

Fine-tuning CLIP’s Last Visual Projector: a Few-shot Cornucopia

Mohammad Fahes¹ Tuan-Hung Vu^{1,2} Andrei Bursuc^{1,2} Patrick Pérez³ Raoul de Charette¹
¹ Inria ² Valeo.ai ³ Kyutai

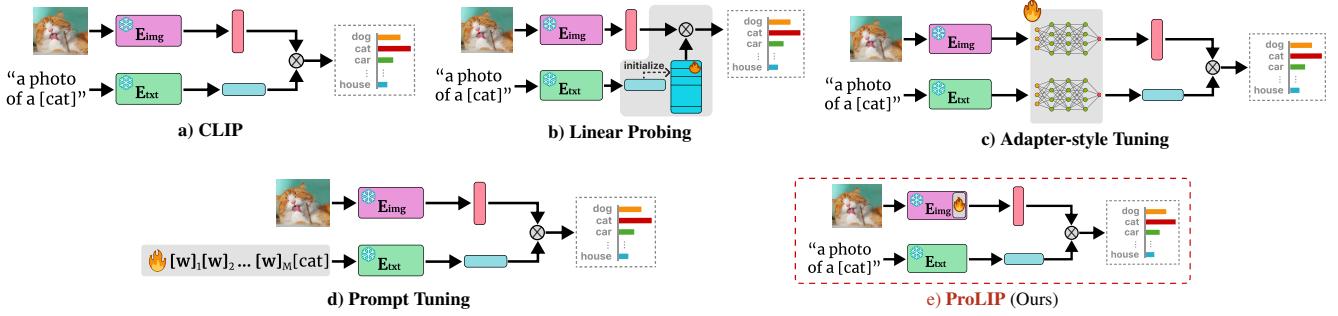


Figure 1. **Few-shot classification with CLIP.** a) Using a pre-trained CLIP, zero-shot classification is performed by measuring text and visual embeddings similarity. Among few-shot adaptation strategies of CLIP, b) Linear Probing trains a linear classifier of the visual features , c) Adapter-style tuning adds external learnable MLPs, d) Prompt Tuning learns word embeddings. Alternatively, we propose e) ProLIP which does not introduce new weights and only fine-tunes the last linear projection layer of the vision encoder.

Abstract

We consider the problem of adapting a contrastively pre-trained vision-language model like CLIP [30] for few-shot classification. The literature addresses this problem by learning a linear classifier of the frozen visual features, optimizing word embeddings, or learning external feature adapters. This paper introduces an alternative way for CLIP adaptation without adding “external” parameters to optimize. We find that simply fine-tuning the last projection matrix of the vision encoder leads to performance better than all baselines. Furthermore, we show that regularizing training with the distance between the fine-tuned and pretrained matrices adds reliability for adapting CLIP. This simple approach, coined ProLIP, yields state-of-the-art performance on 11 few-shot classification benchmarks, few-shot domain generalization, cross-dataset transfer, base-to-new class generalization, and test-time adaptation. Code will be made available at: <https://github.com/astra-vision/ProLIP>.

1. Introduction

Contrastive Language-Image Pretraining (CLIP) [30] has shown that strong visual features can be learned from noisy natural language descriptions at very large scale. The true potential of CLIP lies in its shared vision-text space, break-

ing the long-standing constraints of closed-set systems and enabling non-trivial interactions and querying between text and images via prompts. Such a freedom in the label space makes the model readily applicable to a wide range of specialized downstream applications.

CLIP trains a vision and a text encoder on large batches of image-text pairs using a sum of contrastive image-to-text and text-to-image losses. At inference, given an image and a set of classes expressed within prompts (e.g., “a photo of {class_k}”), one can perform *zero-shot classification*. The predicted class is simply the one for which the textual embedding has the highest similarity with the image embedding. The prompt template can be engineered to boost the zero-shot performance, or automated by querying multiple descriptors of a class from Large Language Model (LLMs), such as GPT-3 [3], and ensembling their embeddings [26]. Yet, the zero-shot performance may still be unsatisfying, especially for data that are supposedly under-represented in CLIP training data. Examples of such cases include geospatial data, e.g., EuroSAT [12] and specialized data, e.g., FGVCAircraft [25]. Thus, an interesting practical setting emerged in transfer learning: *Given a labeled few-shot training dataset of images, how to efficiently adapt CLIP in order to maximize the performance on the test set?*

Hinging on only a few labeled samples for supervision, model training is prone to overfitting. The common strategy is to avoid full fine-tuning and instead adapt only a few parameters [21]. Starting from a concept-rich pretrained CLIP

model, such parameter-efficient strategies have been shown to be effective for few-shot tasks. In this direction, the literature explores three avenues. First, prompt tuning [48, 49] replaces the template with learnable parameters in the word embedding space, while freezing both vision and text encoders. Second, CLIP adapters [9] learn a multi-layer perceptron (MLP) on top of the frozen visual or text features, and use a residual connection aiming at partially using the zero-shot features. In these two paradigms, the text embeddings are used as classification weights. Third, linear probing [30] simply trains a linear classifier on top of the frozen visual features.

Shortcomings. While existing solutions are technically simple and parameter-efficient, we identify several limitations. Prompt tuning methods [4, 49, 50] are slow to train as gradients need to be backpropagated over the entire text encoder. In addition, different context lengths and class-name positions lead to different performances. Adapters [9, 46] impose architectural choices of the MLP, the bottleneck dimension and the residual connection. On the other hand, while initially suggested as a few-shot baseline for CLIP, the performance of linear probing (LP) lies far behind adapters and prompt tuning. Its main shortcoming stems from ignoring the text embeddings during adaptation. Recently, Huang et al. [17] proposed LP++, an improved LP version blending textual embeddings with classification weights using class-wise learnable parameters. While LP++ [17] shows significant improvements over standard LP, we argue in this work that directly using the text embeddings as classification weights might be a better practice for fine-tuning CLIP, as it is closer in principle to its original pretraining regime. Another major limitation of linear probing methods [17, 24, 30, 34] is that they cannot be applied in open-class and cross-dataset transfer settings: the classifier is restricted to the set of classes contained in the downstream few-shot training dataset.

The previous methods either train “external” parameters (e.g., Adapters, LP), or learn parameters in the input space (e.g., prompt tuning). To the best of our knowledge, no existing work tackles few-shot CLIP adaptation problem with parameter-efficient fine-tuning of the model weights. In this work, we propose a first baseline for model weights based few-shot learning. Our method, dubbed “ProLIP”, is both extremely simple to implement, being only a few lines of code, and very effective: *considering pretrained visual and text CLIP encoders f and g , we fine-tune the last projection matrix of f (i.e., the projector mapping visual embeddings into the shared embedding space) with a cross-entropy loss while constraining its weights to remain close to the pretrained ones.* Fig. 1 illustrates existing approaches and our proposed ProLIP, which is further detailed in Sec. 4.

ProLIP is advantageous for a number of reasons:

- It alleviates the need of “external” parameters which usu-

ally imply architectural design search and/or heavy hyperparameter selection.

- As backpropagation is only applied on the last projector of the vision encoder, training is fast, requiring only few seconds like LP++ [17].
 - ProLIP uses native text embeddings as classification weights in the few-shot task, following CLIP pretraining, and therefore preserves its open-class capability.
 - It balances pretraining and adaptation with a regularizer that constrains the distance w.r.t. the pretrained weights.
- Our simple method performs on par with or better than the literature on few-shot adaptation, few-shot domain generalization, cross-dataset generalization and base-to-new class generalization. Additionally, ProLIP significantly outperforms prompt tuning in test-time adaptation, which is an unsupervised learning setting.

2. Related work

Parameter-efficient fine-tuning (PEFT). The advent of increasingly larger pretrained vision foundation models with excellent generalization capabilities has opened the way to new transfer learning approaches towards downstream tasks with limited labeled data. Full fine-tuning of such models turns out to be not only computationally inefficient but also often underperforming, even when compared to linear probing [21, 39]. Parameter-efficient fine-tuning methods aim to adapt models effectively with minimal changes of their parameters while freezing most of the large pretrained backbone. Side-tuning [45] trains a small network in parallel to a frozen pretrained network and avoids catastrophic forgetting. Optimizing only a specific subset of parameters of a model, e.g., bias terms [43], is also an effective strategy. However this still requires full backpropagation through the pretrained model. Adapter-tuning methods add lightweight modules to transformer layers [15, 32], but incur a higher runtime cost. LoRA [16] optimizes new low-rank matrices injected to transformer layers to approximate weight changes during fine-tuning, reducing significantly the number of parameters to learn. Prompt-tuning approaches, such as VPT [18], add a set of learnable prompts to the set of input patch embeddings. In addition to incurring supplementary computational cost for the full backpropagation and for runtime for most of them, these methods are specifically devised for transformer layers and are not directly applicable to convolutional nets.

Few-shot CLIP adaptation. CLIP’s specific interaction between text and image features has enabled new adaptation methods fully exploiting this property in particular for the few-shot regime. Inspired by prompt tuning in natural language processing [23, 47], Zhou et al. [49] proposed context optimization (CoOp) which applies the same concept for pretrained vision-language models. CoOp was

later shown not to generalize well on unseen classes within the same dataset. Thus, conditional context Optimization (CoCoOp) [48] adds a meta-network that generates input-conditional tokens in addition to the learnable vectors, making optimized context less prone to overfitting to the seen classes. Zhu et al. [50] identified a critical problem of unconstrained prompt tuning methods: in extreme low-shot settings, overfitting can even decrease the zero-shot performance. They proposed regularizing the training by only updating prompts whose gradients do not conflict the direction resulting from zero-shot predictions. PLOT [4] applies optimal transport on sets of text and visual features to learn the transport plan between the two sets in an inner loop, which is fixed in the outer loop where prompts are learned. MaPLE [19] learns prompts in both vision and text branches at input and intermediate layers with a coupling function.

A simple approach for few-shot CLIP is training a linear probe on top of the visual features [30]. Lin et al. [24] show that adding a text describing the class to the training data of few-shot images largely boosts linear probing. Huang et al. [17] blend text embeddings with classification weights using class-wise learnable parameters.

