Transductive Information Maximization
For Few-Shot Learning

Malik Boudiaf∗
ÉTS Montreal

Ziko Imtiaz Masud
ÉTS Montreal

Jérôme Rony
ÉTS Montreal

Jose Dolz
ÉTS Montreal

Pablo Piantanida
CentraleSupélec-CNRS
Université Paris-Saclay

Ismail Ben Ayed
ÉTS Montreal

Abstract
We introduce Transductive Information Maximization (TIM) for few-shot learning. Our method maximizes the mutual information between the query features and their label predictions for a given few-shot task, in conjunction with a supervision loss based on the support set. Furthermore, we propose a new alternating-direction solver for our mutual-information loss, which substantially speeds up transductive-inference convergence over gradient-based optimization, while yielding similar accuracy. TIM inference is modular: it can be used on top of any base-training feature extractor. Following standard transductive few-shot settings, our comprehensive experiments demonstrate that TIM outperforms state-of-the-art methods significantly across various datasets and networks, while used on top of a fixed feature extractor trained with simple cross-entropy on the base classes, without resorting to complex meta-learning schemes. It consistently brings between 2% and 5% improvement in accuracy over the best performing method, not only on all the well-established few-shot benchmarks but also on more challenging scenarios, with domain shifts and larger numbers of classes.

1 Introduction
Deep learning models have achieved unprecedented success, approaching human-level performances when trained on large-scale labeled data. However, the generalization of such models might be seriously challenged when dealing with new (unseen) classes, with only a few labeled instances per class. Humans, however, can learn new tasks rapidly from a handful of instances, by leveraging context and prior knowledge. The few-shot learning (FSL) paradigm attempts to bridge this gap, and has recently attracted substantial research interest, with a large body of very recent works, e.g., [29, 8, 45], among many others. In the few-shot setting, a model is first trained on labeled data with base classes. Then, model generalization is evaluated on few-shot tasks, composed of unlabeled samples from novel classes unseen during training (the query set), assuming only one or a few labeled samples (the support set) are given per novel class.

Most of the existing approaches within the FSL framework are based on the “learning to learn” paradigm or meta-learning [9, 48, 40, 22], where the training set is viewed as a series of balanced tasks (or episodes), so as to simulate test-time scenario. Popular works include prototypical networks [38], which describes each class with an embedding prototype and maximizes the log-probability of query samples via episodic training; matching network [45], which represents query predictions as

∗Corresponding author: malik.boudiaf.1@etsmtl.net

2Code publicly available at https://github.com/mboudiaf/TIM
linear combinations of support labels and employs episodic training along with memory architectures; MAML [9], a meta-learner, which trains a model to make it "easy" to fine-tune; and the LSTM meta-learner in [35], which suggests optimization as a model for few-shot learning. A large body of meta-learning works followed-up lately, to only cite a few [37, 33, 30, 40, 49].

1.1 Related work

Transductive inference: In a recent line of work, transductive inference has emerged as an appealing approach to tackling few-shot tasks [7, 14, 19, 28, 34, 32, 27, 51], showing performance improvements over inductive inference. In the transductive setting, the model classifies the unlabeled query examples of a single few-shot task at once, instead of one sample at a time as in inductive methods. These recent experimental observations in few-shot learning are consistent with established facts in classical transductive inference [44, 18, 6], which is well-known to outperform inductive methods on small training sets. While [32] used information of unlabeled query samples via batch normalization, the authors of [28] were the first to model explicitly transductive inference in few-shot learning. Inspired by popular label-propagation concepts [6], they built a meta-learning framework that learns to propagate labels from labeled to unlabeled instances via a graph. The meta-learning transductive method in [14] used attention mechanisms to propagate labels to unlabeled query samples. More closely related to our work, the recent transductive inference of Dhillion et al. [7] minimizes the entropy of the network softmax predictions at unlabeled query samples, reporting competitive few-shot performances, while using standard cross-entropy training on the base classes. The competitive performance of [7] is in line with several recent inductive baselines [5, 46, 41], which reported that standard cross-entropy training for the base classes matches or exceeds the performances of more sophisticated meta-learning procedures. Also, the performance of [7] is in line with established results in the context of semi-supervised learning, where entropy minimization is widely used [11, 31, 2]. It is worth noting that the inference runtimes of transductive methods are, typically, much higher than their inductive counterparts. For instance, the authors of [7] fine-tune all the parameters of a deep network during inference, which is several orders of magnitude slower than inductive methods such as ProtoNet [38]. Also, based on matrix inversion, the transductive inference in [28] has a complexity that is cubic in the number of query samples.

