that objective in (2) reduces to the K-means objective function if $f_\xi(\theta) = \|\theta - \xi\|^2$.

In practice, we can optimize θ_i using stochastic gradient descent, and update ν_i recursively. See Algorithm 1 for details. At each iteration, we draw a (set of) $\xi \sim \pi$, find the θ_i that attains $i = \arg\min_j f_\xi(\theta_j)$ and update θ_i with gradient descent as displayed in line 4, Algorithm 1. The frequency ν_i is also updated accordingly when a θ_i is updated, see line 5 in Algorithm 1.

Replacement During optimization, it is possible that some θ_i are rarely selected and hence its weight ν_i becomes small and causes a degeneration problem (i.e., nearly zero important weight). To address this problem, we monitor the effective sample size of the points, defined as $n_{\textit{eff}} = (\sum_i \nu_i)^2/(\sum_i \nu_i^2)$. If the effective sample size becomes smaller than a threshold, we re-sample a set of new particles θ_i with replacement from $\hat{\rho}_{\theta,\nu} \propto \sum_{i=1}^n \nu_i \delta_{\theta_i}$ and repeat the updates. In practice, we break the ties randomly in argmin to avoid the case when two θ_i remains to be identical throughout the algorithm.

Algorithm 1 Main Algorithm: Argmax Centroids for Approximating ρ^*

Input: $\{\theta_i\}_{i=1}^n$ denotes the learnable centroids, ν_i denotes the updated frequency of θ_i, ϵ denotes the step size for gradient descent, $\alpha \in [0, 1]$ denotes a parameter to control the frequency update and η is a threshold to control the replacement.
while not convergence **do**
  Find $i_* = \arg\min_i f_\xi(\theta_i)$ with $\xi \sim \pi$.
  Update $\theta_{i_*} \leftarrow \theta_{i_*} - \epsilon \nabla_{\theta_{i_*}} f_\xi(\theta_{i_*})$, $\nu_{i_*} \leftarrow \alpha \nu_{i_*} + 1$, and $\nu_i = \alpha \nu_i$ for $i \neq i_*$.
  If $n_{\mathit{eff}} \leq \eta$, resample $\{\theta_i\}_{i=1}^n$ from $\hat{\rho}_{\theta,\nu} \propto \sum_{i=1}^n \nu_i \delta_{\theta_i}$.
end while

Theoretical Analysis We show that our method can be viewed as approximately minimizing the Wasserstein distance between the empirical measure of the centroids and the true distribution ρ^*. Let $\hat{\rho}_{\boldsymbol{\theta},\boldsymbol{\nu}} = \sum_{i=1}^n \nu_i \delta_{\theta_i}$ be the empirical measure on Θ with importance weights $\boldsymbol{\nu} = \{\nu_i\}_{i=1}^n$. Ideally, we would like to choose the centroids $\boldsymbol{\theta}$ and their weights $\boldsymbol{\nu}$ to solve

$$\min_{\boldsymbol{\theta},\, \boldsymbol{\nu}} W_p(\hat{\rho}_{\boldsymbol{\theta},\boldsymbol{\nu}},\ \rho^*), \quad s.t. \quad \boldsymbol{\nu} \in \mathcal{V}, \quad \boldsymbol{\theta} \in \Theta^n,$$

where W_p is the p-Wasserstein distance for $p > 0$ and $\mathcal{V} := \{\boldsymbol{\nu}\colon\ \sum_{i=1}^n \nu_i = 1,\ \nu_i \geq 0\}$ is a probability simplex. In the following, we show that our method in (2) is equivalent to optimizing a bound of the Wasserstein distance.

Assumption 2.1. *Let θ_ξ be the minimum of $f_\xi(\theta)$ as in (1). Assume there exists $h_1, h_2, p_1, p_2 \in (0, \infty)$, such that for any $\theta \in \Theta$ and $\xi \in \Xi$,*

$$h_1 \|\theta - \theta_\xi\|^{p_1} \leq f_\xi(\theta) - f_\xi(\theta_\xi) \leq h_2 \|\theta - \theta_\xi\|^{p_2}.$$

This assumption is highly general and includes both convex and non-convex functions. In particular, it holds with $p_1 = p_2 = 2$ if f is strongly convex w.r.t. θ and $0 < h_1/2 \leq \lambda_{min}(f''_\xi(\theta)) \leq \lambda_{max}(f''_\xi(\theta)) \leq h_2/2 < \infty$ for $\forall \theta$ and ξ, where λ_{min} and λ_{max} are the minimum and maximum eigenvalues, and with $p_2 = 1$ if f is Lipschitz w.r.t. θ with $\|f\|_{Lip} = h_2$.

Theorem 2.2. *Assume Assumption 2.1 holds. Define $L^* = \mathbb{E}_{\xi\sim\pi}[f_\xi(\theta_\xi)]$ where θ_ξ is the minimum point of $f_\xi(\theta)$.*

1) For any $\boldsymbol{\theta}$, we have

$$h_1 \Delta_{p_1}(\boldsymbol{\theta}) \leq L(\boldsymbol{\theta}) - L^* \leq h_2 \Delta_{p_2}(\boldsymbol{\theta}), \tag{4}$$

where $\Delta_{p_i}(\boldsymbol{\theta}) = \min_{\boldsymbol{\nu}\in\mathcal{V}} W_{p_i}(\rho_{\boldsymbol{\theta},\boldsymbol{\nu}},\ \rho^*)^{p_i}$.