Instead of adapting the model in the input space or training a linear probe, CLIP-adapter [9] adds an MLP on top of the features in the shared embedding space, with a residual connection to preserve the pretrained knowledge. Zhang et al. [46] showed that training-free adapters can be competitive to trained ones. They create a cache-model for the few-shot training set from the visual features and the corresponding ground-truth labels, which are converted to the weights of the MLP adapter. Following the success of training-free CLIP adaptation, Wang et al. [38] resorted to Gaussian Discriminative Analysis (GDA) [1] which assumes the conditional distribution of features given the labels follows a multivariate normal distribution. Thus, they construct the GDA classifier using the mean vectors of each class and the inverse covariance matrix, and show that classification can be further improved by ensembling GDA and zero-shot classifiers. Our ProLIP takes advantage of CLIP’s compatibility between text and images and adapts it to downstream tasks without additional learnable parameters or architecture changes.

3. Preliminaries

3.1. Zero-shot classification

We denote f and g the vision and text encoders of CLIP, respectively. During pretraining, CLIP learns a joint embedding space that pulls corresponding image-text representations closer together and pushes away dissimilar ones. At inference, given an image \mathbf{l} , one only needs the names of K candidate classes to perform *zero-shot classification*:

$$\hat{k} = \operatorname{argmax}_k \mathbf{v}^\top \mathbf{t}_k, \quad (1)$$

where $\mathbf{v} = \frac{f(\mathbf{l}; \theta_f)}{\|f(\mathbf{l}; \theta_f)\|_2}$, $\mathbf{t}_k = \frac{g(\mathbf{T}_k; \theta_g)}{\|g(\mathbf{T}_k; \theta_g)\|_2}$; θ_f and θ_g are the frozen parameters of f and g , respectively; \mathbf{T}_k is a text prompt describing the class k , e.g., “a photo of {class $_k$ }”.

3.2. Few-shot classification

Given a set of N labeled samples from each of K classes, research has been conducted to efficiently adapt CLIP using this set. All existing research in this direction can be gathered under three main avenues.

Prompt Tuning. It parameterizes the prompt template, i.e., $\mathbf{T}_k = [\mathbf{w}]_1[\mathbf{w}]_2\dots[\mathbf{w}]_M[\text{class}_k]$. $[\mathbf{w}]_1[\mathbf{w}]_2\dots[\mathbf{w}]_M$ are learned while keeping f and g frozen. Prompt tuning adapts CLIP “indirectly” on the classifier side, i.e., the text embeddings are derived from the learned prompts.

Adapters. They learn a multi-layer perceptron (MLP) h with a residual connection α on top of the frozen visual features \mathbf{v} , i.e., $\mathbf{v} := \alpha\mathbf{v} + (1 - \alpha)h(\mathbf{v})$, or on top of the frozen text features \mathbf{t} , i.e., $\mathbf{t} := \alpha\mathbf{t} + (1 - \alpha)h(\mathbf{t})$, or both.

Linear probing. It trains a linear classifier $\mathbf{W} \in \mathbb{R}^{D \times K}$ on top of the frozen visual features, D being the embedding space dimension. Matrix \mathbf{W} can be initialized with text embeddings \mathbf{t}_k as its columns. Since the classifier is directly tuned, linear probing restricts CLIP to K classes after adaptation and cannot be applied in open-class setting.

3.3. CLIP architecture

CLIP adopts a transformer architecture [36] for the text encoder, but the vision encoder may be either a ResNet [11] or a Vision Transformer (ViT) [7]. We detail both architectures below and later elaborate on our unified method applicable to both architectures regardless of their intrinsic differences.

ResNet. CLIP replaces the global average pooling layer in ResNet with an attention pooling layer. The output of the multi-head attention layer is then projected to the shared latent space using a linear layer. Thus, f can be written as $f = f_1 \circ f_2$, where f_1 represents all the layers up to the attention pooling (included), and f_2 represents the linear projection head. Given an image \mathbf{l} :

$$\mathbf{x}_o = f_1(\mathbf{l}), \quad \mathbf{v} = f_2(\mathbf{x}_o) = \mathbf{W}_o^\top \mathbf{x}_o + \mathbf{b}_o, \quad (2)$$

with $\mathbf{x}_o \in \mathbb{R}^{D_o}$ the output of the attention pooling layer, $\mathbf{W}_o \in \mathbb{R}^{D_o \times D}$ the projection matrix and \mathbf{b}_o a bias term.

ViT. The transformer encoder consists of multiple residual attention blocks. Each block has two main components: a multi-head self-attention and a feed-forward neural network (MLP), with residual connections. The output of the last residual attention block is projected to the latent space using a trainable matrix. Thus, f can be written as $f = f_1 \circ f_2$,

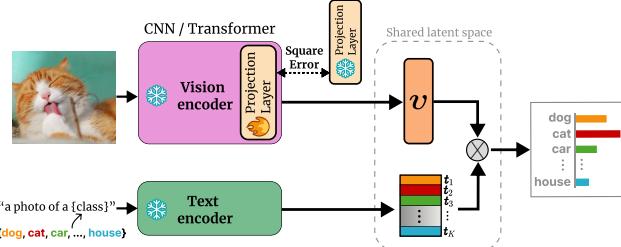


Figure 2. **ProLIP for few-shot adaptation.** Whether the vision encoder is a CNN or a Transformer, ProLIP trains only the final linear layer that projects the visual embeddings into the shared latent space. The text encoder is frozen, and the text embeddings of the K target concepts are used as classification weights. Training with cross-entropy is regularized by a squared error loss ensuring weights of the projection layer to remain close to pretrained ones.

where f_1 represents all the layers up to the last residual attention block (included), and f_2 represents the projection matrix. Given an image \mathbf{l} :

$$\mathbf{x}_o = f_1(\mathbf{l}), \quad \mathbf{v} = f_2(\mathbf{x}_o) = \mathbf{W}_o^\top \mathbf{x}_o, \quad (3)$$

where no bias term is included, unlike Eq. (2).

Similarly on the text side, the embeddings are projected into the shared latent space using a linear layer.

4. ProLIP

As discussed in Sec. 3.3, CLIP projects both visual and text embeddings into the shared latent space using linear layers. We show that fine-tuning only the projection matrix \mathbf{W}_o in Eq. (2) or Eq. (3) can be a strong alternative to prompt tuning and feature adapters. Specifically, the probability that a sample i belongs to the class k is computed as the Softmax over cosine similarities of image-text embeddings:

$$p_{ik}(\mathbf{W}_o) = \frac{\exp((\mathbf{W}_o^\top \mathbf{x}_{oi} + \mathbf{b}_o)^\top \mathbf{t}_k / \tau)}{\sum_{j=1}^K \exp((\mathbf{W}_o^\top \mathbf{x}_{oi} + \mathbf{b}_o)^\top \mathbf{t}_j / \tau)}, \quad (4)$$

with \mathbf{t}_k being fixed since g is frozen, τ the pretraining temperature parameter, and \mathbf{x}_{oi} the pre-projection embedding of sample i . The matrix \mathbf{W}_o is learned with gradient descent using a cross-entropy loss $L(\mathbf{W}_o)$:

$$L(\mathbf{W}_o) = -\frac{1}{N} \sum_{i=1}^N \sum_{k=1}^K y_{ik} \log p_{ik}(\mathbf{W}_o), \quad (5)$$

where y_{ik} is the ground truth.

Regularization. CLIP encoders map text and image modalities into a common latent space where strong image-text representation correspondences are established. We argue that unconstrained fine-tuning can lead to forgetting of the rich pretraining knowledge that appears through non-trivial

zero-shot classification accuracies. Thus, a good fine-tuning strategy should balance pretraining knowledge preservation and adaptation to downstream task. Consequently, to prevent significant drift from the pretraining weights (*i.e.*, knowledge forgetting), we regularize the training with the Frobenius norm of the difference between the pretrained and fine-tuned matrices. The total loss is:

$$\text{Loss} = L(\mathbf{W}_o) + \lambda \|\mathbf{W}_o - \mathbf{W}_o^{(0)}\|_{\text{F}}, \quad (6)$$

where $\mathbf{W}_o^{(0)}$ denotes the pretrained value of \mathbf{W}_o . We show later that λ can be chosen as a decreasing function of the number of shots, as overfitting risk increases with less data [10]. The method is illustrated in Fig. 2.

An argument on simplicity and practicality. Algorithm 1 provides a PyTorch-like [29] pseudo-code for ProLIP, showing that it is extremely simple to implement. Also, it can be applied on pre-processed data, which makes it also extremely fast to train. Practically, we run the inference only one time on the text encoder side to get the classification weights. On the vision encoder side, we save the output embeddings \mathbf{x}_o with different augmentation views since backpropagation is limited to the projection matrix.

Despite its extreme simplicity, adapting CLIP using our approach has not been proposed in the literature. Moreover, its architecture-agnostic nature makes it generic and suitable to different multi-modal pretrained networks. We argue that, due to intrinsic differences across architectures (*e.g.*, ViT *vs.* ResNet), finding a unified method to efficiently fine-tune vision-language models based on the pretrained weights is not trivial. ProLIP is a first to achieve this goal.

Algorithm 1 PyTorch-like pseudo-code for ProLIP.

```

# target: Ground truth
# lmda: regularization loss weight
# Wo : Pretrained projection matrix
# xo: output visual embeddings (N*K, Do)
# text_weights: normalized embeddings ofclassnames (K,D)

# Copy initial weights for use in the regularization loss
Wo_0 = copy.deepcopy(Wo)
# Set last projection matrix as trainable weights
Wo.requires_grad = True
bo.requires_grad = False

v = xo @ Wo + bo
v = 12_normalize(v, dim=-1)

#compute the cosine similarity scores
logits = 100. * v @ text_weights.T

#compute regularized loss
SE_loss = nn.MSELoss(reduction='sum')
loss = CE_loss(logits, target) + lmda * SE_loss(Wo, Wo_0)

```

5. Experiments

Datasets. Following previous CLIP-based few-shot learning works, we experimentally test ProLIP on 11 datasets

for few-shot classification and base-to-new generalization: ImageNet [6], SUN397 [40], DTD [5], Caltech101 [8], UCF101 [35], Flowers102 [27], Stanford-Cars [20], FGVC Aircraft [25], EuroSAT [12], Oxford-Pets [28] and Food101 [2]. For domain generalization experiments we follow ProGrad [50], using ImageNet as source dataset and testing on ImageNet-V2 [31], ImageNet-Sketch [37], ImageNet-A [14] and ImageNet-R [13] as out-of-distribution datasets. For the cross-dataset transfer experiment, ProLIP is trained on ImageNet and evaluated on the other 10 datasets, similar to ProGrad [50]. For test-time adaptation, we use ImageNet and its out-of-distribution (OOD) variants similarly to TPT [33].

