PROMPT GENERATION NETWORKS FOR EFFICIENT ADAPTATION OF FROZEN VISION TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large-scale pretrained models, especially those trained from vision-language data have demonstrated the tremendous value that can be gained from both larger training datasets and models. Thus, in order to benefit from these developments, there is renewed interest in transfer learning and adapting models from large-scale general pretraining to particular downstream tasks. However, the continuously increasing size of the models means that even the classic approach of finetuning is becoming infeasible for all but big institutions. Prompt leaning has emerged as a flexible way to adapt models by solely learning additional inputs to a model that is kept frozen, but so far performances remained inferior to finetuning. To address this, we propose the Prompt Generation Network (PGN) that generates input-dependent prompts by sampling from a learned library of tokens. We show the PGN is effective in adapting pretrained models to various new datasets. It surpasses previous prompt-learning methods by a large margin and even full-finetuning on 5 out of 12 datasets while requiring 100x less parameters. PGN can even be used for training and inferring on multiple datasets simultaneously and learns to allocate tokens between domains. Given these findings, we conclude that PGN is a viable and scalable approach for downstream adaptation of frozen models.

1 INTRODUCTION

Large-scale pretrained models, such as those obtained from self-supervised or image-text training have shown remarkable performance gains for various visual tasks. Particularly models such as CLIP (Radford et al., 2021) and ALIGN (Jia et al., 2021) have demonstrated that large and diverse multi-modal datasets can yield models that exhibit novel abilities in robustness and few- and zero-shot learning. However, the requirements in terms of dataset size and compute infrastructure are exceedingly prohibitive, such that these models are and will likely remain limited in terms of their diversity and availability. Therefore, in order to still benefit from the capabilities of such models, multiple approaches have recently been developed.

The classic approach of finetuning the whole or parts of the model has been carried over to these models, with innovations such as ensembling (Wortsman et al., 2022a), or limited finetuning Cai et al. (2020); Jia et al. (2022). Yet, both the transfer learning paradigm, as well as these novel approaches entangle adaptation and computation phase, reducing the applicability in production settings or where the pretrained model is optimized in hardware. Furthermore, these approaches also face the risk of catastrophic forgetting (Kirkpatrick et al., 2017), whereby models potentially unlearn multiple useful properties stemming from the large-scale training.

An alternative strategy that has recently gained more importance is learning only new inputs to the frozen models, called *prompts*. These are typically learned per domain and downstream task and have shown promising results for adapting to image domains (Bahng et al., 2022) or to video (Ju et al., 2022; Luo et al., 2022). However the resulting performance often fall short compared to finetuning.

In this paper we argue that the main reason for this shortfall is the arbitrary definition of a domain and with it, a limited modeling capacity, as prompts are shared and always used as inputs. In contrast, we propose a new method that allows to adapt to every single input image via a Prompt Generation Network (PGN). This model learns to generate prompts by combining items from a common learned

library of tokens, which allows for efficient, yet flexible modeling capabilities. Furthermore, as these prompts' purpose is to only aid the large-scale model, the PGN can be kept lightweight. On a benchmark covering 12 datasets, we show that our PGN approach achieves performances that can match those of fully finetuning models, while being two orders of magnitude more efficient in terms of additional parameters.

Overall, this paper makes three main contributions:

- We develop a simple, yet effective framework for learning input-dependent visual prompts via a Prompt Generation Networks that combines items from a Token Library
- We provide extensive ablations and comparisons that demonstrate the generalizability of the proposed method across 12 datasets, architectures and settings such as multi-dataset inference
- Finally, via quantitative and qualitative analyses, we showcase how our method allows a "division of labor" between the frozen vision transformer and the PGN

2 Related Works

2.1 Large-scale pretrained models

Multi-modal Pretraining. While alignment of modalities as a learning signal has been proposed early on (de Sa, 1993), Srivastava & Salakhutdinov (2012) were the first to train a Deep Boltzmann machine on unified image-text representations for image classification. Learning joint representations was further popularized in the context image captioning, using an encoder-decoder pipeline with a common embedding space for the image and text features (Kiros et al., 2014). Most modern works in vision-language modeling are inspired by the recent trend of large-scale pretraining, involving data-hungry transformer architectures (Vaswani et al., 2017) fed by web-scale datasets. Two recent approaches are CLIP (Radford et al., 2021) and ALIGN (Jia et al., 2021), which use a contrastive objective to jointly optimize an image and text encoder to align the embeddings and are trained on datasets with hundreds of millions of image-text pairs.