2) For the optimal $\boldsymbol{\theta}^*$ and $\boldsymbol{\nu}^*$ in (2)-(3), we have

$$W_{p_1}(\rho_{\boldsymbol{\theta}^*,\boldsymbol{\nu}^*},\ \rho^*)^{p_1} \leq \frac{h_2}{h_1} \inf_{\boldsymbol{\theta}\in\Theta,\boldsymbol{\nu}\in\mathcal{V}} W_{p_2}(\rho_{\boldsymbol{\theta},\boldsymbol{\nu}},\ \rho^*)^{p_2}. \tag{5}$$

Proof. Note that $\nabla_x f_\xi(\theta_\xi) = 0$. By Taylor expansion, we have $L(\boldsymbol{\theta}) - L^* = \mathbb{E}_{\xi\sim\pi}[\min_i f_\xi(\theta_i) - f_\xi(\theta_\xi)]$. By the assumption, we have $h_1 \mathbb{E}[\min_i \|\theta_i - \theta_\xi\|^{p_1}] \leq L(\boldsymbol{\theta}) - L^* \leq h_2 \mathbb{E}[\min_i \|\theta_i - \theta_\xi\|^{p_2}]$. Then the bound in (4) follows the fact that $\mathbb{E}[\min_i \|\theta_i - \theta_\xi\|^p] = \min_{\boldsymbol{\nu}\in\mathcal{V}} W_p(\hat{\rho}_{\boldsymbol{\theta},\boldsymbol{\nu}},\ \ \rho^*)^p$, where the optimum $\boldsymbol{\nu}$ is achieved by $\nu_j \propto \mathbb{E}_{\xi\sim\pi}[j = \arg\min_i \|\theta_i - \theta_\xi\|]$. Therefore, for any $\boldsymbol{\theta} \in \Theta^n$, we have $L(\boldsymbol{\theta}^*) \leq L(\boldsymbol{\theta})$, and hence

$$\Delta_{p_1}(\boldsymbol{\theta}^*) \leq \frac{1}{h_1}(L(\boldsymbol{\theta}^*) - L^*) \leq \frac{1}{h_1}(L(\boldsymbol{\theta}) - L^*) \leq \frac{h_2}{h_1}\Delta_{p_2}(\boldsymbol{\theta}),$$

which is equivalent to (5). □

3 Applications and Related Works

Approximating argmax distribution can be found as a key component of many machine learning methods. This includes methods like bootstrap, Thompson sampling, and random MAP, in which the argmax distribution plays a key role by design, and as well as novel applications from meta learning and multi-task learning, which we study extensively in the experiments in Section 4.

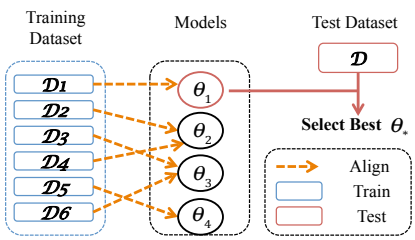


Figure 1: An illustration of our selection process for both training and evaluation.

Learning with Ensembles of Datasets Argmax distributions find a natural application in meta learning and multi-task/multi-domain learning, in which we need learn from an ensemble or a distribution of datasets. Denote by D a dataset on which we want to learn a model parameter θ with a loss function $\ell(\theta, D)$. Instead of having a single dataset, we consider the case when we have a large number of datasets $\{D_i\}$ which we assume is drawn from a distribution $\mathcal{P}$.

We aim to find a set of models $\{\theta_i\}_{i=1}^n$ where exists at least one good model for every dataset D with small n. Denote by $\theta_D = \arg\min_\theta \ell(\theta, D)$ and ρ^* the distribution of θ_D as $D \sim \mathcal{P}$. Then the problem can framed as finding $\{\theta_i\}_{i=1}^n$ to approximate ρ^*, and hence can be framed into

$$\min_{\{\theta_i\}_{i=1}^n} \left\{ \mathbb{E}_{D\sim\mathcal{P}} \left[\min_{i\in[n]} \ell(\theta_i, D) \right] \right\}. \tag{6}$$