Training details. We follow previous works and use $N \in \{1, 2, 4, 8, 16\}$ shots as support training set for few-shot classification. LP++ [17] identifies a flaw in the few-shot CLIP literature [46], which is the use of a large validation set for hyperparameter tuning. Instead, authors of LP++ propose using a validation set with as many shots as in the training set. We adopt this protocol in our few-shot classification experiments (cf. Sec. 5.1), though also experimenting in a more realistic setting without use of any validation set [34], which is adopted starting from Sec. 5.2. Huang et al. [17] also remark that prior works evaluate their methods based on one or three support sets, leading to large standard deviations when the few-shot set is not representative of the class distribution. We follow their practice and evaluate ProLIP on 10 random seeds (i.e., support training sets) for each dataset. For domain generalization and cross-dataset transfer experiments, we select $N=4$ like ProGrad [50]. For base-to-new generalization, we select $N=4$ like ProGrad [50] when using ResNet-50 and $N=16$ like MaPLe [19] when using ViT-B/16. Unless otherwise stated, we employ ResNet-50 with CLIP weights as the visual encoder, similarly to the literature. Training runs for 300 epochs on a full-batch of features, requiring few seconds on one Tesla V100. For ProLIP, the learning rate (LR) and regularizer loss weight λ are selected by grid search on the few-shot validation set, with $LR \in \{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}\}$ and $\lambda \in \{10, 1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 0\}$. We later show that using the regularizer ($\lambda > 0$) is better to avoid severe overfitting, and that state-of-the-art results can be still achieved with a fixed LR over all the datasets, and a fixed λ chosen as a decreasing function of the number N of shots.

5.1. Few-shot classification

We compare ProLIP to baselines covering the variety of existing adaptation strategies. For prompt tuning methods, we report CoOp [49] and its other variants PLOT [4], KgCoOp [41] and ProGrad [50]. For adapters, we compare to CLIP-adapter [9] and Tip-adapter [46]. Note that Tip-adapter performance is reported in two settings follow-

Method	$N = 1$	2	4	8	16
CLIP				58.89	
<i>Prompt tuning</i>					
CoOp [49]	59.62 ± 3.11	63.80 ± 2.32	67.23 ± 1.64	71.30 ± 0.86	74.06 ± 0.55
PLOT [4]	61.51 ± 2.91	65.67 ± 2.06	68.39 ± 1.17	71.96 ± 0.70	74.35 ± 0.66
KgCoOp [41]	61.36 ± 3.04	63.23 ± 2.06	65.73 ± 1.15	67.50 ± 1.11	69.01 ± 0.79
ProGrad [50]	62.46 ± 1.89	65.88 ± 1.46	68.52 ± 1.15	71.82 ± 0.11	73.95 ± 0.68
<i>Adapters</i>					
CLIP-Adapter [9]	60.32 ± 0.80	61.93 ± 0.93	65.12 ± 0.80	69.20 ± 0.56	72.57 ± 0.54
Tip-Adapter-F [46]	61.29 ± 0.92	62.94 ± 0.75	66.02 ± 0.80	69.88 ± 0.51	73.82 ± 0.55
Tip-Adapter-F* [46]	63.06 ± 1.05	66.47 ± 0.65	68.71 ± 0.96	71.78 ± 1.00	74.37 ± 0.35
<i>Linear Probing</i>					
LP [30]	36.10 ± 1.43	46.99 ± 1.29	56.72 ± 1.20	64.66 ± 0.55	70.56 ± 0.44
LP++ [17]	63.43 ± 0.90	66.20 ± 0.72	69.16 ± 0.79	72.04 ± 0.46	74.42 ± 0.45
<i>Model weights</i>					
ProLIP	64.59 ± 0.98	67.09 ± 0.87	70.53 ± 0.69	73.40 ± 0.45	76.55 ± 0.41

Table 1. **Few-shot image classification based on CLIP.** We report the classification accuracy (%) averaged over 11 datasets and 10 support sets, along with standard deviation, comparing ProLIP to other baselines. We highlight **best** and 2nd best. First row provides zero-shot classification accuracy for reference.

ing [17]: Tip-adapter-F where its two crucial hyperparameters are set to 1 and the validation set is used for early stopping, and Tip-adapter-F* where intensive hyperparameter search is performed to find the best values of the same hyperparameters based on the same validation set. For linear probing, we report LP [30] and LP++ [17]. Baseline results are taken from [17] which employs early stopping for all the methods based on the validation set. For a fair comparison we follow the same experimental protocol (i.e., few-shot validation set, 10 random support sets) although we stress that we do not use early stopping but rather report the last model performance.

Tab. 1 reports the average classification accuracy and standard deviation, across 11 datasets. Per-dataset performances are reported in Appendix A. In all few-shots settings (i.e., $N \in \{1, 2, 4, 8, 16\}$), ProLIP clearly outperforms all the baselines, showing a great potential of the extremely simple approach of fine-tuning the last visual linear projector for adaptation.

Hyperparameters sensitivity. The benefit of the regularization appears by testing ProLIP with different hyperparameters, which are fixed across all the datasets. Fig. 3 reports the average accuracy across the same 11 datasets, for 5 different LR values combined either with regularization ($\lambda \in \{10^{-2}, 10^{-1}, 1\}$) or without regularization ($\lambda=0$). For $\lambda=0$, we note that the accuracy drops dramatically for larger learning rates due to overfitting to the few-shot training set, and the subsequent drift from the robust pretrained CLIP representation. On the other hand, using the weight regularizer ($\lambda > 0$) makes ProLIP less prone to overfitting and less sensitive to learning rate.

This observation is corroborated by the statistics of the hyperparameters found by grid search (cf. Appendix E)

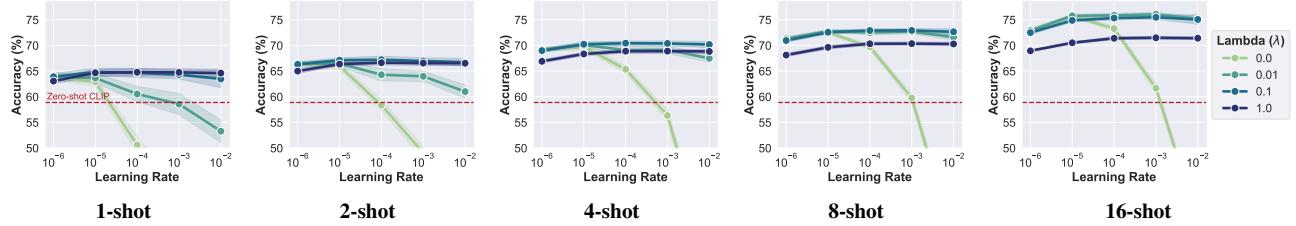


Figure 3. **ProLIP sensitivity to hyperparameter choice.** Accuracy of ProLIP to the hyperparameters (learning rate and regularization weight λ) for $N \in \{1, 2, 4, 8, 16\}$ -shot settings. Each data point is an average over 11 datasets, 10 runs for each.

showing that the best learning rates span a wide range of values. It follows that our regularization alleviates overfitting on the training set, allowing larger learning rate (e.g., 10^{-2}). This important property motivates our investigation of a more realistic setting, where hyperparameters are never tuned: *Having no validation data*.

5.2. Few-shot classification without validation set

In contrast to [17], we argue that relying even on N -shot validation sets is a violation of the N -shot setting since it effectively requires access to $2N$ examples (N for training and N for validation). Subsequently, like Silva et al. [34], we advocate for a true N -shot setting, using no validation data at all. It follows that the true merit of a method stems from its lower sensitivity to hyperparameters, which we demonstrated in the previous section.

It can be observed from Fig. 3 that for lower-shot settings, higher λ values lead to better accuracy, and vice versa. Therefore, we formulate λ as a decreasing function of the number of shots N , reporting in Tab. 2a the average performance over learning rates $LR \in \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$. It results that our simple parametric formulations of λ (i.e., $1/N$, $1/N^2$) lead to almost identical, strong and stable results, competing with our state-of-the-art grid search variant, albeit without the need of a validation set. A byproduct of our regularizer is the reduced sensitivity to the learning rate (i.e., low variance as seen in Tab. 2a), whereas removing the regularizer (i.e., $\lambda = 0$) proves to result in dramatically large variance. Detailed numbers for each combination are reported in Tab. 11.

Therefore, in Tab. 2b we compare the average across 11 datasets of the validation-free baselines from [34], and the validation-free ProLIP variant, coined as $\text{ProLIP}_\varnothing$, with $\lambda = 1/N$ and average over the 4 tested learning rates. The reported performance shows a consistent improvement for any number of shots.

5.3. Generalization of few-shot models

Real-world scenarios impose an additional challenge of distribution shift for model adaptation. Supposing the test data to follow the same distribution as the training is often unrealistic, and a model can be of practical interest only if it ex-

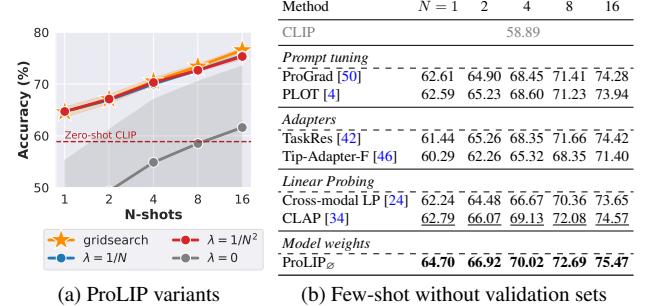


Table 2. **Few-shot without validation set.** (a) ProLIP variants with performance averaged over 11 datasets, 10 runs, and 4 learning rates $LR \in \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$ to study sensitivity. Note the low variance of the parametric formulations ($\lambda = 1/N$, $\lambda = 1/N^2$) which are also reaching performance of the grid search variant — despite having no access to a validation set. (b) Compares to validation-free baselines from [34] and shows the consistent superiority of $\text{ProLIP}_\varnothing$.

hibits resilient generalization capabilities when confronted with out-of-distribution data. Achieving this generalization in a few-shot framework is highly challenging yet important to benchmark when assessing the potential practical use of few-shot methods. While few-shot adaptation is crucial to optimize in-domain performance, such a domain-specific learning process often impairs the generalization capability of the vanilla CLIP model, which indeed remains a competitive baseline in generalization. In the following experiments, we evaluate few-shot methods on domains different from ones used in few-shot training. For CLIP, we use the 7 simple generic templates [30] that report more competitive generalization results than “a photo of a {class}”. Comparison is done only among the few-shot methods; the zero-shot CLIP performance is included as reference. For $\text{ProLIP}_\varnothing$, we systematically use $\lambda = 1/N$ and a fixed LR of 10^{-5} .