Info-max principle: While the semi-supervised and few-shot learning works in [11, 7] build upon Barlow’s principle of entropy minimization [1], our few-shot formulation is inspired by the general info-max principle enunciated by Linsker [25], which formally consists in maximizing the Mutual Information (MI) between the inputs and outputs of a system. In our case, the inputs are the query features and the outputs are their label predictions. The idea is also related to info-max in the context of clustering [21, 16, 17]. More generally, info-max principles, well-established in the field of communications, were recently used in several deep-learning problems, e.g., representation learning [13, 43], metric learning [3], or domain adaptation [24], among other problems.

1.2 Contributions

- We propose Transductive Information Maximization (TIM) for few-shot learning. Our method maximizes the MI between the query features and their label predictions for a few-shot task at inference, while minimizing the cross-entropy loss on the support set.
- We derive an alternating-direction solver for our loss, which substantially speeds up transductive inference over gradient-based optimization, while yielding competitive accuracy.
- Following standard transductive few-shot settings, our comprehensive evaluations show that TIM outperforms state-of-the-art methods substantially across various datasets and networks, while using a simple cross-entropy training on the base classes, without complex meta-learning schemes. It consistently brings between 2% and 5% of improvement in accuracy over the best performing method, not only on all the well-established few-shot benchmarks but also on more challenging, recently introduced scenarios, with domain shifts and larger numbers of ways. Interestingly, our MI loss includes a label-marginal regularizer, which has a significant effect: it brings substantial improvements in accuracy, while facilitating optimization, reducing transductive runtimes by orders of magnitude.

Transductive few-shot inference is not to be confused with semi-supervised few-shot learning [36, 23]. The latter uses extra unlabeled data during meta-training. Transductive inference has access to exactly the same training/testing data as its inductive counterpart.
2 Transductive Information Maximization

2.1 Few-shot setting

Assume we are given a labeled training set, $X_{\text{base}} := \{x_i, y_i\}_{i=1}^{N_{\text{base}}}$, where $x_i$ denotes raw features of sample $i$ and $y_i$ its associated one-hot encoded label. Such labeled set is often referred to as the meta-training or base dataset in the few-shot literature. Let $Y_{\text{base}}$ denote the set of classes for this base dataset. The few-shot scenario assumes that we are given a test dataset: $X_{\text{test}} := \{x_i, y_i\}_{i=1}^{N_{\text{test}}}$, with a completely new set of classes $Y_{\text{test}}$ such that $Y_{\text{base}} \cap Y_{\text{test}} = \emptyset$, from which we create randomly sampled few-shot tasks, each with a few labeled examples. Specifically, each $K$-way $N_S$-shot task involves sampling $N_S$ labeled examples from each of $K$ different classes, also chosen at random. Let $S$ denote the set of these labeled examples, referred to as the support set with size $|S| = N_S \cdot K$. Furthermore, each task has a query set denoted by $Q$ composed of $|Q| = N_Q \cdot K$ unlabeled (unseen) examples from each of the $K$ classes. With models trained on the base set, few-shot techniques use the labeled support sets to adapt to the tasks at hand, and are evaluated based on their performances on the unlabeled query sets.

2.2 Proposed formulation

We begin by introducing some basic notation and definitions before presenting our overall Transductive Information Maximization (TIM) loss and the different optimization strategies for tackling it. For a given few-shot task, with a support set $S$ and a query set $Q$, let $X$ denote the random variable associated with the raw features within $S \cup Q$, and let $Y \subseteq Y = \{1, \ldots, K\}$ be the random variable associated with the labels. Let $f_{\phi} : X \rightarrow \mathcal{Z} \subset \mathbb{R}^d$ denote the encoder (i.e., feature-extractor) function of a deep neural network, where $\phi$ denotes the trainable parameters, and $\mathcal{Z}$ stands for the set of embedded features. The encoder is first trained from the base training set $X_{\text{base}}$ using the standard cross-entropy loss, without any meta training or specific sampling schemes. Then, for each specific few-shot task, we propose to minimize a mutual-information loss defined over the query samples.