Visual-only Pretraining. Another source for obtaining strong visual encoders is to apply self-supervised learning on visual inputs. Here, typically a pretext task such as clustering (Caron et al., 2018; Asano et al., 2020) or contrastive discrimination (Wu et al., 2018; He et al., 2020; Chen et al., 2020) is used to train on large datasets without any annotations. More recently, DINO (Caron et al., 2021), based on training an L2 loss between a fast-updated student network and a slowly-updated teacher network, has shown the potential for training Vision Transformers architectures (Dosovitskiy et al., 2020) using self-supervision.

2.2 Adaptation Methods

While pretrained vision models generally need to be finetuned to any specific downstream task, vision-language models can perform zero-shot transfer by prompting the text encoder to perform downstream tasks. However, for most datasets there is still a significant performance gap with respect to full-finetuning, which is computationally expensive and yields models that lack robustness against distribution shift. A range of alternative methods have been proposed to address these issues.

Adapters and partial finetuning. Inspired by previous works in NLP (Houlsby et al., 2019; Pfeiffer et al., 2020), lightweight feature adapters (made of trainable parts in between frozen ones) have shown performance gains (Gao et al., 2021; Zhang et al., 2021). Requiring very few additional parameters, only finetuning the bias parameters of pretrained models has also been proposed Zaken et al. (2021); Cai et al. (2020). In contrast, (Jia et al., 2022) learns both additional inputs to multiple layers in a pretrained vision transformer and also finetunes a linear classifier on top. Another finetuning-based approach proposed in (Wortsman et al., 2022b) is to ensemble the weights between zero-shot and finetuned models and (Zhang et al., 2020) which trains additional networks that are fused via summation. Specific to vision-language models, (Zhou et al., 2022b) learn an adaptation network between CLIP's vision and text encoders.

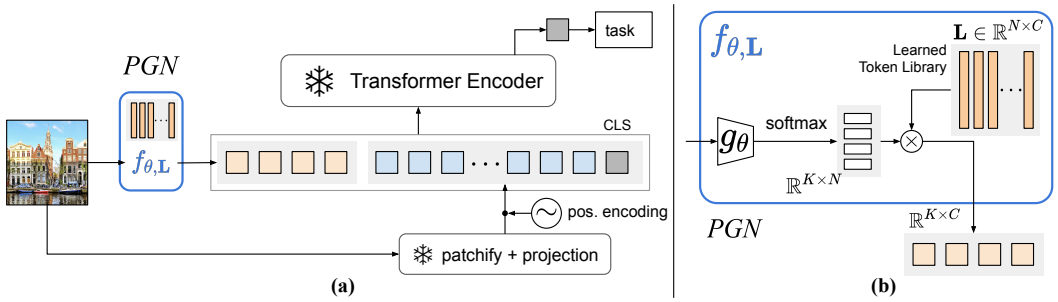

Figure 1: We propose the **Prompt Generation Network (PGN)**, a simple yet effective method that generates prompts conditioned on the input images that benefits the domain adaptation process, while keeping the whole pretrained transformer frozen. **(a)**: the overall prompt learning pipeline including PGN, denoted by $f_{\theta,\mathbf{L}}$: the learned prompt vectors are fed into the pretrained transformer encoder together with image patches for the task or domain of interest. **(b)**: the detailed structure of PGN, a lightweight neural network g_θ learns probability distributions to select multiple prompt vectors from a Token Library $\mathbf{L}$.