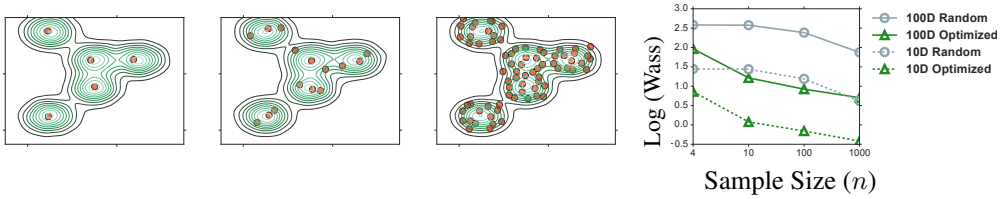
In the test phase, given a new coming dataset D, which we assume includes a validation set $\tilde{D}$ with true labels, we select the best model θ_{i_*} among the centroids that minimizes the loss $i_* = \arg\min_{i\in[n]} \ell(\theta_i, \tilde{D})$. The pipeline is shown in Figure 1. The idea here is to prepare a pool of candidate models to best cover all different scenarios (according to probability $\theta_D \sim \rho^*$), so that we can select the best model for any new random task drawn from ρ^*. This is different from typical ensemble learning which averages the output of multiple models during evaluation; instead, we use the best single model θ_{i_*} selected from the pool during evaluation. The objective ℓ and how to optimize ℓ is problem-dependent. For example, in meta learning, ℓ is updated with gradient descent and meta gradient. The details about the formulation of ℓ and its update rule is shown in Section 4 for different tasks. The method in (6) can also be used in boostrapping, in which we are given a dataset D_0 of true interest and we draw random datasets $D \sim \mathcal{P}$ by resampling from D_0 with replacement. Using the argmax centroids can yield better approximation with small n than naive random sampling that calculates θ_D from randomly drawn $D \sim \mathcal{P}$. The theoretical properties and efficient algorithms specified to bootstrap desires an independent treatment, which we explore separately in another work. Another related method is multiple choice learning (MCL) (Guzman-Rivera et al., 2012; Yu et al., 2018; Lee et al., 2016), which is an ensemble learning method using a similar objective function as (6) but with very different motivation and settings. MCL can be viewed as minimizing (6), but assuming each D to be a single data example rather than a dataset, with the goal of learning a set of models and ensemble them for a single dataset. During evaluation, the outputs of the multiple models from MCL are averaged to give the final output, rather than selecting the best one as we suggest above based on the perspective of approximating ρ^* .

Multi-task Learning Jointly learning models from multiple tasks has attracted long-term attention in machine learning community. In multi-task learning (MTL), there are multiple learning tasks and most of them (or all of them) are assumed to be related to each other. MTL algorithms can be grouped into two main approaches: the feature learning methods and task relation learning methods. Since tasks are related, it is intuitive to assume that different tasks share some common feature representation, and thus feature learning approaches focus on learning the common representation with regularizations (Yang et al., 2009), structure designing (Caruana, 1997), etc. The other approach focuses on learning task relations. The task relations are learnt from data automatically with a given prior (Yu et al., 2005; Xue et al., 2007; Zhang & Yeung, 2012), matrix decomposition (Chen et al., 2012) or clustering (Thrun & O'Sullivan, 1996). Recently, meta learning (Finn et al., 2017), lifelong learning (Thrun, 1998) and continual learning (Zenke et al., 2017) are proposed to solve MTL under different conditions.

Our method is similar to learning task relation with clustering. Different from previous works which cluster the tasks with latent representation (Zhou & Zhao, 2015), learnable relation matrix (Zhou et al., 2011; Kang et al., 2011) or side information (Caruana, 1997), we automatically assign a model to a certain domain/task based on its loss.

4 Experiments

In our experiments, we start with a toy example on which we show that our algorithm approximates the argmax distribution better than simple Monte Carlo estimation (Section 4.1). We then apply our method to standard few-shot learning benchmarks, in which parameters are learnt on thousands of domains (Section 4.2). We further apply our method to personalized dialogue systems (Peng et al., 2020) (Section 4.3) and multiple target domain adaptation (Yu et al., 2018) (Section 4.4) to verify whether our proposed method can be applied to different real-world applications. In all experiments, we set the replacement controller $\eta = 1.2$ and $\alpha = 0.5$ for Algorithm 1.



(a) (b) (c) (d)

Figure 2: 2D toy example. Figure (a), (b) and (c) show the estimated centroids $\{\theta_i\}_{i=1}^n$ (the red dots) and contour (green) of the true distribution ρ^*. Figure(d) shows The 2-Wasserstein distance between the empirical measure $\hat{\rho}_{\boldsymbol{\theta}^*,\boldsymbol{\nu}^*}$ of the centroids and the true distribution ρ^* w.r.t. the sample size n, when we vary the dimension in $d \in \{10, 100\}$. 'Log (Wass)' denotes log-scale Wasserstein distance.

4.1 Toy Examples

We start with a toy example to verify that our method can provide a better approximation of ρ^* than Monte Carlo estimation with a same number of samples. We set $f_\xi(\theta) = \|\theta - \xi\|^2$, so that $\theta_\xi = \xi$ and $\rho^* = \pi$. We set π to be d-dimension Gaussian mixture model with five modes. Note that our method is equivalent to a gradient descent variant of K-means in this case. Figure 2 demonstrates the two-dimension case, from which we can see that argmax centroids captures the different modes in the distribution and are aligned optimally to get better approximation then random sampling.

Figure 2(d) shows the 2-Wasserstein distance from the empirical measure $\hat{\rho}_{\boldsymbol{\theta}^*,\boldsymbol{\nu}^*}$ of the centroids and the true distribution ρ^* (calculated by drawing a large sample from ρ^*), when the dimension d of θ varies. We can see that for both 100-dimension and 10-dimension cases, our method yields lower Wasserstein distance than random sampling for different n.