Formally, we define a soft-classifier, parametrized by weight matrix $W \in \mathbb{R}^{K \times d}$, whose posterior distribution over labels given features is $p_{ik} := \mathbb{P}(Y = k | X = x_i; W, \phi)$, and marginal distribution over query labels. $\hat{p}_k = \mathbb{P}(Y_Q = k; W, \phi)$, are given by:

$$p_{ik} \propto \exp \left( -\frac{\tau}{2} \| w_k - z_i \|^2 \right), \quad \text{and} \quad \hat{p}_k = \frac{1}{|Q|} \sum_{i \in Q} p_{ik} \quad (1)$$

where $W := [w_1, \ldots, w_K]$ denotes classifier weights, $z_i = \frac{f_{\phi}(x_i)}{\| f_{\phi}(x_i) \|_2}$ the L2-normalized embedded features, and $\tau$ is a temperature parameter.

Now, for each single few-shot task, we introduce our empirical weighted mutual information between the query samples and their latent labels, which integrates two terms: The first is an empirical (Monte-Carlo) estimate of the conditional entropy of labels given the query raw features, denoted $\hat{H}(Y_Q | X_Q)$, while the second is the empirical label-marginal entropy, $\hat{H}(Y_Q)$:

$$\hat{I}_\alpha(X_Q; Y_Q) := -\sum_{k=1}^{K} \hat{p}_k \log \hat{p}_k + \alpha \frac{1}{|Q|} \sum_{i \in Q} \sum_{k=1}^{K} p_{ik} \log(p_{ik}) \quad (2)$$

with $\alpha$ a non-negative hyper-parameter. Notice that setting $\alpha = 1$ recovers the standard mutual information. Setting $\alpha < 1$ allows us to down-weight the conditional entropy term, whose gradients may dominate the marginal entropy’s gradients as the predictions move towards the vertices of the simplex. The role of both terms in $\hat{I}_\alpha$ will be discussed after introducing our overall transductive inference loss in the following, by embedding supervision from the task’s support set.

We embed supervision information from support set $S$ by integrating a standard cross-entropy loss $\text{CE}$ with the information measure in Eq. (2), which enables us to formulate our Transductive Information

---

4In order to simplify our notations, we deliberately omit the dependence of posteriors $p_{ik}$ on the network parameters $(\phi, W)$. Also, $p_{ik}$ takes the form of softmax predictions, but we omit the normalization constants.
Maximization (TIM) loss as follows:

$$\min_{W} \lambda \cdot \text{CE} - \hat{H}(Y_Q|X_Q)$$

with

$$\text{CE} := \frac{1}{|S|} \sum_{i \in S} \sum_{k=1}^{K} y_{ik} \log(p_{ik}), \quad (3)$$

where \( \{y_{ik}\} \) denotes the \( k^{th} \) component of the one-hot encoded label \( y_k \) associated to the \( i \)-th support sample. Non-negative hyper-parameters \( \alpha \) and \( \lambda \) will be fixed to \( \alpha = \lambda = 0.1 \) in all our experiments. It is worth to discuss in more details the role (importance) of the mutual information terms in \( \hat{H}(Y_Q|X_Q) \):

- Conditional entropy \( \hat{H}(Y_Q|X_Q) \) aims at minimizing the uncertainty of the posteriors at unlabeled query samples, thereby encouraging the model to output confident predictions\(^3\). This entropy loss is widely used in the context of semi-supervised learning (SSL) \( [11, 31, 2] \), as it models effectively the cluster assumption: The classifier’s boundaries should not occur at dense regions of the unlabeled features \( [11] \). Recently, \( [7] \) introduced this term for few-shot learning, showing that entropy fine-tuning on query samples achieves competitive performances. In fact, if we remove the marginal entropy \( \hat{H}(Y_Q) \) in objective \( [5] \), our TIM objective reduces to the loss in \( [7] \). The conditional entropy \( \hat{H}(Y_Q|X_Q) \) is of paramount importance but its optimization requires special care, as its optima may easily lead to degenerate (non-suitable) solutions on the simplex vertices, mapping all samples to a single class. Such care may consist in using small learning rates and fine-tuning the whole network (which itself often contains several layers of regularization) as done in \( [7] \), both of which significantly slow down transductive inference.

- The label-marginal entropy regularizer \( \hat{H}(Y_Q) \) encourages the marginal distribution of labels to be uniform, thereby avoiding degenerate solutions obtained when solely minimizing conditional entropy. Hence, it is highly important as it removes the need for implicit regularization, as mentioned in the previous paragraph. In particular, high-accuracy results can be obtained even using higher learning rates and fine-tuning only a fraction of the network parameters (classifier weights \( W \) instead of the whole network), speeding up substantially transductive runtimes. As it will be observed from our experiments, this term brings substantial improvements in performances (e.g., up to 10% increase in accuracy over entropy fine-tuning on the standard few-shot benchmarks), while facilitating optimization, thereby reducing transductive runtimes by orders of magnitude.