While our goal is also to adapt a pretrained model, we aim to separate the adaption stage from the computation stage. This has advantages when the frozen model is either hard to interfere with (e.g. application specific integrated circuits, or served via an API) or sensitive to changes (e.g. due to quantization).

Input prompt learning and reprogramming. Optimizing solely in the *input-space*, prompt learning originates from language modeling in NLP and is a lightweight approach to adapting a model to perform a downstream task (Radford et al., 2019; Brown et al., 2020; Li & Liang, 2021). For computer vision, initial efforts focused on learning continuous prompts (as opposed to prompt sentences/words) for the text encoder of pretrained vision-language models to perform image (Zhou et al., 2022a; Yao et al., 2021) and video tasks (Ju et al., 2022).

Most related to our work is the concurrent work of Bahng et al. (2022) that proposes visual prompting through the addition of learnable pixels, effectively learning prompts in the data space. This is the same setting as adversarial reprogramming (Elsayed et al., 2018; Kloberdanz et al., 2021), where pretrained CNNs are repurposed to classify images from different datasets, or even achieve entirely different goals such as counting. The reprogramming is restricted to transformations of the input, making it distinct from classic transfer learning.

Although these works show significant gains, they do not match classic model adaptation techniques like finetuning and linear probing. We aim to bridge the gap by learning more flexible input prompts that, unlike previous works, are adapted to each image.

3 Methods

The key idea of our method is to generate input-dependent visual prompts that benefit the domain adaptation process. In this section, we first briefly review the recent prompt learning methods in Section 3.1, then introduce the proposed Prompt Generator Network in Section 3.2.

3.1 Review of prompting learning methods

In NLP, prompt learning offers a lightweight approach to tune large-scale pretrained models for performing downstream tasks. Let $\Phi_T(.)$ denote the pretrained language model and $\mathbf{x}_T = [a_1, a_2, ..., a_n]$ denote the input language tokens. Traditionally, the model is trained to produce a latent language embedding for the given input as $\mathbf{z}_T = \Phi_T(\mathbf{x}_T; \texttt{CLS})$, where ';' denotes concatenation and `CLS` is the special token for classification. The latent embedding $\mathbf{z}_T$ can be used for downstream tasks. In order to adapt the model on different tasks or different datasets, one would have to finetune the large-scale pretrained model Φ_T, which is potentially very resource demanding and could be infeasible for models with billions of parameters.

As its name suggests, prompt learning provides a set of learnable vectors $\mathbf{h}_T = [h_T^1, ..., h_T^k]$ called prompt vectors, that are fed to the pretrained model and encourage it to produce desirable outputs. Formally, the prompt learning process can be written as

$$\hat{\mathbf{z}}_T = \Phi_T(\mathbf{h}_T; \mathbf{x}_T; \texttt{CLS}), \tag{1}$$

The prompted language embedding $\hat{\mathbf{z}}_T$ can also be used for downstream tasks. Due to the flexibility of the prompt learning method, one can adapt the model on new tasks or new datasets by only training the lightweight prompt vectors $\mathbf{h}_T$, rather than finetuning the heavy pretrained model Φ_T. This method was originally proposed in the NLP community (Li & Liang, 2021; Lester et al., 2021), and it was later used to prompt the language branch in pretrained visual-language models Ju et al. (2022); Zhou et al. (2022a).

Recently, the prompt learning technique has also been applied to large-scale pretrained *visual* models. Similarly, let $\Phi_V(.)$ denote the pretrained visual model – typically a variant of the Vision Transformer (Dosovitskiy et al., 2020), $\mathbf{x}_V = [E(c_1), E(c_2), ..., E(c_m)]$ denote the encoded input image or video patches, and $\mathbf{h}_V = [h_V^1, ..., h_V^k]$ denote the visual prompt vectors. Visual prompt learning can be formalized as

$$\hat{\mathbf{z}}_V = \Phi_V(\mathbf{h}_V; \mathbf{x}_V; \texttt{CLS}). \tag{2}$$

The previous works (Jia et al., 2022; Bahng et al., 2022) apply visual prompt learning in this way. Note that the prompt vectors $\mathbf{h}_V$ in (Jia et al., 2022) are in the feature space whereas those in (Bahng et al., 2022) are in the pixel space.