Method	Transductive	Backbone	1-shot (Acc)	5-shot (Acc)
MAML (Finn et al., 2017)	✓	Conv4	48.7±1.8	64.6±1.2
(Tian et al., 2020)	✓	ResNet12	64.8±0.6	82.1±0.4
Baseline (Dhillon et al., 2020)	×	WRN-28-10	68.1±0.7	80.4±0.5
SIB (Hu et al., 2020)	✓	WRN-28-10	70.0±0.6	80.0±0.3
+ Ours	✓	WRN-28-10	**71.2±0.6**	**81.1±0.4**
IFSL (Yue et al., 2020)	✓	WRN-28-10	73.2±0.3	82.9±0.4
+ Ours	✓	WRN-28-10	**73.6±0.4**	**83.3±0.3**

Table 1: Few-shot learning results averaged over 3 trials on Mini-ImageNet.

4.2 Few-Shot Supervised Learning

We now apply our method to few-shot image classification using the formulation in (6), in combination with model agnostic meta learning (MAML) (Finn et al., 2017).

To introduce the setting, let $D = \{D_j^{Train}, D_j^{test}\}$ be a random dataset (or task) drawn $\mathcal{P}$, which includes a training and test set. Let $\theta' = \mathcal{A}(\theta, D^{train})$ be the result of training the parameter on D^{train} starting from θ, where $\mathcal{A}$ denotes the training algorithm, which is typically one step or few steps of stochastic gradient descent with training set. In MAML, we optimize θ so that the peformance of

Method	Transductive	Backbone	1-shot (Acc)	5-shot (Acc)
MAML (Finn et al., 2017)	✓	Conv4	-	-
(Tian et al., 2020)	✓	ResNet12	71.5±0.2	86.0±0.5
Baseline (Dhillon et al., 2020)	×	WRN-28-10	72.9±0.1	86.2±0.5
SIB (Hu et al., 2020)	✓	WRN-28-10	81.2±0.6	87.1±0.4
+ Ours	✓	WRN-28-10	**81.8±0.6**	**87.6±0.4**
IFSL (Yue et al., 2020)	✓	WRN-28-10	82.4±0.5	88.3±0.5
+ Ours	✓	WRN-28-10	**82.7±0.5**	**88.5±0.5**

Table 2: Few-shot Learning results averaged over 3 trials on TieredImageNet.

$\theta' = \mathcal{A}(\theta, D^{train})$ is maximized on D^{test}. The MAML loss is $\ell(\theta, D) = L(\mathcal{A}(\theta, D^{train}), D^{test})$, with the hope to rapidly solve a new coming task after learning several other similar tasks.

Standard MAML learns a single model θ, which may not work well for all the tasks D when the discrepancy between tasks is large. By plugging the MAML loss into (6), we obtain a new generalization of MAML with which we can prepare a pool of candidate models $\{\theta_i\}_{i=1}^n$ (the argmax centroids), out of which we can select the best one during testing. Note that this is different from the existing ensemble MAML methods such as (e.g., Lee et al., 2019; Yoon et al., 2018), which averages (instead of selecting the best among) the outputs of multiple models.

Baselines We compare and implement our method based on the recent state-of-the-art few-shot learning methods (e.g Hu et al., 2020; Yue et al., 2020; Wang et al., 2019). These methods train a network consisting of a backbone network shared by all the datasets (tasks), which is fed into classification heads specialized to individual datasets. During training, we first train a backbone network on *all* the training datasets in the training tasks using typical method, and then use meta-learning style training to tune the classification heads while keep the backbone frozen.

We implement our proposed argmax centroids upon two recent proposed few-learning methods, SIB and IFSL. These recently-proposed methods achieve state-of-the-art results on supervised few-shot image classification benchmarks [1]. SIB (Hu et al., 2020) proposes to learn synthetic gradient for new coming tasks, and IFSL (Yue et al., 2020) finds out that pre-trained knowledge is essentially a confounder that causes spurious correlations between the sample features and class labels in support set. They further propose regularization terms to eliminate the mismatch.

Method	MAML (Finn et al., 2017)	Ensemble (Yoon et al., 2018)	Bayesian (Yoon et al., 2018)	Ours
ResNet-18	56.6±.3	56.7±.3	57.0±.4	**57.4±.2**
WRN-28-10	57.4±.4	57.4±.5	57.7±.4	**58.1±.4**

Table 3: Comparison to ensemble MAML methods on Mini-ImageNet. We demonstrate the accuracy for one-shot classification with ResNet-10 and WRN-28-10. We set the number of heads to 16 while sharing the backbone parameters. 'Ensemble' and 'Bayesian' denotes ensemble MAML and Bayesian MAML, respectively.

All these baselines and other recent works (e.g. Hu et al., 2020; Yue et al., 2020; Wang et al., 2019; Dhillon et al., 2020) focus on learning a better single model by introducing strong regularization methods. For the baselines we focus on, a clear disadvantage of SIB and IFSL is that they share one single θ for all the datasets and may not work well once the discrepancy between task is large. By applying our method, we train a fixed number of classification heads, which is treated as the argmax centroids. This allows us to pick the best head θ and further finetune it on the training set for each test task during testing. We simply combine these baseline approaches with our algorithm to obtain further improvements. To save computation cost in practice, we evaluate $\mathcal{A}(\theta, D^{Train})$ by one-step gradient descent and then select θ_i with lowest loss among $\boldsymbol{\theta}$.