Domain generalization. In this setting, the set of classes is fixed in both in-domain and OOD datasets. Following ProGrad, we train ProLIP on ImageNet (IN) as source dataset (with the number of shots $N=4$), and assess it on ImageNet-V2 (IN-V2), ImageNet-Sketch (IN-S), ImageNet-A (IN-A) and ImageNet-R (IN-R). Tab. 3 shows that ProLIP is on par

Method	RN50		RN101		ViT-B/16		ViT-B/32	
	IN	OOD	IN	OOD	IN	OOD	IN	OOD
CLIP	60.34	43.31	61.24	48.71	68.79	59.87	62.00	50.06
LP	41.29	21.19	47.01	28.33	54.70	35.09	46.77	28.81
CoOp	61.34	40.84	63.99	47.48	69.86	58.32	64.74	48.06
CoCoOp	61.04	40.42	63.59	47.34	70.13	58.17	64.63	47.93
ProGrad	62.17	42.23	64.98	48.53	70.45	59.05	65.36	49.39
ProLIP	62.37	43.25	65.13	48.84	70.78	59.92	65.90	50.01
ProLIP \varnothing	62.55	43.20	65.11	48.81	70.94	59.97	66.00	49.86

Table 3. Domain generalization. 4-shot training on ImageNet (source) and evaluation on OOD variants (IN-V2, IV-S, IN-A, IN-R) with different visual backbones. We report accuracy over source ImageNet (IN) along with the average only over OOD variants to show generalization ('OOD'). Baselines are 3 runs averages reported from ProGrad [50]. We report results of two ProLIP variants: the standard ProLIP using few-shot validation set and the validation-free ProLIP \varnothing . All the baselines use validation set.

with or better than other methods both on source and unseen target domains, for both ResNet and ViT CLIP backbones. Our consistent improvements over zero-shot CLIP highlight the merit of ProLIP in preserving generalization while still maximizing in-domain results (IN), whereas the other methods fall short in generalization. The validation-free ProLIP \varnothing performs on-par with ProLIP.

Cross-dataset generalization. This is an interesting generalization setting previously addressed in prompt tuning works [48, 50] where OOD datasets not only come from other domains but may also contain different or more fine-grained classes compared to ones in source. Tab. 4 shows the generalization from ImageNet as source dataset (4-shot) to the 10 other datasets. ProLIP outperforms ProGrad on 6 out of 11 datasets and on average. However, it is worth noting that zero-shot CLIP remains the strongest baseline in this setting. As argued in CoCoOp [48], ImageNet contains 1000 classes, mainly consisting of objects. Dog breeds are also present, so good generalization (or at most small zero-shot performance drop) to datasets like OxfordPets and Caltech101 is expected. However, for datasets presenting a larger gap (e.g., fine-grained and/or specialized datasets), generalization is expected to be lower. For such datasets, like FGVCAircraft and DTD, ProLIP outperforms other adaptation methods, but remains behind zero-shot accuracy. Among all the target datasets, all methods exhibit a significant drop on EuroSAT (26-35% from the zero-shot model). Interestingly, ProLIP not only does not exhibit the same drop, but retains the zero-shot performance. In short, looking at the generalization of ProLIP on each of the 10 datasets, our method is overall retaining zero-shot capability the most and showing better cross-dataset transferability. Again, we observe similar results of the two ProLIP variants, further consolidating the potential of the more straightforward and less experimentally intensive ProLIP \varnothing .

Base-to-new generalization. In this setting, we divide all

Method	Source	Target										
		ImageNet	Caltech101	OxfordPets	StanfordCars	Flowers102	Food101	FGVCAircraft	SUN397	DTD	Eurosat	UCF101
CLIP	60.35	85.84	85.75	55.78	65.98	77.35	17.07	58.85	42.69	36.22	61.80	58.88
CoOp	61.34	84.48	85.99	54.16	60.10	75.48	14.09	57.48	35.32	26.72	57.56	55.70
CoCoOp	61.04	84.73	86.42	52.34	61.24	73.79	13.74	55.94	36.60	23.46	57.97	55.21
ProGrad	62.17	88.30	86.43	55.61	62.69	76.76	15.76	60.16	39.48	24.87	58.70	57.36
ProLIP	62.37	87.08	84.57	54.56	64.79	75.56	16.94	59.78	40.96	36.31	61.24	58.56
ProLIP \varnothing	62.55	86.99	84.00	54.19	64.03	74.95	16.99	59.67	41.09	36.55	60.95	58.36

Table 4. Cross-dataset generalization. Training is performed on 4-shot ImageNet (source). The learned models are evaluated on 10 other datasets (target). Baselines' scores are average of 3 runs reported from ProGrad [50]. We report results of two ProLIP variants: the standard ProLIP using few-shot validation set and the validation-free ProLIP \varnothing .

classes into two groups: base and new classes. Training is performed on base classes and testing on both base and new classes. Moreover, the harmonic mean is reported to assess the trade-off. Here, we simply use the validation-free variant (*i.e.*, ProLIP \varnothing). In Tab. 5, we see that ProLIP \varnothing significantly outperforms ProGrad [50] in *total harmonic mean* across 11 datasets. Additionally, ProLIP \varnothing is competitive to MaPLe [19], which consists of multi-modal prompt learning for ViT architectures.

Metrics details. Previous works [19, 50] calculate the *total harmonic mean* over datasets in two different ways.

We report in Tab. 5 for each architecture both ways of calculating the *total harmonic means*, renaming them H_{t1} and H_{t2} . Our method is superior to baselines, regardless of the total harmonic mean used. We also detail the computation below.

In ProGrad [50], the total harmonic mean over the 11 datasets is computed as *the average harmonic means of individual datasets*. This writes:

$$H_{t1} = \frac{1}{11} \sum_{i=1}^{11} HM_i , \quad (7)$$

$HM_i = 2 \times \frac{\text{acc}_{bi} * \text{acc}_{ni}}{\text{acc}_{bi} + \text{acc}_{ni}}$ being the harmonic mean of dataset i . Here, acc_{bi} and acc_{ni} denote the accuracy on base and new classes for dataset i , respectively.

Instead in MaPLe [19], the total harmonic mean over the 11 datasets is calculated as *the harmonic mean of average base and average new classes accuracies*, being:

$$H_{t2} = 2 \times \frac{\text{acc}_b \times \text{acc}_n}{\text{acc}_b + \text{acc}_n} , \quad (8)$$

where $\text{acc}_b = \frac{1}{11} \sum_{i=1}^{11} \text{acc}_{bi}$ and $\text{acc}_n = \frac{1}{11} \sum_{i=1}^{11} \text{acc}_{ni}$. Per-dataset performance is reported in Appendix B.

5.4. Analyses

Number of augmented views. Following the literature [17, 46], we apply RandomResizedCrop and

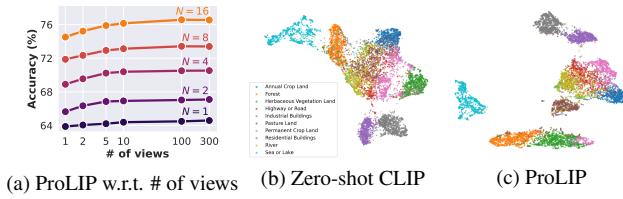
	Base	New	H_{t1}	H_{t2}
CLIP	61.72	65.91	63.64	63.75
CoOp	71.96	61.26	65.58	66.18
CoCoOp	72.23	60.77	65.35	66.01
ProGrad	73.29	65.96	69.06	69.43
ProLIP \varnothing	75.45	69.43	72.12	72.31

(a) ResNet-50

	Base	New	H_{t1}	H_{t2}
CLIP	69.34	74.22	71.59	71.70
CoOp	82.69	63.22	70.83	71.66
CoCoOp	80.47	71.69	75.44	75.83
MaPLe	82.28	75.14	78.27	78.55
ProLIP \varnothing	84.01	73.86	78.28	78.61

(b) ViT-B/16

Table 5. **Base-to-new.** Performance comparison of methods on ResNet-50 and ViT-B/16 architectures across 11 datasets with either H_{t1} (equation 7) or H_{t2} (equation 8).



(a) ProLIP w.r.t. # of views

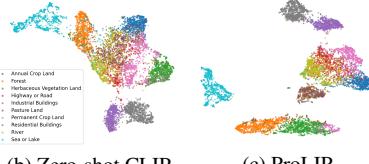


Figure 4. **Ablation and UMAP Visualization.** (a) Ablation of ProLIP using varying number of views and shots. (b) and (c) UMAP of Zero-shot CLIP vs ProLIP on EuroSAT, showing that some classes (e.g., ‘Pasture Land’, ‘Permanent Crop Land’, ‘Sea or Lake’, etc.) are better clustered with our method.

Method	$N = 1$	2	4	8	16
CLIP			58.89		
Full Fine-tuning	46.09 ± 6.33	51.85 ± 5.32	58.06 ± 6.19	62.22 ± 1.23	67.74 ± 0.68
ProLIP \varnothing ($\lambda = 0$)	62.91 ± 0.85	65.97 ± 0.67	69.76 ± 0.55	72.57 ± 0.35	75.73 ± 0.30
ProLIP \varnothing ($\lambda = 1/N$)	64.67 ± 0.63	66.80 ± 0.39	69.73 ± 0.37	72.44 ± 0.33	75.34 ± 0.31
ProLIP \varnothing ($\lambda = 1/N^2$)	64.67 ± 0.63	67.04 ± 0.42	70.27 ± 0.44	72.84 ± 0.32	75.77 ± 0.30

Table 6. **Comparison to full fine-tuning.** We report the classification accuracy (%) averaged over 11 datasets and 10 support sets, along with standard deviation, comparing ProLIP \varnothing to full fine-tuning of the vision encoder.