3.2 Prompt Generator Network

Although learning prompt vectors brings flexibility to the pretrained model, a key limitation of the classic prompt learning method is that the prompt vectors are shared within the dataset and task. In other words, the prompt vectors are conditioned on the domain. However, what exactly constitutes a domain is somewhat arbitrary, *e.g.*, ImageNet both contains fine-grained distinctions (such as different dog breeds) and very course ones (such as mushroom). Having a single set of learned vectors for adaptation therefore results in modeling capacity being wasted on items that might already be well encoded. In this section we introduce our Prompt Generator Network as a more flexible alternative to this.

Fig. 1 shows an overview of our method. Given the input image $I_i \in \mathbb{R}^{3\times H\times W}$ and the pretrained vision model Φ_V, we first cut the image to $s \times s$ patches $\{c_1, c_2, ..., c_{H'\times W'}\}$, where $c_j \in \mathbb{R}^{2\times s\times s}$, $H' = H/s$ and $W' = W/s$, and then encode with a linear layer $E(.)$ To construct the visual inputs $\mathbf{x}_V = [E(c_1), E(c_2), ..., E(c_{H'\times W'})]$. Rather than introducing a set of *shared* prompt vectors as described in Section 3.1, we propose to use a set of *input-dependent* prompt vectors. Formally, we use a function $f(.)$ to learn the dependency

$$\hat{\mathbf{h}}_V^i = f(I_i), \tag{3}$$

where the function $f(.)$ is typically learned by a neural network. While it is possible to directly transform the inputs continuously to the prompt vectors, in practice this results in a high number of parameters due to the fully-connected layers being proportional to the large input dimensionalities of transformers.

Instead, we propose to use a Token Library $\mathbf{L} \in \mathbb{R}^{N\times C}$ that consists of N learnable feature vectors with channel dimension C. Compared to works that use memory (Sukhbaatar et al., 2015; Han et al., 2020; Kumar et al., 2016), the Token Library's purpose is not to learn characteristics of the dataset, but instead to steer the pretrained model towards a certain dataset, therefore saving modeling capacity. The prompt generation network learns to generate K prompt vectors $\hat{\mathbf{h}}_V = [\hat{\mathbf{h}}_V^1, ..., \hat{\mathbf{h}}_V^K]$, each of which is a combination of the feature vectors from the Token Library $\mathbf{L}$. In detail,

$$\hat{\mathbf{h}}_V^i = f_{\theta,\mathbf{L}}(I_i) = \text{Softmax}(g_\theta(I_i)) \cdot \mathbf{L} \in \mathbb{R}^{1\times C}, \tag{4}$$

where the function $g_\theta(.)$ learns a mapping $\{\mathbb{R}^{3\times H\times W} \mapsto \mathbb{R}^{1\times N}\}$, learned by a lightweight neural network parameterized with θ. Finally all prompt vectors are fed into the frozen transformer encoder,

$$\hat{\mathbf{z}}_V = \Phi_V(\hat{\mathbf{h}}_V; \mathbf{x}_V; \texttt{CLS}). \tag{5}$$

The prompted visual features $\hat{\mathbf{z}}_V$ act as input to a classifier for downstream classification tasks.

4 EXPERIMENTS

4.1 IMPLEMENTATION DETAILS

Datasets. In our experiments we cover 12 public image datasets: CIFAR100 & CIFAR10 (Krizhevsky et al., 2009), Oxford Flowers (Nilsback & Zisserman, 2008), Food101 (Bossard et al., 2014), EuroSAT (Helber et al., 2019), SUN397 (Xiao et al., 2010), UCF101 (Soomro et al., 2012), SVHN (Netzer et al., 2011), Oxford-IIIT Pets (Parkhi et al., 2012), DTD (Cimpoi et al., 2014), RESISC (Cheng et al., 2017) and CLEVR (Johnson et al., 2017). Out of these, we chose CIFAR100 and SUN397 for ablations in Section 4.3, as they vary in resolution, difficulty and the degree of spatial distribution of relevant image features. Please refer to Appendix for the statistics for each dataset.