Datasets Standard benchmarks of few-shot classification are chosen for experiments. We evaluate all the baselines and our algorithms on two subset of ImageNet, Mini-ImageNet and TieredImageNet (Sun et al., 2019). Mini-ImageNet contains 64 classes for training, 16 for validation and 20 for test. Training/validation/test tasks are sampled from the 64/16/20 classes. TieredImageNet (Sun

[1] https://few-shot.yyliu.net

et al., 2019) is much larger compared to miniImageNet and other benchmarks, with 608 classes and 1300 samples per class. These classes are partitioned into 351/97/160 disjoint sets for train/val/test to achieve larger domain difference between training and testing During training, the images in Mini-ImageNet and tieredImageNet are resized to 80×80 and 84×84, respectively. Each sampled dataset D is a 5-class classification, with 1-shot/5-shot images per class for training and 15 images per class for test. When evaluating the trained models, 2000 test tasks are sampled from the test set, and the accuracy on D^{test} of all the test tasks are reported for comparison.

Implementation Details For each baseline method, we create a small number of additional parameters. During the second training stage, given a backbone, e.g., Wide-ResNet-28-10 (Zagoruyko & Komodakis, 2016), we use additional parameters $\{\theta_i\}_i^n$ to re-weight the final-layer features. In the experiments, for few-shot learning based on SIB and IFSL, we set $n = 16$. We only introduce a small number of additional parameters (0.01M parameters) compared to the model size (22M parameters).

For both the training and testing, we keep all the settings (e.g. optimizer, learning rate, number of tasks) the same for each baseline method. We first train the feature extractors using the standard cross-entropy loss on the base classes (i.e., the classes in the training set in all training tasks). Secondly, training the heads following the standard meta learning pipeline. Finally, we test the trained models on sampled test tasks. For each test task, we do few gradient descent on the training set. The number of iterations follows the baseline's.

Results Table 3 shows the comparison between our proposed method and standard ensemble MAML methods.

We implement our method upon several recently-proposed strong baselines. As demonstrated in Table 1 and 2, our algorithm consistently improves all the baseline methods in all settings, which suggests that our proposed algorithm is a general method for few-shot image classification. For example, on Mini-ImageNet, we improve the accuracy of SIB by a large margin for both 1-shot (70.0% to 71.2%) and 5-shot (80.0% to 81.1%) classification. For IFSL, we can also boost the performance under most of the settings.

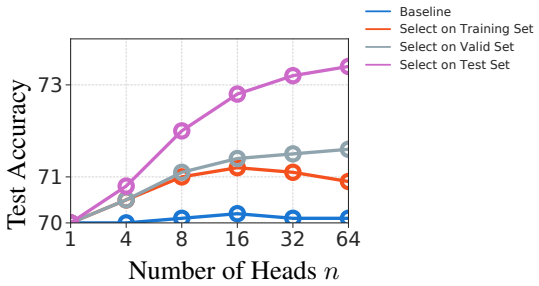


Figure 3: Training n different heads for the few-shot learning task MiniImageNet with SIB. 'Baseline' denotes training one model , creating $\{\theta_i\}_{i=1}^n$ by dropout and selecting $\{\theta_i\}_{i=1}^n$ during evaluation.

Comparison to Ensemble Methods Here, we follow the setting in and ensemble MAML (Yoon et al., 2018) where the backbone is not pretrained on all the training images and is jointly training during the meta learning step. We compare to two popular ensemble baselines, ensemble MAML and Bayesian MAML (Yoon et al., 2018). Ensemble MAML trains n models independently and take an average of the outputs of all the models during evaluation. Bayesian MAML (Yoon et al., 2018) uses SVGD (Liu & Wang, 2016) to jointly update n models during training and ensemble them during evaluation. Instead of choosing one best model for a test domain like our method, the ensemble methods average the outputs of all the models. From Table 3, we can see that our method is better than both ensemble approaches on one-shot classification on Mini-ImageNet with different neural architectures. On ResNet-18 and WRN-28-10, our method achieves better results than ensemble methods.

Ablation Study In Figure 3, we change n and make selection based on training/test/validation loss to have a deeper understand of our algorithm. We notice that, if we do not have a separate validation set, we may need to select the best model on the training set during evaluation, which cause overfitting. We investigate this issue in Figure 3, in which we show the results when we select the best models using training, validation and, test set, respectively. In order to do the selection on the validation set, we additionally draw 5 images per class for each test task. The result shows that, in our case, both selecting on training and validation set is worse than the "oracle" way (i.e., selecting on the test set and evaluate on the test set). When the number of head increasing, the performance of selecting on training set drops and overfitting happens.