RandomHorizontalFlip augmentations during training. As mentioned earlier, ProLIP can be applied on pre-computed visual embeddings (before the projection layer). We ablate the number of views in which the features are saved. Fig. 4a shows that average accuracy over 11 datasets increases with more views. Interestingly, ~ 10 views are sufficient to get results close to those with 300 views. In contrast, Lin et al. [24] showed that the gain saturates after more than two views for their cross-modal linear probe.

Comparison to full fine-tuning. We compare ProLIP \varnothing with full fine-tuning of the visual backbone. Results in Tab. 6 show that full fine-tuning is far behind ProLIP \varnothing , and even degrades zero-shot performance for $N = 1, 2$ and 4-shots. The learning rate is 10^{-5} for these experiments, and ProLIP \varnothing is shown for different λ values (including $\lambda = 0$). These results confirm that full fine-tuning faces a high risk of overfitting especially in low-shot regimes, advocating for parameter-efficient fine-tuning methods like ProLIP.

Visualization. We use UMAP to visualize EuroSAT test set feature manifolds, before and after 16-shot training (i.e.,

Method	IN	IN-A	IN-V2	IN-R	IN-S	Average	Avg. OOD
CLIP	60.33	23.79	53.31	60.58	35.46	46.69	43.29
w/o few-shot training on IN							
TPT [33]	<u>60.74</u>	<u>26.67</u>	<u>54.70</u>	<u>59.11</u>	<u>35.09</u>	<u>47.26</u>	<u>43.89</u>
ProLIP _{test-time}	62.00	33.76	56.03	62.69	37.29	50.35	47.44
w/ 16-shot training on IN							
CoOp [33]	63.33	23.06	55.40	56.60	34.67	46.61	42.43
TPT + CoOp [33]	<u>64.73</u>	<u>30.32</u>	<u>57.83</u>	<u>58.99</u>	<u>35.86</u>	<u>49.55</u>	<u>45.75</u>
ProLIP	64.48	22.75	56.24	<u>59.56</u>	34.80	47.57	43.34
ProLIP _{test-time} + ProLIP	66.90	32.96	58.77	61.78	36.97	51.48	47.62

Table 7. **Robustness to natural distribution shifts in test-time adaptation.** Experiments are done with RN50 backbone, without few-shot training on IN (top) and with 16-shot training (bottom).

zero-shot vs. ProLIP). The results are illustrated in Figs. 4b and 4c. We observe that the features are generally better clustered for ProLIP. Confusing categories like *Highway* or *Road*, *Permanent Crop Land* and *Pasture Land* exhibit remarkably better separation for our few-shot adapted model compared to zero-shot. This visualization hints that ProLIP learns better feature manifolds in the few-shot classification setting.

5.5. Test-time ProLIP

In this section, our goal is to show that ProLIP can be applied beyond supervised few-shot CLIP adaptation. Motivated by the risk of “overfitting” the source domain in classic prompt tuning methods [48, 49], Shu et al. [33] pioneered test-time prompt tuning (TPT), aiming to learn adaptive prompts on the fly using a single test image.

TPT background knowledge. TPT aims to learn a context specific to each test image in an unsupervised way. Given an unlabeled test image I_{test} , the prompt is learned by minimizing the average prediction entropy over different augmented views of I_{test} . Moreover, *confidence selection* filters out the augmented views with high entropy predictions, which might lack important information for classification. More details are provided in Appendix C.

Test-time ProLIP. We do not introduce a new way for CLIP test-time adaptation but simply follow the same experimental setting as TPT (i.e., 1-step entropy minimization of averaged prediction probability distribution, confidence selection), although ProLIP optimizes the projection weight matrix W_o instead of the prompt as in TPT. We name this ProLIP variant as ProLIP_{test-time}. Tab. 7 shows that ProLIP_{test-time} yields superior results to TPT on ImageNet and natural distribution shifts, while being one order of magnitude faster to train. For direct comparison, we separate methods that perform 16-shot training on ImageNet. Of note, even without few-shot training, ProLIP_{test-time} still outperforms CoOp and TPT+CoOp. We further advance ProLIP_{test-time} results with 16-shot training.

6. Conclusion

We propose an extremely simple way for efficient adaptation of CLIP based on the model weights. We identify that fine-tuning the last visual projection matrix, which projects the visual embedding into the multi-modal latent space, is a strong alternative for few-shot classification with CLIP. Moreover, we show advantages of including a squared error regularizer: it prevents the drift from pretrained weights and improves robustness to hyperparameter choice, thus making our method a trustworthy candidate for realistic few-shot adaptation. Additionally, we provide corroboration on the competitiveness of ProLIP in few-shot classification, domain generalization, cross-dataset transfer, base-to-new generalization and test-time adaptation, rendering it a potential general framework for further applications.

Limitation. Many recent pretrained models are only accessible via APIs (e.g., GPT, Claude, Gemini), providing only encoder endpoints. Similarly to prompt tuning methods, ProLIP does not apply to such closed models.

Future directions. Our framework can be applied to other foundation models with different modalities, downstream tasks and training objectives [22, 44]. Future research can explore other alternatives for model weights based adaptation, and theoretical investigation on the effect of fine-tuning the last projection matrix.

Acknowledgment. This work was partially funded by French project SIGHT (ANR-20-CE23-0016). It was performed using HPC resources from GENCI-IDRIS (Grants AD011014477R1, AD011012808R3). The authors thank Clément Weinreich for insightful discussion.

Appendix

We provide additional details on few-shot classification in Appendix A, base-to-new generalization in Appendix B and test-time adaptation in Appendix C. Further, we provide additional details on the training of ProLIP in Appendix D and finally a detailed hyperparameter study in Appendix E.

A. Details on few-shot classification

In addition to the average across datasets in Tab. 1, we report the *per-dataset performance* of all methods in Tabs. 14 and 15, reporting for each the average accuracy over 10 seeds (i.e., support sets). ProLIP performs particularly well on DTD, UCF101, StanfordCars, FGVCAircraft and EuroSAT. For some specific settings, e.g., 1-shot DTD, 1-shot EuroSAT, 16-shot StanfordCars, 8 and 16-shot FGVCAircraft, the improvements over state-of-the-art are significant. On the other hand, for datasets like OxfordPets and Food101, where the zero-shot performance is already good, ProLIP and other baselines are outperformed by prompt learning methods (e.g., ProGrad). This can be related to the

relatively lower number of parameters in the latter, making them less prone to overfitting in very low-shot settings; when the number of shots increases, e.g., 8-16 shots, ProLIP and prompt learning perform on par.

Future research may include the zero-shot accuracy on the few-shot training set in the parametric formulation of the regularization loss weight (i.e., λ). That is, the higher the zero-shot accuracy, the smaller should be the distance between the fine-tuned projection matrix and the pretrained one (i.e., higher λ).

B. Details on base-to-new generalization

Per-dataset performance. In addition to average performance reported in Tab. 5, we report in Tab. 12 and Tab. 13 the per-dataset accuracy for base and new classes, as well as the harmonic mean metrics.

C. Details on test-time ProLIP

TPT [33] learns a single prompt for each test image using an unsupervised loss function. Given a test image \mathbf{I}_{test} , the image is augmented N_{views} times using a family of random augmentations \mathcal{A} . Predictions are made for each view, and the training consists of minimizing the entropy of the averaged probability distribution of these predictions:

$$\mathbf{p}^* = \operatorname{argmin}_{\mathbf{p}} - \sum_{i=1}^K \tilde{p}_{\mathbf{p}}(y_i | \mathbf{I}_{\text{test}}) \log \tilde{p}_{\mathbf{p}}(y_i | \mathbf{I}_{\text{test}}), \quad (9)$$

where

$$\tilde{p}_{\mathbf{p}}(y_i | \mathbf{I}_{\text{test}}) = \frac{1}{N_{\text{views}}} \sum_{i=1}^{N_{\text{views}}} p_{\mathbf{p}}(y_i | \mathcal{A}_i(\mathbf{I}_{\text{test}})). \quad (10)$$

In addition, *confidence selection* is used to filter out predictions with high entropy, which are considered as noisy. Self-entropy is computed for each of the N_{views} , a fixed cut-off percentile ρ keeps only predictions with lower entropy than τ . $\tilde{p}_{\mathbf{p}}$ in Equation 9 becomes:

$$\tilde{p}_{\mathbf{p}}(y | \mathbf{I}_{\text{test}}) = \frac{1}{\rho N} \sum_{i=1}^{N_{\text{views}}} \mathbb{1}_{\{H(p_i) \leq \tau\}} p_{\mathbf{p}}(y | \mathcal{A}_i(\mathbf{I}_{\text{test}})). \quad (11)$$

We apply the same framework (i.e., loss function, confidence selection) with the only difference of minimizing Equation 9 over \mathbf{W}_o instead of the prompt \mathbf{p} . For a fair comparison, we use the same number of steps for training (i.e., 1 step) and the same value of the cutoff percentile $\rho = 0.1$. The learning rate is 10^{-4} . Note that, measured on ImageNet, ProLIP is ~ 13 times faster than TPT, as the latter requires backpropagation through the whole text encoder, while in our case backpropagation is limited to the last visual

projection layer and is not applied on the text encoder. We also stress that since we perform only 1 step of training, the regularization loss cannot be used as the first value it takes is 0 (initially the fine-tuned projection matrix is equal to the pre-trained one).

D. ProLIP training details

The text encoder is fully frozen during training of ProLIP. The templates are similar to previous works [17, 46] for fair comparison, and are detailed in Tab. 8 for each dataset.

Dataset	Template
Caltech101	“a photo of a {class}.”
StanfordCars	“a photo of a {class}.”
SUN397	“a photo of a {class}.”
DTD	“{class} texture.”
Eurosat	“a centered satellite photo of {class}.”
FGVCAircraft	“a photo of a {class}, a type of aircraft.”
Food101	“a photo of {class}, a type of food.”
Flowers102	“a photo of a {class}, a type of flower.”
OxfordPets	“a photo of a {class}, a type of pet.”
UCF101	“a photo of a person doing {class}.”
ImageNet	Ensemble of 7 templates:
ImageNet-A	“itap of a {class}.”, “a bad photo of the {class}.”,
ImageNet-V2	“a origami {}.”, “a photo of the large {class}.”,
ImageNet-R	“a {class} in a video game.”, “art of the {class}.”,
ImageNet-Sketch	“a photo of the small {class}.”]