Architectures. We use the pretrained CLIP (Radford et al., 2021) weights (ViT-B-32) and keep it frozen during our experiments. In the PGN, we experiment with the lightweight ResNet-based architectures (He et al., 2016), namely ResNet10 and ResNet18. We obtain the ResNet10 architecture by reducing the layers at each residual block from ResNet18 and also reducing the depth of the first layer to 16. Please refer to Appendix for the architectural details.

Training details. We train the PGN with a learning rate of 0.1 and apply a cosine decay learning schedule ending at zero learning rate with a linear warmup for the first 50 epochs. We use SGD optimizer with 0.9 momentum. Except when specified, we use a batch size of 128 images on one Nvidia-1080TI GPU, and the network is trained for 500 epochs by default in Section 4.2 and 4.3. Inspired by the concurrent work (Bahng et al., 2022) that trains for 1500 epochs, in Table 5 we train the network for 1000 epochs and we find extra epochs do not bring significant performance gain.

4.2 PROOF OF PRINCIPLE

Table 1: **Comparison of prompting methods.** We show performances of prompting CLIP ViT-B32 using input independent prompts (IIP), Visual Prompts (VP) (Bahng et al., 2022) and our PGN. The top row shows the supervised performance of the PGN backbone. We find that our PGN outperforms existing methods with the help of a light-weight and otherwise not strong backbone.

Method	Accuracy
ResNet-10 superv.	63.7
CLIP + TP	63.1
CLIP + TP + IIP	73.5
CLIP + TP + VP	75.3
CLIP + TP + PGN (Ours)	79.3

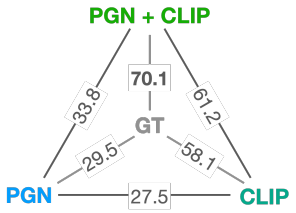

Figure 2: **Feature similarities.** We show the mutual information between the clustered representations of the various components and the ground-truth labels (details in Appendix). PGN learns very different embeddings to CLIP but when combined achieves strong alignment with the ground-truth labels.

In this section we investigate the basic feasibility of our proposed method for generating input-dependent prompts via PGNs. For this, in Table 1, we compare the results of prompting a CLIP ViT-B32 model to correctly select the right text prompts (TP) that are automatically generated as "This is a photo of a [class name]". We observe that compared to the zero-shot baseline of CLIP, training a common set of input independent tokens (IIP) either directly or as pixel based visual prompts (VP) already adds around 10% in terms of performance. Despite these large gains, our proposed method further surpasses these by 4-5%. In the first row we also show that the supervised performance of our light-weight ResNet10 which we use as the backbone to the PGN on its own is relatively weak, and does not even surpass the basic CLIP+VP baseline. Yet, when utilised in our method the two models symbiotically increase the performance to beyond each individual one.

This is further demonstrated in Fig. 2, where we compare the similarities of the representations of the various components. For this we cluster the outputs of each visual encoder (PGN, CLIP and

Table 3: **Ablations** for our PGN method. We vary the number of prompts provided to the frozen model, the size of the Token Library and the PGN backbone. Default settings are in gray.

(a) Number of prompts.

Num.	CIFAR100	SUN397
1	75.8	68.3
4	77.8	69.6
8	77.9	**70.6**
16	**78.3**	70.0
64	77.3	68.9

(b) Size of the token library.

Size	CIFAR100	SUN397
8	75.3	69.0
16	76.4	69.6
64	77.9	70.6
256	78.4	**70.6**
1024	**78.6**	69.9

(c) PGN backbone.