### 2.3 Optimization

At this stage, we consider that the feature extractor has already been trained on base classes (using standard cross-entropy). We now propose two methods for minimizing our objective \( \hat{H}(Y_Q|X_Q) \) for each test task. The first one is based on standard Gradient Descent (GD). The second is a novel way of optimizing mutual information, and is inspired by the Alternating Direction Method of Multipliers (ADMM). For both methods:

- The pre-trained feature extractor \( f_\phi \) is kept fixed. Only the weights \( W \) are optimized for each task. Such a choice is discussed in details in subsection \( 3.4 \). Overall, and interestingly, we found that fine-tuning only classifier weights \( W \), while fixing feature-extractor parameters \( \phi \), yielded the best performances for our mutual-information loss.

- For each task, weights \( W \) are initialized as the class prototypes of the support set:

$$w_k^{(0)} = \frac{\sum_{i \in S} y_{ik} z_i}{\sum_{i \in S} y_{ik}}$$

**Gradient descent (TIM-GD):** A straightforward way to minimize our loss in Eq. \( (3) \) is to perform gradient descent over \( W \), which we update using all the samples of the few-shot task (both support and query) at once (i.e., no mini-batch sampling). This gradient approach yields our overall best results, while being one order of magnitude faster than the transductive entropy-based fine-tuning in \( [7] \). As will be shown later in our experiments, the method in \( [7] \) needs to fine-tune the whole network (i.e., to update both \( \phi \) and \( W \)), which provides implicit regularization, avoiding the degenerate solutions of entropy minimization. However, TIM-GD (with \( W \)-updates only) still remains two

\(^3\)The global minima of each pointwise entropy in the sum of \( \hat{H}(Y_Q|X_Q) \) are one-hot vectors at the vertices of the simplex.
orders of magnitude slower than inductive closed-form solutions \(^\text{[38]}\). In the following, we present a more efficient solver for our problem.

**Alternating direction method (TIM-ADM):** We derive an Alternating Direction Method (ADM) for minimizing our objective in \(^\text{[3]}\). Such scheme yields substantial speedups in transductive learning (one order of magnitude), while maintaining the high levels of accuracy of TIM-GD. To do so, we introduce auxiliary variables representing latent assignments of query samples, and minimize a mixed-variable objective by alternating two sub-steps, one optimizing w.r.t classifier’s weights \(W\), and the other w.r.t the auxiliary variables \(q\).

**Proposition 1.** The objective in \(^\text{[3]}\) can be approximately minimized via the following constrained formulation of the problem:

\[
\min_{W,q} \left( \sum_{i \in S} \sum_{k=1}^{K} y_{ik} \log(p_{ik}) + \sum_{k=1}^{K} q_k \log \hat{q}_k - \frac{\alpha}{|Q|} \sum_{i \in Q} \sum_{k=1}^{K} q_{ik} \log(p_{ik}) + \frac{1}{|Q|} \sum_{i \in Q} \sum_{k=1}^{K} q_{ik} \log \frac{q_{ik}}{p_{ik}} \right)
\]

\[\text{s.t. } \sum_{k=1}^{K} q_{ik} = 1, \quad q_{ik} \geq 0, \quad i \in Q, \quad k \in \{1, \ldots, K\},\]

where \(q = [q_{ik}] \in \mathbb{R}^{|Q| \times K}\) are auxiliary variables, \(p = [p_{ik}] \in \mathbb{R}^{|Q| \times K}\) and \(\hat{q}_k = \frac{1}{|Q|} \sum_{i \in Q} q_{ik}\).