Table 8. **Dataset-specific templates.** Following the literature, all but ImageNet dataset and its variants use a single template.

For training, only the weight matrix \mathbf{W}_o in Eq. (2) and Eq. (3) is fine-tuned. Note that for ResNets, a bias term \mathbf{b}_o exists while for ViTs no bias is added in pretraining. We stress that fine-tuning also the bias term for ResNets does not change the results, as most of the parameters are concentrated in the weight matrix. In detail for ResNet-50, $\mathbf{W}_o \in \mathbb{R}^{D_o \times D}$, where $D_o = 2048$ and $D = 1024$, this makes a total of $\sim 2M$ parameters, while $\mathbf{b}_o \in \mathbb{R}^D$ has only 1024 parameters. Tab. 9 shows the number of trainable parameters in ProLIP for different backbones.

Backbone	$D_o \times D$	Parameters in \mathbf{W}_o
ResNet-50	2048×1024	2.097M
ResNet-101	2048×512	1.049M
ViT-B/32	768×512	0.393M
ViT-B/16	768×512	0.393M

Table 9. **Number of trainable parameters per backbone.** It is the number of elements in the projection matrix $\mathbf{W}_o \in \mathbb{R}^{D_o \times D}$.

E. ProLIP hyperparameters study

Grid search. Fig. 5 shows the distribution of hyperparameters found by grid search on the few-shot validation set (cf. Tab. 1). We draw two observations:

1. The learning rates span a wide range of values, and high values like 10^{-3} and 10^{-2} are selected several times,

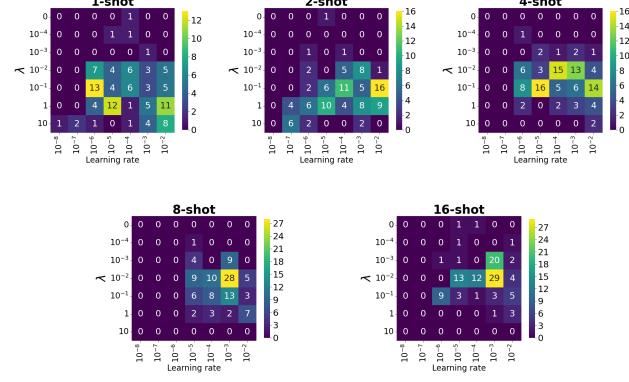


Figure 5. **Hyperparameters selected by grid search.** Learning rates and regularization loss weights λ found with grid search on the few-shot validation set. The distribution of these hyperparameters are shown for each few-shot setting ($N = 1, 2, 4, 8, 16$).

which would cause severe overfitting when no regularization is used (cf. Tab. 11 and Fig. 3).

2. $\lambda = 0$ is rarely selected, meaning that based on the few-shot validation set, regularized projection matrices generalize better.

Hyperparameter sensitivity. Tab. 10 complements Fig. 3, where ProLIP is trained for different fixed learning rates, with fixed regularization loss weight values (i.e., λ). Looking at the values, we make the following observations:

1. For low learning rates (i.e., $10^{-5}, 10^{-6}$), unregularized ProLIP shows good performance for different values of N , demonstrating the effectiveness of simply fine-tuning the last visual projection matrix. However, the performance drops significantly when the LR increases.
2. A higher value of λ works better for fewer training shots N , and vice versa. This effect is increasingly visible when the LR increases. Such observation is expected: with less data we need more regularization as overfitting risk is higher, and is the base for formulating λ as a decreasing function of N (See Tab. 11, which shows the detailed numerical results of Tab. 2a).

Method	$N = 1$	2	4	8	16
CLIP	58.89				
ProLIP (grid search)	64.59 ± 0.98	67.09 ± 0.87	70.53 ± 0.69	73.40 ± 0.45	76.55 ± 0.41
ProLIP, LR= 10^{-6}	$\lambda = 1$	63.05 ± 0.52	65.00 ± 0.40	66.91 ± 0.30	68.13 ± 0.31
	$\lambda = 10^{-1}$	63.90 ± 0.60	66.32 ± 0.41	68.99 ± 0.34	70.96 ± 0.31
	$\lambda = 10^{-2}$	63.94 ± 0.65	66.46 ± 0.42	69.17 ± 0.37	71.30 ± 0.33
	$\lambda = 0$	63.94 ± 0.66	66.46 ± 0.42	69.19 ± 0.37	71.34 ± 0.34
ProLIP, LR= 10^{-5}	$\lambda = 1$	64.67 ± 0.63	66.37 ± 0.35	68.34 ± 0.33	69.63 ± 0.30
	$\lambda = 10^{-1}$	64.82 ± 0.71	67.13 ± 0.45	70.23 ± 0.42	72.57 ± 0.33
	$\lambda = 10^{-2}$	63.64 ± 0.86	66.47 ± 0.63	70.07 ± 0.50	72.79 ± 0.32
	$\lambda = 0$	62.91 ± 0.85	65.97 ± 0.67	69.76 ± 0.55	72.57 ± 0.35
ProLIP, LR= 10^{-4}	$\lambda = 1$	64.78 ± 0.67	66.63 ± 0.41	68.88 ± 0.34	70.33 ± 0.31
	$\lambda = 10^{-1}$	64.65 ± 0.86	67.22 ± 0.66	70.44 ± 0.39	72.91 ± 0.31
	$\lambda = 10^{-2}$	60.54 ± 1.47	64.29 ± 1.16	69.10 ± 0.70	72.51 ± 0.40
	$\lambda = 0$	50.61 ± 1.60	58.36 ± 1.05	65.35 ± 0.67	69.70 ± 0.45
ProLIP, LR= 10^{-3}	$\lambda = 1$	64.75 ± 0.68	66.56 ± 0.46	68.89 ± 0.35	70.35 ± 0.30
	$\lambda = 10^{-1}$	64.32 ± 0.84	66.97 ± 0.59	70.39 ± 0.47	72.92 ± 0.35
	$\lambda = 10^{-2}$	58.58 ± 2.04	63.99 ± 0.97	69.15 ± 0.69	72.69 ± 0.42
	$\lambda = 0$	40.17 ± 1.65	49.08 ± 1.19	56.34 ± 1.08	59.78 ± 0.92
ProLIP, LR= 10^{-2}	$\lambda = 1$	64.61 ± 0.72	66.53 ± 0.45	68.83 ± 0.37	70.29 ± 0.31
	$\lambda = 10^{-1}$	63.47 ± 1.68	66.72 ± 0.74	70.18 ± 0.57	72.66 ± 0.59
	$\lambda = 10^{-2}$	53.29 ± 2.19	61.00 ± 1.27	67.49 ± 0.73	71.62 ± 0.54
	$\lambda = 0$	20.02 ± 2.21	23.91 ± 2.17	28.05 ± 2.43	32.02 ± 1.93

Table 10. **ProLIP sensitivity to hyperparameter choice.** Accuracy of ProLIP to the hyperparameters (learning rate LR and regularization weight λ) for $N \in \{1, 2, 4, 8, 16\}$ -shot settings. Each number is an average over 11 datasets, 10 runs for each.

Method	$N = 1$	2	4	8	16
CLIP	58.89				
ProLIP $_{\emptyset}$, $\lambda = 1/N$	LR= 10^{-5}	64.67 ± 0.63	66.80 ± 0.39	69.73 ± 0.37	72.44 ± 0.33
	LR= 10^{-4}	64.78 ± 0.67	67.02 ± 0.46	70.19 ± 0.36	72.83 ± 0.35
	LR= 10^{-3}	64.75 ± 0.68	66.93 ± 0.49	70.12 ± 0.39	72.84 ± 0.34
	LR= 10^{-2}	64.60 ± 0.72	66.93 ± 0.56	70.03 ± 0.50	72.63 ± 0.48
	Average	64.70 ± 0.08	66.92 ± 0.09	70.02 ± 0.20	72.69 ± 0.19
ProLIP $_{\emptyset}$, $\lambda = 1/N^2$	LR= 10^{-5}	64.67 ± 0.63	67.04 ± 0.42	70.27 ± 0.44	72.84 ± 0.32
	LR= 10^{-4}	64.78 ± 0.67	67.22 ± 0.50	70.42 ± 0.43	72.78 ± 0.38
	LR= 10^{-3}	64.75 ± 0.68	67.12 ± 0.52	70.42 ± 0.52	72.89 ± 0.40
	LR= 10^{-2}	64.60 ± 0.72	67.05 ± 0.55	70.09 ± 0.81	72.19 ± 0.42
	Average	64.70 ± 0.08	67.11 ± 0.08	70.30 ± 0.16	72.68 ± 0.33
ProLIP $_{\emptyset}$, $\lambda = 0$	LR= 10^{-5}	62.91 ± 0.85	65.97 ± 0.67	69.76 ± 0.55	72.57 ± 0.35
	LR= 10^{-4}	50.61 ± 1.60	58.36 ± 1.05	65.35 ± 0.67	69.70 ± 0.45
	LR= 10^{-3}	40.17 ± 1.65	49.08 ± 1.19	56.34 ± 1.08	59.78 ± 0.92
	LR= 10^{-2}	20.02 ± 2.21	23.91 ± 2.17	28.05 ± 2.43	32.02 ± 1.93
	Average	43.43 ± 18.16	49.33 ± 18.30	54.88 ± 18.74	58.52 ± 18.50

Table 11. **ProLIP $_{\emptyset}$ with a parametric λ .** Accuracy (%) of ProLIP $_{\emptyset}$ with fixed learning rate (LR) and λ as a function of N . For each λ value, we report performance for different LRs and averaged across LRs. Numbers are averages over 11 datasets and 10 runs. We highlight **best** and 2nd best for averages across LRs.