Model (resolution)	CIFAR100	SUN397
IIP	72.1	69.6
MLP (10×10)	73.5	69.6
ResNet10 (64×64)	77.9	70.6
ResNet10 (224×224)	**79.3**	**70.6**
ResNet18 (224×224)	**79.3**	69.6

CLIP+PGN) unsupervisedly and measure the pairwise alignment using normalised mutual information (see Appendix for details). We find that PGN+CLIP's features are closest to the ground-truth (NMI of 70.1%) compared to both PGN (29.5%) and standalone CLIP (58.1%), confirming the results from Table 1. The low performance of only the PGN also demonstrates that it is not the PGN that is doing the "heavy-lifting", as its outputs are least aligned to the ground-truth labels. Instead, the fact that PGN and CLIP's outputs share the lowest similarity of only 27.5%, the superior performance of the combined model might be a result of the PGN learning features that are quite dissimilar and therefore more informative to the frozen CLIP model.

4.3 Ablation Studies

Next, we ablate and analyze the various choices of the PGN architecture and setting. We conduct ablation studies on CIFAR100 and SUN397 in Table 3. Unless otherwise stated, we learn 8 prompts and a Token Library of size 64 to speed up computations.

Number of Prompts. In Table 3a, we vary the number of prompts that the vision transformer receives as output from the PGN. We find that increasing the number of prompts generally yields a better performance, yet, the gains decrease when moving beyond 4 prompts. This shows that even a modest number of 4-8 additional tokens – when added to the 49 spatial plus 1 `CLS` token – can yield significant benefits.

Token Library Size. Next, in Table 3b, we compare the size of the Token Library's number of items. We find that larger Token Library generally leads to better performance, although no further improvement is observed beyond 256 tokens. We conjecture the additional library items beyond 256 tokens only learn redundant information, therefore do not benefit the performance.

Architectures. In Table 3c, we ablate the backbone used for the PGN with input-independent prompts (IIP), where the prompts are simply fixed items in the Token Library. With this as a baseline, we compare against input-dependent prompting methods including a 2-layer MLP operating on the center 10×10 pixels of an image, a ResNet10 at resolutions of 64×64 and 224×224, as well as a heavier ResNet18 at resolutions of 224×224. First, we observe that even a simple model such as a 2-layer MLP obtains a small performance benefit over the IIP. This might be explained by IIP being a strict subset of the input-dependent methods, as these could zero-out the input and instead supply constant prompts. The benefits of input-dependency is further demonstrated by using convolutional neural networks as backbones with gains of up to +5.8% for CIFAR100 (77.9 vs. 72.1). Increasing input resolution from 64 to 224 also slightly improve +1.4% accuracy for CIFAR100. Finally, we observe that the overall performance saturates even when using a ResNet18 compared to the ResNet10, suggesting that the PGN backbone can be lightweight and that the method automatically divides the labor between fine-grained recognition and providing the right prompts as cues.

Token Library vs Direct Learning. In Fig. 3 and Table 2 we study the approach of obtaining the prompts. We compare our token library (TL) with the more direct approach that obtains the prompts through a linear transformation of the image features. While from Table 2, we observe that both methods can be made to achieve similar performances, Fig. 3 clearly demonstrates the superiority of the TL in terms of parameter efficiency.

Scaling to different frozen transformer models. Finally, in Table 4, we explore the use of different pretrained ViT backbones, from supervised training (Dosovitskiy et al., 2020) and from

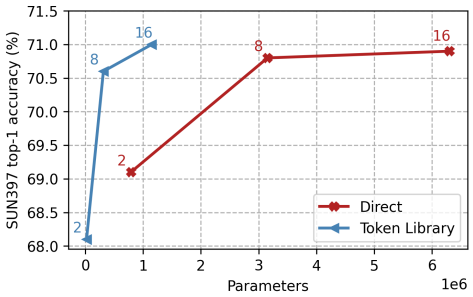

Figure 3: **Parameter efficiency.** We vary the number of generated tokens. A Token Library achieves higher performances at significantly less parameters compared to obtaining prompts directly. The library size used here is $8\times$ the number of prompts.