*Proof.* It is straightforward to notice that, when equality constraints \(q_{ik} = p_{ik}\) are satisfied, the last term in objective \(^\text{[4]}\), which can be viewed as a soft penalty for enforcing those equality constraints, vanishes. Objectives \(^\text{[3]}\) and \(^\text{[4]}\) then become equivalent. \(\square\)

Splitting the problem into sub-problems on \(W\) and \(q\) as in Eq. \(^\text{[4]}\) is closely related to the general principle of ADMM (Alternating Direction Method of Multipliers) \(^\text{[4]}\), except that the KL divergence is not a typical penalty for imposing the equality constraints. The main idea is to **decompose the original problem into two easier sub-problems**, one over \(W\) and the other over \(q\), which can be alternately solved, each in closed-form. Interestingly, this KL penalty is important as it completely removes the need for dual iterations for the simplex constraints in \(^\text{[4]}\), yielding closed-form solutions:

**Proposition 2.** ADM formulation in Proposition 1 can be approximately solved by alternating the following closed-form updates w.r.t auxiliary variables \(q\) and classifier weights \(W\) (\(t\) is the iteration index):

\[
q_{ik}^{(t+1)} \propto \frac{\left(p_{ik}^{(t)}\right)^{1+\alpha}}{\left(\sum_{i \in Q} \left(p_{ik}^{(t)}\right)^{1+\alpha}\right)^{1/2}}
\]

\[
w_k^{(t+1)} \leftarrow \frac{\lambda}{1 + \alpha} \sum_{i \in S} y_{ik} z_i + p_{ik}^{(t)}(u_k^{(t)} - z_i) + \frac{|S|}{|Q|} \sum_{i \in Q} \left(q_{ik}^{(t+1)} z_i + p_{ik}^{(t)}(u_k^{(t)} - z_i)\right)
\]

\[
\sum_{i \in S} y_{ik} + \frac{|S|}{|Q|} \sum_{i \in Q} q_{ik}^{(t+1)}
\]

*Proof.* A detailed proof is deferred to the supplementary material. Here, we summarize the main technical ingredients of the approximation. Keeping the auxiliary variables \(q\) fixed, we optimize a convex approximation of Eq. \(^\text{[4]}\) w.r.t \(W\). With \(W\) fixed, the objective is strictly convex w.r.t the auxiliary variables \(q\) whose updates come from a closed-form solution of the KKT (Karush–Kuhn–Tucker) conditions. Interestingly, the negative entropy of auxiliary variables, which appears in the penalty term, handles implicitly the simplex constraints, which removes the need for dual iterations to solve the KKT conditions. \(\square\)

Typically, ADMM methods use multiplier-based quadratic penalties for enforcing the equality constraint.
3 Experiments

Hyperparameters: To keep our experiments as simple as possible, our hyperparameters are kept fixed across all the experiments and methods (TIM-GD and TIM-ADM). The conditional entropy weight $\alpha$ and the cross-entropy weights $\lambda$ in Objective (3) are both set to 0.1. The temperature parameter $\gamma$ in the classifier is set to 15. In our TIM-GD method, we use the ADAM optimizer with the recommended parameters [20], and run 1000 iterations for each task. For TIM-ADM, we run 150 iterations.

Base-training procedure: The feature extractors are trained following the same simple base-training procedure as in [51] and using standard networks (ResNet-18 and WRN28-10), for all the experiments. Specifically, they are trained using the standard cross-entropy loss on the base classes, with label smoothing. The label-smoothing parameter is set to 0.1. We emphasize that base training does not involve any meta-learning or episodic training strategy. The models are trained for 90 epochs, with the learning rate initialized to 0.1, and divided by 10 at epochs 45 and 66. Batch size is set to 256 for ResNet-18, and to 128 for WRN28-10. During training, all the images are resized to $84 \times 84$, and we used the same data augmentation procedure as in [51], which includes random cropping, color jitter and random horizontal flipping.

Datasets: We resort to 3 few-shot learning datasets to benchmark the proposed models. As standard few-shot benchmarks, we use the mini-ImageNet [45] dataset, with 100 classes split as in [35], the Caltech-UCSD Birds 200 [47] (CUB) dataset, with 200 classes, split following [5], and finally the larger tiered-ImageNet dataset, with 608 classes split as in [36].

3.1 Comparison to state-of-the-art

We first evaluate our methods TIM-GD and TIM-ADM on the widely adopted mini-ImageNet, tiered-ImageNet and CUB benchmark datasets, in the most common 1-shot 5-way and 5-shot 5-way scenarios, with 15 query shots for each class. Results are reported in Table 1, and are averaged over 10,000 episodes, following [46]. We can observe that both TIM-GD and TIM-ADM yield state-of-the-art performances, consistently across all standard datasets, scenarios and backbones, improving over both transductive and inductive methods, by significant margins.