Dataset	Method	$N = 1$	2	4	8	16
CLIP						
		60.35				
<i>ImageNet</i>	CoOp [49]	61.19 ± 0.17	61.58 ± 0.40	62.22 ± 0.22	62.87 ± 0.21	63.70 ± 0.13
	PLOT [4]	60.46 ± 0.34	60.73 ± 0.60	61.79 ± 0.39	62.48 ± 0.32	63.08 ± 0.26
	KgCoOp [41]	60.90 ± 0.16	61.44 ± 0.15	62.00 ± 0.11	62.20 ± 0.15	62.43 ± 0.11
	ProGrad [50]	61.58 ± 0.27	62.14 ± 0.13	62.59 ± 0.09	63.04 ± 0.11	63.54 ± 0.18
	CLIP-Adapter [9]	59.82 ± 0.11	59.94 ± 0.05	59.97 ± 0.04	59.98 ± 0.09	61.31 ± 0.39
	Tip-Adapter-F [46]	60.59 ± 0.14	61.42 ± 0.05	62.12 ± 0.06	63.41 ± 0.07	65.06 ± 0.04
	Tip-Adapter-F* [46]	60.98 ± 0.15	61.23 ± 0.11	61.72 ± 0.25	62.84 ± 0.10	64.03 ± 0.12
	Standard LP [30]	22.21 ± 0.31	31.96 ± 0.25	41.48 ± 0.25	49.49 ± 0.16	56.04 ± 0.13
	LP++ [17]	61.18 ± 0.08	61.56 ± 0.14	62.55 ± 0.12	63.76 ± 0.07	64.73 ± 0.05
	ProLIP	61.29 ± 0.13	61.81 ± 0.18	62.37 ± 0.18	63.30 ± 0.11	64.28 ± 0.12
	CLIP	58.85				
<i>SUN397</i>	CoOp [49]	61.79 ± 0.45	63.32 ± 0.47	65.79 ± 0.44	67.89 ± 0.38	70.15 ± 0.32
	PLOT [4]	62.53 ± 0.30	63.87 ± 0.26	65.85 ± 0.48	67.83 ± 0.36	69.90 ± 0.31
	KgCoOp [41]	62.91 ± 0.59	64.38 ± 0.30	66.06 ± 0.37	66.66 ± 0.10	67.68 ± 0.78
	ProGrad [50]	62.79 ± 0.50	64.12 ± 0.55	66.32 ± 0.59	68.33 ± 0.28	70.18 ± 0.27
	CLIP-Adapter [9]	60.78 ± 0.16	61.79 ± 0.23	63.84 ± 0.35	66.26 ± 0.14	67.66 ± 1.05
	Tip-Adapter-F [46]	61.02 ± 0.36	62.15 ± 0.28	63.86 ± 0.19	67.25 ± 0.16	70.94 ± 0.13
	Tip-Adapter-F* [46]	62.58 ± 0.22	63.79 ± 0.13	65.49 ± 0.35	67.43 ± 0.11	69.25 ± 0.16
	Standard LP [30]	32.56 ± 0.40	43.77 ± 0.41	54.49 ± 0.39	61.83 ± 0.30	67.03 ± 0.16
	LP++ [17]	62.47 ± 0.27	64.65 ± 0.25	67.28 ± 0.27	69.34 ± 0.14	71.23 ± 0.07
	ProLIP	62.71 ± 0.46	65.58 ± 0.13	67.68 ± 0.46	69.17 ± 0.07	71.29 ± 0.23
	CLIP	42.69				
<i>DTD</i>	CoOp [49]	42.31 ± 1.86	47.13 ± 1.93	54.06 ± 1.49	59.21 ± 0.91	63.67 ± 0.83
	PLOT [4]	45.82 ± 1.72	51.32 ± 1.61	55.67 ± 1.14	61.38 ± 1.04	65.29 ± 1.05
	KgCoOp [41]	45.46 ± 2.83	50.01 ± 2.71	53.37 ± 0.71	58.38 ± 1.34	62.71 ± 0.92
	ProGrad [50]	44.19 ± 2.38	50.41 ± 1.74	54.82 ± 1.28	60.31 ± 0.99	63.89 ± 0.88
	CLIP-Adapter [9]	43.49 ± 0.68	44.49 ± 1.07	48.95 ± 0.85	57.52 ± 0.67	62.97 ± 0.60
	Tip-Adapter-F [46]	46.92 ± 1.01	48.50 ± 1.08	57.16 ± 0.53	62.38 ± 0.47	65.23 ± 0.82
	Tip-Adapter-F* [46]	47.68 ± 1.43	52.24 ± 0.74	56.09 ± 0.99	61.05 ± 0.71	65.04 ± 0.21
	Standard LP [30]	29.63 ± 1.53	41.19 ± 1.45	51.72 ± 0.57	58.78 ± 0.52	64.56 ± 0.69
	LP++ [17]	46.97 ± 1.37	52.44 ± 0.99	57.75 ± 0.82	62.42 ± 0.53	66.40 ± 0.50
	ProLIP	49.99 ± 2.28	54.93 ± 1.29	59.25 ± 1.03	64.12 ± 0.64	67.69 ± 0.87
	CLIP	85.84				
<i>Caltech10f</i>	CoOp [49]	87.06 ± 1.24	89.14 ± 0.87	90.00 ± 0.63	91.00 ± 0.66	91.77 ± 0.29
	PLOT [4]	89.41 ± 0.41	90.22 ± 0.25	90.69 ± 0.37	91.55 ± 0.38	92.17 ± 0.30
	KgCoOp [41]	88.24 ± 0.49	88.85 ± 0.43	89.89 ± 0.31	90.32 ± 0.43	90.93 ± 0.26
	ProGrad [50]	88.34 ± 1.64	89.01 ± 0.61	90.13 ± 0.45	90.76 ± 0.32	91.67 ± 0.39
	CLIP-Adapter [9]	87.69 ± 0.41	89.37 ± 0.29	90.21 ± 0.32	91.33 ± 0.15	92.10 ± 0.20
	Tip-Adapter-F [46]	87.35 ± 0.64	88.17 ± 0.29	89.49 ± 0.25	90.54 ± 0.34	92.10 ± 0.25
	Tip-Adapter-F* [46]	88.68 ± 0.44	89.36 ± 0.59	90.40 ± 0.26	91.62 ± 0.23	92.63 ± 0.21
	Standard LP [30]	68.88 ± 1.68	78.41 ± 0.54	84.91 ± 0.45	88.70 ± 0.40	91.14 ± 0.19
	LP++ [17]	88.56 ± 0.43	89.53 ± 0.35	90.87 ± 0.19	91.84 ± 0.24	92.73 ± 0.17
	ProLIP	89.16 ± 0.48	89.48 ± 1.15	91.44 ± 0.42	92.43 ± 0.32	93.39 ± 0.38
	CLIP	61.80				
<i>UCF10f</i>	CoOp [49]	62.80 ± 1.26	65.62 ± 1.09	68.69 ± 0.76	72.57 ± 0.80	76.39 ± 0.54
	PLOT [4]	63.22 ± 1.05	66.49 ± 0.92	70.12 ± 0.62	74.63 ± 0.79	77.39 ± 0.53
	KgCoOp [41]	64.37 ± 1.66	64.91 ± 1.01	68.41 ± 0.38	69.86 ± 0.33	71.73 ± 0.78
	ProGrad [50]	65.13 ± 0.87	66.57 ± 0.62	69.80 ± 0.62	73.01 ± 0.52	75.76 ± 0.47
	CLIP-Adapter [9]	64.25 ± 0.54	66.68 ± 0.31	69.77 ± 0.40	73.90 ± 0.50	77.26 ± 0.39
	Tip-Adapter-F [46]	64.28 ± 0.54	65.48 ± 0.43	67.61 ± 0.28	72.05 ± 0.53	77.30 ± 0.21
	Tip-Adapter-F* [46]	65.50 ± 0.59	68.55 ± 0.45	70.55 ± 0.58	74.25 ± 0.48	76.83 ± 0.24
	Standard LP [30]	40.80 ± 1.05	51.71 ± 0.79	61.64 ± 0.50	68.47 ± 0.44	73.38 ± 0.43
	LP++ [17]	65.41 ± 0.37	69.20 ± 0.52	71.68 ± 0.41	74.86 ± 0.36	77.46 ± 0.39
	ProLIP	67.05 ± 0.12	70.02 ± 0.46	71.90 ± 1.01	76.36 ± 0.67	80.29 ± 0.22
	CLIP	65.98				
<i>Flowers102</i>	CoOp [49]	69.00 ± 2.44	78.47 ± 1.88	85.34 ± 1.69	91.68 ± 0.82	94.47 ± 0.36
	PLOT [4]	71.09 ± 1.44	81.22 ± 0.92	87.61 ± 0.79	92.60 ± 0.55	95.18 ± 0.40
	KgCoOp [41]	68.73 ± 1.97	69.63 ± 1.25	76.51 ± 0.51	80.71 ± 0.63	84.48 ± 0.70
	ProGrad [50]	72.16 ± 1.74	79.55 ± 0.88	84.56 ± 1.41	91.73 ± 0.35	94.10 ± 0.41
	CLIP-Adapter [9]	66.86 ± 0.73	69.71 ± 0.46	77.42 ± 0.60	87.20 ± 0.52	91.16 ± 0.23
	Tip-Adapter-F [46]	67.73 ± 0.57	68.18 ± 0.84	71.17 ± 0.67	84.11 ± 0.49	93.02 ± 0.28
	Tip-Adapter-F* [46]	78.46 ± 1.01	85.14 ± 0.81	88.53 ± 0.54	92.33 ± 0.32	94.26 ± 0.38
	Standard LP [30]	56.98 ± 1.56	73.40 ± 0.87	84.38 ± 0.53	91.81 ± 0.34	95.05 ± 0.29
	LP++ [17]	78.21 ± 1.01	84.69 ± 0.70	89.56 ± 0.45	92.61 ± 0.32	94.26 ± 0.24
	ProLIP	76.13 ± 1.35	82.31 ± 1.36	88.37 ± 0.78	92.79 ± 0.61	95.15 ± 0.35

Table 14. Comparison to state-of-the-art methods. Average classification accuracy (%) and standard deviation over 10 tasks for 11 benchmarks. Best values are highlighted in bold.