4.3 Personalized Dialogue System

Problem Set We consider learning personalized chit-chat dialogue agents for making chat-bots more consistent for each user (e.g. Zhang et al., 2018; Madotto et al., 2019b; Tian et al., 2020). Previous approaches usually learning persona similarity from persona descriptions (Zhang et al., 2018). Different from these methods, current personalized dialogue systems focus on learning to quickly adapt to new personas by using few samples. In this problem, each dataset $D = (D^{Train}, D^{Test}) \sim \mathcal{P}$ contains several dialogues of a user. A dialogue contains a set of utterances, and the objective is to predict the next utterance given previous utterances. We use the MAML objective for training.

Method	PPL ↓	BLEU ↑
PAML (Madotto et al., 2019a)	46.35	0.77
Ours	**43.94**	**0.91**

Table 4: Results of automatic evaluation on personalized dialogue generation. We report case-insensitive BLEU and PPL.

Baselines We use a recently-proposed meta learning dialogue system (Madotto et al., 2019b) as the baseline. We measure the difference between the ground-truth output and model output to evaluate the model performance by PPL (Perplexity) and case-insensitive BLEU score (Papineni et al., 2002).

Implementation Details Before training, the datasets sampled from $\mathcal{P}$ is split into meta-training, meta-validation and meta-test dataset. The meta-learner learns how to learn by training and evaluating on the meta-training set.

Meta-learning hyperparameters are tuned on the meta-validation set. The meta-test set measures generalization on new, unseen tasks. We use the BERT-base (Devlin et al., 2019) model as the backbone and tune the parameters in the backbone together with the final softmax layer. For meta training, we use Adam (Kingma & Ba, 2014) with learning rate 10^{-3}, 10^{-2} for inner and outer loop training, respectively. During evaluation, for all the models, we used beam search with beam size 4 and length penalty 1.2. These hyper-parameters are selected based on the performance on the meta-validation dataset.

For our method, we create $n = 16$ softmax head (i.e. a linear transform followed by a softmax layer), share the backbone parameters and select one head for each persona with the lowest loss on the test set for this persona during training, and select the softmax head with the lowest loss on training set during evaluation.

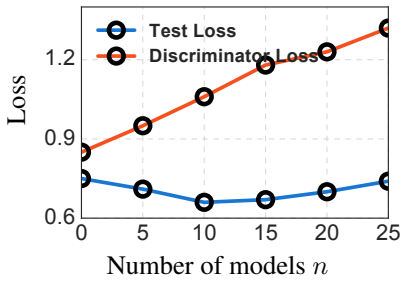


Figure 4: Illustration of the per domain test loss, and per domain negative discriminator loss. For the multi-domain adaptation task, when the number n of models increases, the negative discriminator loss always increases while the test loss first decreases and then increases.

Result Table 4 shows that, our proposed algorithm beats the meta-learning baseline on both the PPL and BLEU measure. We improves the BLEU from 0.77 to 0.91 and PPL from 46.35 to 43.94, respectively.

4.4 Multi-Target Domain Adaptation

Domain adaptation is a powerful approach for learning under distributional shift between training and test data (e.g. Wu et al., 2018; Tzeng et al., 2017). Here, we show that our proposed algorithm can also be applied to multi-target domain adaptation (Yu et al., 2018) using the dataset proposed in Peng et al. (2019). For multi-target domain adaptation, one source domain and m multi-target domain is given. In this experiment, we use the classical domain adaptation method proposed by Tzeng et al. (2017), in which an adversarial net is used to learn the adaptation from the latent space in the source domain.

We first pretrain a CNN using labeled image examples from the source domain and denotes the backbone as g. Next, for target domain, we perform adversarial adaptation by learning a map ϕ after g such that a discriminator that sees source examples $g(\cdot)$ and target examples $\phi(g(\cdot))$ cannot reliably predict their domain label. During evaluation, target images are mapped with backbone g followed by the learned map to the shared feature space and classified by the final-layer classifier trained on the source domain.

Our Method After pretraining a backbone on the source domain, We consider to use n adversarial nets $\{\theta_i\}_{i=1}^n$ as domain adaptors to match the representation of the source domain and target domains. Each $\theta_i = (\phi_i, \gamma_i)$ contains two part of parameters where ϕ_i is a feature map and γ_i is a discriminator. ϕ_i is a MLP whose input and output has the same dimension and γ_i maps the input to a one-dimension scalar followed by a sigmoid function.

Denote $z = g(x)$ the features extracted by the backbone and S the source domain, $\ell(\theta_i, D)$ equals to

$$-\mathbb{E}_{z=g(x),x\sim D}\Big\{\log\gamma_i(\phi_i(z)) + \log(1-\gamma_i(\phi_i(z))\Big\} - \mathbb{E}_{z=g(x),x\sim S}\Big\{\log\gamma_i(z) + \log(1-\gamma_i(z))\Big\},$$

is an adversarial loss which forces the discriminator to make uniform prediction.

Baselines We compare our method with three simple baselines. One is using a shared adversarial domain adaptors for all the target domains. Another is using n different domain adaptors for m target domains. The third baseline is randomly assigning domains with models for each iteration during training and average all the outputs for all models during evaluation.