3.2 Impact of domain-shift

Chen et al. [5] recently showed that the performance of most meta-learning methods may drop drastically when a domain-shift exists between the base training data and test data. Surprisingly, the simplest discriminative baseline exhibited the best performance in this case. Therefore, we evaluate our methods in this challenging scenario. To this end, we simulate a domain shift by training the feature encoder on mini-ImageNet while evaluating the methods on CUB, similarly to the setting introduced in [5]. TIM-GD and TIM-ADM beat previous methods by significant margins in the domain-shift scenario, consistently with our results in the standard few-shot benchmarks, thereby demonstrating an increased potential of applicability to real-world situations.

3.3 Pushing the meta-testing stage

Most few-shot papers only evaluate their method in the usual 5-way scenario. Nevertheless, [5] showed that meta-learning methods could be beaten by their discriminative baseline when more ways were introduced in each task. Therefore, we also provide results of our method in the more challenging 10-way and 20-way scenarios on mini-ImageNet. These results, which are presented in Table 3, show that TIM-GD outperforms other methods by significant margins, in both settings.

3.4 Ablation study

Influence of each term: We now assess the impact of each term in our loss in Eq. (5) on the final performance of our methods. The results are reported in Table 4. We observe that integrating the three terms in our loss consistently outperforms any other configuration. Interestingly, removing