Dataset	Method	$N = 1$	2	4	8	16
CLIP						
		55.78				
<i>StanfordCars</i>	CoOp [49]	57.00 ± 0.93	58.96 ± 0.78	62.81 ± 0.71	68.40 ± 0.61	72.87 ± 0.50
	PLOT [4]	57.47 ± 0.58	59.89 ± 0.60	63.49 ± 0.80	68.75 ± 0.46	73.86 ± 0.39
	KgCoOp [41]	57.19 ± 0.65	58.94 ± 0.33	59.85 ± 0.51	61.42 ± 0.40	62.99 ± 1.33
	ProGrad [50]	58.63 ± 0.39	61.23 ± 0.65	65.02 ± 0.78	69.43 ± 0.40	72.76 ± 0.45
	CLIP-Adapter [9]	56.67 ± 0.22	57.94 ± 0.27	61.13 ± 0.30	65.43 ± 0.10	70.24 ± 0.79
	Tip-Adapter-F [46]	57.24 ± 0.23	58.12 ± 0.50	59.34 ± 0.20	64.25 ± 0.19	71.38 ± 0.23
	Tip-Adapter-F* [46]	57.85 ± 0.33	60.55 ± 0.34	64.22 ± 0.52	68.75 ± 0.31	74.19 ± 0.30
	Standard LP [30]	22.94 ± 0.61	35.48 ± 0.51	47.49 ± 0.67	59.34 ± 0.30	69.11 ± 0.18
	LP++ [17]	57.20 ± 0.65	59.95 ± 0.36	63.44 ± 0.34	67.81 ± 0.24	72.33 ± 0.18
	ProLIP	58.29 ± 0.49	61.94 ± 0.37	66.09 ± 0.27	69.82 ± 0.43	76.03 ± 0.09
	CLIP	17.07				
<i>FGVCAircraft</i>	CoOp [49]	12.50 ± 6.16	17.59 ± 3.70	21.27 ± 2.54	26.85 ± 0.63	31.20 ± 0.40
	PLOT [4]	17.75 ± 1.36	19.55 ± 0.99	22.26 ± 0.89	26.70 ± 0.70	32.09 ± 0.68
	KgCoOp [41]	18.61 ± 0.65	18.93 ± 1.01	21.16 ± 0.82	22.80 ± 0.44	24.10 ± 0.59
	ProGrad [50]	18.41 ± 0.98	20.51 ± 1.11	23.65 ± 0.50	26.98 ± 0.50	30.47 ± 0.76
	CLIP-Adapter [9]	18.56 ± 0.20	19.18 ± 0.28	21.00 ± 0.21	23.76 ± 0.33	33.37 ± 0.23
	Tip-Adapter-F [46]	18.23 ± 0.19	19.12 ± 0.20	20.55 ± 0.20	23.60 ± 0.29	30.37 ± 0.25
	Tip-Adapter-F* [46]	19.08 ± 0.15	20.79 ± 0.59	23.99 ± 0.57	30.58 ± 0.29	36.16 ± 0.34
	Standard LP [30]	12.66 ± 0.59	16.92 ± 0.56	21.11 ± 0.83	26.53 ± 0.38	32.42 ± 0.54
	LP++ [17]	19.69 ± 0.39	21.58 ± 0.46	24.22 ± 0.60	27.73 ± 0.48	31.73 ± 0.44
	ProLIP	17.86 ± 1.18	20.88 ± 0.69	27.35 ± 0.19	32.59 ± 0.37	40.09 ± 0.12
	CLIP	36.22				
<i>EuroSAT</i>	CoOp [49]	40.36 ± 7.19	56.15 ± 5.82	66.13 ± 3.62	77.02 ± 1.78	82.59 ± 1.00
	PLOT [4]	44.22 ± 9.14	64.19 ± 6.24	69.37 ± 3.26	78.84 ± 1.33	81.76 ± 1.43
	KgCoOp [41]	43.86 ± 9.17	52.92 ± 5.92	59.51 ± 3.46	63.23 ± 3.03	64.04 ± 1.40
	ProGrad [50]	49.37 ± 5.03	65.22 ± 4.01	69.57 ± 2.88	78.44 ± 1.69	82.17 ± 0.98
	CLIP-Adapter [9]	43.00 ± 2.27	48.60 ± 2.76	59.15 ± 2.26	69.92 ± 1.49	75.38 ± 0.78
	Tip-Adapter-F [46]	47.63 ± 2.64	57.62 ± 1.86	69.30 ± 2.41	75.22 ± 1.32	78.59 ± 1.48
	Tip-Adapter-F* [46]	49.27 ± 2.88	65.66 ± 1.39	70.72 ± 2.73	74.66 ± 3.15	78.73 ± 0.81
	Standard LP [30]	48.29 ± 2.95	56.81 ± 2.93	64.99 ± 3.47	74.56 ± 0.98	80.29 ± 0.90
	LP++ [17]	57.23 ± 1.63	61.65 ± 1.66	68.67 ± 2.21	75.86 ± 0.99	80.58 ± 1.05
	ProLIP	65.78 ± 0.45	67.26 ± 0.57	77.03 ± 1.19	80.08 ± 0.48	85.82 ± 0.63
	CLIP	77.35				
<i>Food101</i>	CoOp [49]	75.58 ± 1.29	77.49 ± 0.41	77.93 ± 0.58	78.92 ± 0.19	79.21 ± 0.36
	PLOT [4]	77.46 ± 0.55	77.72 ± 0.26	78.		

References

- [1] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*. Springer, 2006. 3
- [2] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101-mining discriminative components with random forests. In *ECCV*, 2014. 5
- [3] Tom B Brown. Language models are few-shot learners. *NeurIPS*, 2020. 1
- [4] Guangyi Chen, Weiran Yao, Xiangchen Song, Xinyue Li, Yongming Rao, and Kun Zhang. PLOT: Prompt learning with optimal transport for vision-language models. In *ICLR*, 2023. 2, 3, 5, 6, 13
- [5] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *CVPR*, 2014. 5
- [6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 5
- [7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 3
- [8] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *CVPR Workshops*, 2004. 5
- [9] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. Clip-adapter: Better vision-language models with feature adapters. *IJCV*, 2024. 2, 3, 5, 13
- [10] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Taylor & Francis, 2009. 4
- [11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3
- [12] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2019. 1, 5
- [13] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *ICCV*, 2021. 5
- [14] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *CVPR*, 2021. 5
- [15] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *ICML*, 2019. 2
- [16] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *ICLR*, 2022. 2
- [17] Yunshi Huang, Fereshteh Shakeri, Jose Dolz, Malik Boudiaf, Houda Bahig, and Ismail Ben Ayed. LP++: A surprisingly strong linear probe for few-shot clip. In *CVPR*, 2024. 2, 3, 5, 6, 7, 10, 13
- [18] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *ECCV*, 2022. 2
- [19] Muhammad Uzair Khattak, Hanoona Rasheed, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Maple: Multi-modal prompt learning. In *CVPR*, 2023. 3, 5, 7
- [20] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *ICCV Workshops*, 2013. 5
- [21] Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *ICLR*, 2022. 1, 2
- [22] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *CVPR*, 2022. 9
- [23] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *ACL*, 2021. 2
- [24] Zhiqiu Lin, Samuel Yu, Zhiyi Kuang, Deepak Pathak, and Deva Ramanan. Multimodality helps unimodality: Cross-modal few-shot learning with multimodal models. In *CVPR*, 2023. 2, 3, 6, 8
- [25] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv*, 2013. 1, 5
- [26] Sachit Menon and Carl Vondrick. Visual classification via description from large language models. In *ICLR*, 2023. 1
- [27] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *ICVGIP*, 2008. 5
- [28] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *CVPR*, 2012. 5
- [29] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 4
- [30] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 1, 2, 3, 5, 6, 13
- [31] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *ICML*, 2019. 5
- [32] Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. Adapterdrop: On the efficiency of adapters in transformers. In *EMNLP*, 2021. 2

- [33] Manli Shu, Weili Nie, De-An Huang, Zhiding Yu, Tom Goldstein, Anima Anandkumar, and Chaowei Xiao. Test-time prompt tuning for zero-shot generalization in vision-language models. In *NeurIPS*, 2022. 5, 8, 9
- [34] Julio Silva-Rodriguez, Sina Hajimiri, Ismail Ben Ayed, and Jose Dolz. A closer look at the few-shot adaptation of large vision-language models. In *CVPR*, 2024. 2, 5, 6
- [35] K Soomro. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv*, 2012. 5
- [36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 3
- [37] Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *NeurIPS*, 2019. 5
- [38] Zhengbo Wang, Jian Liang, Lijun Sheng, Ran He, Zilei Wang, and Tieniu Tan. A hard-to-beat baseline for training-free CLIP-based adaptation. In *ICLR*, 2024. 3
- [39] Zhixiang Wei, Lin Chen, Yi Jin, Xiaoxiao Ma, Tianle Liu, Pengyang Ling, Ben Wang, Huaian Chen, and Jinjin Zheng. Stronger fewer & superior: Harnessing vision foundation models for domain generalized semantic segmentation. In *CVPR*, 2024. 2
- [40] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010. 5
- [41] Hantao Yao, Rui Zhang, and Changsheng Xu. Visual-language prompt tuning with knowledge-guided context optimization. In *CVPR*, 2023. 5, 13
- [42] Tao Yu, Zhihe Lu, Xin Jin, Zhibo Chen, and Xinchao Wang. Task residual for tuning vision-language models. In *CVPR*, 2023. 6
- [43] Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *ACL*, 2022. 2
- [44] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *ICCV*, 2023. 9
- [45] Jeffrey O Zhang, Alexander Sax, Amir Zamir, Leonidas Guibas, and Jitendra Malik. Side-tuning: a baseline for network adaptation via additive side networks. In *ECCV*, 2020. 2
- [46] Renrui Zhang, Wei Zhang, Rongyao Fang, Peng Gao, Kun-chang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. Tip-adapter: Training-free adaption of clip for few-shot classification. In *ECCV*, 2022. 2, 3, 5, 6, 7, 10, 13
- [47] Zexuan Zhong, Dan Friedman, and Danqi Chen. Factual probing is [mask]: Learning vs. learning to recall. *ACL*, 2021. 2
- [48] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *CVPR*, 2022. 2, 3, 7, 8
- [49] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *IJCV*, 2022. 2, 5, 8, 13
- [50] Beier Zhu, Yulei Niu, Yucheng Han, Yue Wu, and Hanwang Zhang. Prompt-aligned gradient for prompt tuning. In *ICCV*, 2023. 2, 3, 5, 6, 7, 13