Experiment Settings We use the DomainNet dataset proposed by Peng et al. (2019). DomainNet contains 6 styles, 345 classes and 569,010 images. We create a subset of DomainNet to do experiments. We sample 25 classes in the same style and sample 300 images for each class. For each sample subset (domain), we use 5 different data shift (e.g. brightness, fog, snow, MotionBlur, Gaussian noise) (Hendrycks et al., 2019) to create new domains. Finally, we obtain 30 different domains. We randomly select one domain data as source domain and the rest as target domains during experiments. For the neural architecture, we use a standard ResNet-18 (He et al., 2016) to train on the source domain, a three-layer MLP as γ_i and another three-layer MLP as ϕ_i. During training and testing, we resize the image into 64×64 size. During training, we use standard data augmentation (mirror and flip).

Method	1 Adaptor	m Adaptor	Ensemble	Ours
Acc ↑	39.5±0.4	41.8±0.6	41.3±0.7	**44.3±0.6**

Table 5: Results on multi-target domain adaptation. The reported accuracy are averaged over 5 trials. "Ensemble" denotes randomly assigning models with domains during training and ensembling all the models during evaluation.

Results As demonstrated in Table 5, compared to sharing one domain adaptor for all the domains, using domain-specific domain adaptor improves the accuracy from 39.5% to 41.8%. With our proposed method, we can further improve the 41.8% accuracy to 44.3%. In Table 5, we also demonstrate that, randomly aligning different domains hurts the performance. It indicates that, in this setting our method is much better than ensemble multiple models. We further set up an ablation study on the number of heads. As shown in Figure 4, similar to the results in Figure 3, increasing the number of heads can always improve the training loss (i.e., discriminator loss in (**??**)). Here, increasing n makes the negative discriminator loss higher indicates that it mixes the two domains successfully. However, the test loss increases once n is too large. It demonstrates that our argmax centroids method implicitly serves as a regularization by sharing a same linear map ϕ for multiple similar domains. Sharing the same linear map increases the number of training data for each θ, and therefore avoids overfitting.

5 Conclusion

In this work, we propose a method to estimate the argmax distribution. It can be an alternative for bootstrap and other related methods, and we further apply it to many deep learning applications. Aligning similar domains into a same model, the proposed algorithm boost the SOTA performance for several few-shot classification and meta learning tasks. For the next step, we will consider the connection with diversifying models and other topics. We will also apply our method to more meaningful and challenging tasks, e.g. lifelong learning, continual learning.

Limitations and negative societal impacts Our algorithm slightly increase the time cost and memory cost in the experiments. Once we open-source our model checkpoints and code, we will have no control of the code and model checkpoints.

References

Caruana, Rich. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

Chen, Jianhui, Liu, Ji, and Ye, Jieping. Learning incoherent sparse and low-rank patterns from multiple tasks. *ACM Transactions on Knowledge Discovery from Data*, 5(4):1–31, 2012.

Devlin, Jacob, Chang, Ming-Wei, Lee, Kenton, and Toutanova, Kristina. Bert: Pre-training of deep bidirectional transformers for language understanding. *North American Chapter of the Association for Computational Linguistics*, 2019.

Dhillon, Guneet S, Chaudhari, Pratik, Ravichandran, Avinash, and Soatto, Stefano. A baseline for few-shot image classification. *International Conference on Learning Representations*, 2020.

Efron, Bradley and Tibshirani, Robert J. *An introduction to the bootstrap*. CRC press, 1994.

Finn, Chelsea, Abbeel, Pieter, and Levine, Sergey. Model-agnostic meta-learning for fast adaptation of deep networks. *International Conference on Machine Learning*, 2017.

Guzman-Rivera, Abner, Batra, Dhruv, and Kohli, Pushmeet. Multiple choice learning: Learning to produce multiple structured outputs. In *Advances in neural information processing systems*, volume 1, pp. 3. Citeseer, 2012.

Hazan, Tamir, Maji, Subhransu, Keshet, Joseph, and Jaakkola, Tommi. Learning efficient random maximum a-posteriori predictors with non-decomposable loss functions. Advances in neural information processing systems, 2013.

He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, and Sun, Jian. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Hendrycks, Dan, Mu, Norman, Cubuk, Ekin D, Zoph, Barret, Gilmer, Justin, and Lakshminarayanan, Balaji. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019.

Hu, Shell Xu, Moreno, Pablo G, Xiao, Yang, Shen, Xi, Obozinski, Guillaume, Lawrence, Neil D, and Damianou, Andreas. Empirical bayes transductive meta-learning with synthetic gradients. *International Conference on Learning Representations*, 2020.

Kang, Zhuoliang, Grauman, Kristen, and Sha, Fei. Learning with whom to share in multi-task feature learning. In *International Conference on Machine Learning*, 2011.

Kingma, Diederik P and Ba, Jimmy. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2014.

Kuo, Frances Y and Sloan, Ian H. Lifting the curse of dimensionality. *Notices of the AMS*, 52(11): 1320–1328, 2005.

Lee, Hae Beom, Nam, Taewook, Yang, Eunho, and Hwang, Sung Ju. Meta dropout: Learning to perturb latent features for generalization. In *International Conference on Learning Representations*, 2019.

Lee, Stefan, Purushwalkam, Senthil, Cogswell, Michael, Ranjan, Viresh, Crandall, David, and Batra, Dhruv. Stochastic multiple choice learning for training diverse deep ensembles. *Advances in neural information processing systems*, 2016.