The $W$ and $q$ updates of TIM-ADM associated to each configuration can be found in the supplementary material.
Table 1: Comparison to the state-of-the-art methods on mini-ImageNet, tiered-Imagenet and CUB. The methods are sub-grouped into transductive and inductive methods, as well as by backbone architecture. Our results (gray-shaded) are averaged over 10,000 episodes. "-" signifies the result is unavailable.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>MAML [9]</td>
<td></td>
<td>ResNet-18</td>
<td>49.6</td>
<td>65.7</td>
<td>-</td>
<td>-</td>
<td>68.4</td>
<td>83.5</td>
</tr>
<tr>
<td>RelatNet [40]</td>
<td></td>
<td>ResNet-18</td>
<td>52.5</td>
<td>69.8</td>
<td>-</td>
<td>-</td>
<td>68.6</td>
<td>84.0</td>
</tr>
<tr>
<td>MatchNet [45]</td>
<td></td>
<td>ResNet-18</td>
<td>52.9</td>
<td>68.9</td>
<td>-</td>
<td>-</td>
<td>73.5</td>
<td>84.5</td>
</tr>
<tr>
<td>ProtoNet [38]</td>
<td></td>
<td>ResNet-18</td>
<td>54.2</td>
<td>73.4</td>
<td>-</td>
<td>-</td>
<td>73.0</td>
<td>86.6</td>
</tr>
<tr>
<td>MTL [39]</td>
<td>×</td>
<td>ResNet-12</td>
<td>61.2</td>
<td>75.5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>vFSL [50]</td>
<td></td>
<td>ResNet-12</td>
<td>61.2</td>
<td>77.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Neg-cosine [26]</td>
<td></td>
<td>ResNet-18</td>
<td>62.3</td>
<td>80.9</td>
<td>-</td>
<td>-</td>
<td>72.7</td>
<td>89.4</td>
</tr>
<tr>
<td>MetaOpt [22]</td>
<td></td>
<td>ResNet-12</td>
<td>62.6</td>
<td>78.6</td>
<td>66.0</td>
<td>81.6</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SimpleShot [46]</td>
<td></td>
<td>ResNet-18</td>
<td>62.9</td>
<td>80.0</td>
<td>68.9</td>
<td>84.6</td>
<td>68.9</td>
<td>84.0</td>
</tr>
<tr>
<td>Distill [41]</td>
<td></td>
<td>ResNet-12</td>
<td>64.8</td>
<td>82.1</td>
<td>71.5</td>
<td>86.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>RelatNet + T [14]</td>
<td></td>
<td>ResNet-12</td>
<td>52.4</td>
<td>65.4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ProtoNet + T [14]</td>
<td></td>
<td>ResNet-12</td>
<td>55.2</td>
<td>71.1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MatchNet+T [14]</td>
<td></td>
<td>ResNet-12</td>
<td>56.3</td>
<td>69.8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TPN [28]</td>
<td></td>
<td>ResNet-12</td>
<td>59.5</td>
<td>75.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TEAM [34]</td>
<td>✓</td>
<td>ResNet-18</td>
<td>60.1</td>
<td>75.9</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Ent-min [7]</td>
<td></td>
<td>ResNet-12</td>
<td>62.4</td>
<td>74.5</td>
<td>68.4</td>
<td>83.4</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CAN+T [14]</td>
<td></td>
<td>ResNet-12</td>
<td>67.2</td>
<td>80.6</td>
<td>73.2</td>
<td>84.9</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>LaplacianShot [51]</td>
<td></td>
<td>ResNet-18</td>
<td>72.1</td>
<td>82.3</td>
<td>79.0</td>
<td>86.4</td>
<td>81.0</td>
<td>88.7</td>
</tr>
<tr>
<td>TIM-ADM</td>
<td></td>
<td>ResNet-18</td>
<td>73.6</td>
<td>85.0</td>
<td>80.0</td>
<td>88.5</td>
<td>81.9</td>
<td>90.7</td>
</tr>
<tr>
<td>TIM-GD</td>
<td></td>
<td>ResNet-18</td>
<td>73.9</td>
<td>85.0</td>
<td>79.9</td>
<td>88.5</td>
<td>82.2</td>
<td>90.8</td>
</tr>
<tr>
<td>LEO [37]</td>
<td></td>
<td>WRN28-10</td>
<td>61.8</td>
<td>77.6</td>
<td>66.3</td>
<td>81.4</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SimpleShot [46]</td>
<td></td>
<td>WRN28-10</td>
<td>63.5</td>
<td>80.3</td>
<td>69.8</td>
<td>85.3</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MatchNet [45]</td>
<td>×</td>
<td>WRN28-10</td>
<td>64.0</td>
<td>76.3</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CC+rot+unlabeled [10]</td>
<td></td>
<td>WRN28-10</td>
<td>64.0</td>
<td>80.7</td>
<td>70.5</td>
<td>85.0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>FEAT [49]</td>
<td></td>
<td>WRN28-10</td>
<td>65.1</td>
<td>81.1</td>
<td>70.4</td>
<td>84.4</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>AWGIM [12]</td>
<td></td>
<td>WRN28-10</td>
<td>63.1</td>
<td>78.4</td>
<td>67.7</td>
<td>82.8</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Ent-min [7]</td>
<td></td>
<td>WRN28-10</td>
<td>65.7</td>
<td>78.4</td>
<td>73.3</td>
<td>85.5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SIB [15]</td>
<td></td>
<td>WRN28-10</td>
<td>70.0</td>
<td>79.2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BD-CSPN [27]</td>
<td>✓</td>
<td>WRN28-10</td>
<td>70.3</td>
<td>81.9</td>
<td>78.7</td>
<td>86.9</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>LaplacianShot [51]</td>
<td></td>
<td>WRN28-10</td>
<td>74.9</td>
<td>84.1</td>
<td>80.2</td>
<td>87.6</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TIM-ADM</td>
<td></td>
<td>WRN28-10</td>
<td>77.5</td>
<td>87.2</td>
<td>82.0</td>
<td>89.7</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TIM-GD</td>
<td></td>
<td>WRN28-10</td>
<td>77.8</td>
<td>87.4</td>
<td>82.1</td>
<td>89.8</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 2: The results for the domain-shift setting mini-Imagenet → CUB. The results obtained by our models (gray-shaded) are averaged over 10,000 episodes.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Backbone</th>
<th>5-shot</th>
</tr>
</thead>
<tbody>
<tr>
<td>MatchNet [45]</td>
<td>ResNet-18</td>
<td>53.1</td>
</tr>
<tr>
<td>ProtoNet [38]</td>
<td>ResNet-18</td>
<td>62.0</td>
</tr>
<tr>
<td>RelatNet [40]</td>
<td>ResNet-18</td>
<td>57.7</td>
</tr>
<tr>
<td>SimpleShot [46]</td>
<td>ResNet-18</td>
<td>64.0</td>
</tr>
<tr>
<td>GNN [42]</td>
<td>ResNet-10</td>
<td>66.9</td>
</tr>
<tr>
<td>Neg-Cosine [26]</td>
<td>ResNet-18</td>
<td>67.0</td>
</tr>
<tr>
<td>Baseline [5]</td>
<td>ResNet-18</td>
<td>65.6</td>
</tr>
<tr>
<td>LaplacianShot [51]</td>
<td>ResNet-18</td>
<td>66.3</td>
</tr>
<tr>
<td>TIM-ADM</td>
<td>ResNet-18</td>
<td>70.3</td>
</tr>
<tr>
<td>TIM-GD</td>
<td>ResNet-18</td>
<td>71.0</td>
</tr>
</tbody>
</table>
We hypothesize that the network’s intrinsic regularization (such as batch normalizations) and the use of ADM optimize w.r.t. \( \hat{H}(Y_Q) \) when used alone. This emphasizes the importance of the label-marginal entropy term \( H(Y_Q) \). The absence of \( \hat{H}(Y_Q) \) in the entropy-based loss \( CE + \hat{H}(Y_Q) \) does not cause the same drastic drop in performance as observed earlier when optimizing with respect to \( W \) only. We hypothesize that the network’s intrinsic regularization (such as batch normalizations) and the use of small learning rates, as prescribed by \([7]\), help the optimization process, preventing the predictions from approaching the vertices of the simplex, where entropy’s gradients diverge.

### 3.5 Inference run-times

Transductive methods are generally slower at inference than their inductive counterparts, with run-times that are, typically, several orders of magnitude larger. In Table 5 we measure the average inference run-times for the different methods.
Figure 1: Convergence plots for our methods on mini-ImageNet with a ResNet-18. Solid lines are averages, while shadows are 95% confidence intervals. Time is in logarithmic scale. **Left:** Evolution of the test accuracy during transductive inference. **Right:** Evolution of the mutual information between query features and predictions $\hat{I}(X_Q; Y_Q)$, computed as in Eq. (2), with $\alpha = 1$.

Table 5: Inference run-time per few-shot task for a 5-shot 5-way task on mini-ImageNet with a WRN28-10 backbone.

<table>
<thead>
<tr>
<th>Method</th>
<th>Parameters</th>
<th>Transductive</th>
<th>Inference/task (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SimpleShot</td>
<td>${W}$</td>
<td>✓</td>
<td>$9.0 \times 10^{-3}$</td>
</tr>
<tr>
<td>TIM-ADM</td>
<td>${W}$</td>
<td>✓</td>
<td>$1.2 \times 10^{-1}$</td>
</tr>
<tr>
<td>TIM-GD</td>
<td>${W}$</td>
<td>✓</td>
<td>$2.2 \times 10^{+0}$</td>
</tr>
<tr>
<td>Ent-min [7]</td>
<td>${\phi, W}$</td>
<td>✓</td>
<td>$2.1 \times 10^{+1}$</td>
</tr>
</tbody>
</table>

adaptation time per few-shot task, defined as the time required by each method to build the final classifier, for a 5-shot 5-way task on mini-ImageNet using the WRN28-10 network. Table 5 conveys that our ADM optimization gains one order of magnitude in run-time over our gradient-based method, and more than two orders of magnitude in comparison to [7], which fine-tunes the whole network. Note that TIM-ADM still remains slower than the inductive baseline. Our methods were run on the same GTX 1080 Ti GPU, while the run-time of [7] is directly reported from the paper.

4 Conclusion and future work

Our TIM inference establishes new state-of-the-art results on the standard few-shot benchmarks, as well as in more challenging scenarios, with larger numbers of classes and domain shifts. We used feature extractors based on a simple base-class training with the standard cross-entropy loss, without resorting to the complex meta-training schemes that are often used and advocated in the recent few-shot literature. TIM is modular: it could be plugged on top of any feature extractor and base training, regardless of how the training was conducted. Therefore, while we do not claim that the very challenging few-shot problem is solved, we believe that our model-agnostic TIM inference should be used as a strong baseline for future few-shot learning research. In future work, we target on giving a more theoretical ground for our proposed mutual-information objective, and on exploring further generalizations of the objective, e.g., via embedding domain-knowledge priors. Specifically, one of our theoretical goals will be to connect TIM’s objective to the classifier’s empirical risk on the query set, showing that the former could be viewed as a surrogate for the latter.

5 Acknowledgements

This research was supported by the National Science and Engineering Research Council of Canada (NSERC), via its Discovery Grant program.
Broader impact

Due to the simplicity and efficiency of our method, we lower the barrier of entry to few-shot learning. In turn, we think that it will make a wider breadth of real-world applications tractable. The impact (positive or negative) on society is similar to that of any other few-shot method: being only a tool, its impact is entirely dependent on the final applications, and on the intentions of the people and institutions deploying it.

In our strive towards finding simple and efficient formulations -- for instance, we stick to a standard cross-entropy, which not only eases implementation, but also avoid the huge memory consumption of more complex methods -- we believe our method can enable and empower persons and communities that are unable to afford the costly resources and infrastructures required. This may help level the playing field with larger and better funded entities. For instance, to be adapted to a new task, our TIM-ADM method requires a little more than a recent smartphone computational power. This could spawn a lot of fresh and new applications on edge devices, closer to the end-users, in real-time.
References