Liu, Qiang and Wang, Dilin. Stein variational gradient descent: A general purpose bayesian inference algorithm. *Advances in neural information processing systems*, 2016.

Madotto, Andrea, Lin, Zhaojiang, Wu, Chien-Sheng, and Fung, Pascale. Personalizing dialogue agents via meta-learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5454–5459, 2019a.

Madotto, Andrea, Lin, Zhaojiang, Wu, Chien-Sheng, and Fung, Pascale. Personalizing dialogue agents via meta-learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5454–5459, 2019b.

Papineni, Kishore, Roukos, Salim, Ward, Todd, and Zhu, Wei-Jing. Bleu: a method for automatic evaluation of machine translation. In *Annual Meeting of the Association for Computational Linguistics*, pp. 311–318, 2002.

Peng, Baolin, Zhu, Chenguang, Li, Chunyuan, Li, Xiujun, Li, Jinchao, Zeng, Michael, and Gao, Jianfeng. Few-shot natural language generation for task-oriented dialog. *Conference on Empirical Methods in Natural Language Processing*, 2020.

Peng, Xingchao, Bai, Qinxun, Xia, Xide, Huang, Zijun, Saenko, Kate, and Wang, Bo. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1406–1415, 2019.

Russo, Daniel, Van Roy, Benjamin, Kazerouni, Abbas, Osband, Ian, and Wen, Zheng. A tutorial on thompson sampling. *arXiv preprint arXiv:1707.02038*, 2017.

Sun, Qianru, Liu, Yaoyao, Chua, Tat-Seng, and Schiele, Bernt. Meta-transfer learning for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 403–412, 2019.

Thrun, Sebastian. Lifelong learning algorithms. In *Learning to learn*, pp. 181–209. Springer, 1998.

Thrun, Sebastian and O'Sullivan, Joseph. Discovering structure in multiple learning tasks: The tc algorithm. In *International Conference on Machine Learning*, volume 96, pp. 489–497, 1996.

Tian, Yonglong, Wang, Yue, Krishnan, Dilip, Tenenbaum, Joshua B, and Isola, Phillip. Rethinking few-shot image classification: a good embedding is all you need? *Proceedings of the European Conferenceon Computer Vision*, 2020.

Tzeng, Eric, Hoffman, Judy, Saenko, Kate, and Darrell, Trevor. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7167–7176, 2017.

Wang, Yan, Chao, Wei-Lun, Weinberger, Kilian Q, and van der Maaten, Laurens. Simpleshot: Revisiting nearest-neighbor classification for few-shot learning. *Advances in neural information processing systems*, 2019.

Wu, Zuxuan, Han, Xintong, Lin, Yen-Liang, Gokhan Uzunbas, Mustafa, Goldstein, Tom, Nam Lim, Ser, and Davis, Larry S. Dcan: Dual channel-wise alignment networks for unsupervised scene adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 518–534, 2018.

Xue, Ya, Liao, Xuejun, Carin, Lawrence, and Krishnapuram, Balaji. Multi-task learning for classification with dirichlet process priors. *Journal of Machine Learning Research*, 8(1), 2007.

Yang, Xiaolin, Kim, Seyoung, and Xing, Eric. Heterogeneous multitask learning with joint sparsity constraints. *Advances in neural information processing systems*, 22:2151–2159, 2009.

Yoon, Jaesik, Kim, Taesup, Dia, Ousmane, Kim, Sungwoong, Bengio, Yoshua, and Ahn, Sungjin. Bayesian model-agnostic meta-learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 7343–7353, 2018.

Yu, Huanhuan, Hu, Menglei, and Chen, Songcan. Multi-target unsupervised domain adaptation without exactly shared categories. *Advances in neural information processing systems*, 2018.

Yu, Kai, Tresp, Volker, and Schwaighofer, Anton. Learning gaussian processes from multiple tasks. In *International conference on Machine learning*, pp. 1012–1019, 2005.

Yue, Zhongqi, Zhang, Hanwang, Sun, Qianru, and Hua, Xian-Sheng. Interventional few-shot learning. *Advances in neural information processing systems*, 2020.

Zagoruyko, Sergey and Komodakis, Nikos. Wide residual networks. *BMVC*, 2016.

Zenke, Friedemann, Poole, Ben, and Ganguli, Surya. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pp. 3987–3995. PMLR, 2017.

Zhang, Saizheng, Dinan, Emily, Urbanek, Jack, Szlam, Arthur, Kiela, Douwe, and Weston, Jason. Personalizing dialogue agents: I have a dog, do you have pets too? *Association for Computational Linguistics*, 2018.

Zhang, Yu and Yeung, Dit-Yan. A convex formulation for learning task relationships in multi-task learning. *Association for Uncertainty in Artificial Intelligence*, 2012.

Zhou, Jiayu, Chen, Jianhui, and Ye, Jieping. Clustered multi-task learning via alternating structure optimization. *Advances in neural information processing systems*, 2011:702, 2011.

Zhou, Qiang and Zhao, Qi. Flexible clustered multi-task learning by learning representative tasks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):266–278, 2015.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [N/A]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]