Table 2: **Similar performances.** The Token Library (TL) achieves comparable results compared to direct generation of prompts, but only requires a fraction of the added parameters. Reported results are obtained for 8 prompts and a token library of 64.

	Direct	**TL**
CIFAR100	79.0	79.3
SUN397	70.8	70.6
#parameters	*3.2M*	*0.3M*

Table 4: **Different pretrained models.** We compare using PGN with two other ViT-B32 backbones from self-supervised DINO (Caron et al., 2021) and from supervised training Dosovitskiy et al. (2020). Best results for each frozen backbone bolded.

	DINO		ViT sup.	
	IIP	PGN	IIP	PGN
CIFAR100	13.2	**53.0**	18.8	**50.7**
SUN397	2.6	**14.2**	2.3	**3.2**
EuroSAT	85.8	**94.6**	91.9	**96.1**

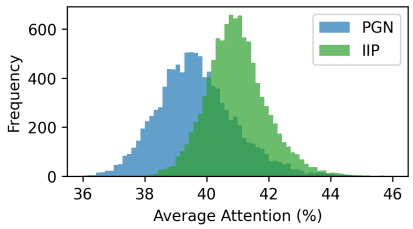

Figure 4: **Attention value histograms.** We show the individual attention values from the CLS token to the supplied prompts of PGN and IIP. We find that while PGN has an overall lower average attention, the input-dependency successfully yields a wider distribution in adapting the original model.

self-supervised training using DINO (Caron et al., 2021). First, we find that generally the performances are lower compared to those obtained from the CLIP backbone in Table 5. This means that adapting these networks are more challenging, potentially owing to the vastly larger pretraining dataset of CLIP. Second, we find that both IIP and PGN struggle with adapting the backbones to the SUN397 dataset. This might be explained by: (i) the difference in image contents – while SUN contains scene images, the pretraining datasets of the supervised ViT and DINO are strongly object-centric (Kuznetsova et al., 2020) – and (ii) the relatively few image per class (≈190). Third, we find that the PGN approach vastly outperforms the baseline IIP approach in adapting these models to the datasets, *e.g.*, showing gains of 40% for CIFAR100.

4.4 Large-scale comparisons

Table 5: **Performance across 12 datasets.** We compare using CLIP with text prompting (TP), visual prompting (VP), linear or full-finetuning (ft.) and our PGN method. The green shade indicates cases where PGN outperforms or matches full-finetuning performance the bolding refers to best performance on a given dataset. We also report the additional number of parameters of the visual encoder for these 12 experiments for each configuration.

Method	CIFAR100	CIFAR10	Flowers	Food	EuroSAT	SUN	UCF	SVHN	Pets	DTD	RESISC	CLEVR	Avg.	Σ params
CLIP+TP	63.1	89.0	61.9	79.8	40.0	60.0	59.9	5.1	85.9	43.0	42.4	20.2	54.2	-
+ VP	75.3	94.2	70.3	78.9	96.4	60.6	66.1	88.4	85.0	57.1	84.5	81.4	78.2	0.94M
+ PGN (ours)	79.3	**96.1**	94.0	82.5	**98.0**	70.9	77.6	94.2	**91.5**	71.5	92.1	**95.1**	**86.9**	12.4M
Linear ft.	80.0	95.0	96.9	**84.6**	95.3	**75.0**	**83.3**	65.4	89.2	**74.6**	92.3	66.0	83.1	0.75M
full-ft.	**82.1**	95.8	**97.4**	80.5	97.9	64.0	80.9	**95.7**	88.5	72.3	**93.3**	94.4	**86.9**	1032M

We next compare prompt learning with PGNs to other domain adaptation techniques on a wide range of vision datasets. Based on the findings of our ablation study, we choose the ResNet10 as the

PGN's backbone, outputting 8 prompts from a Token Library of size 256 feeding into a pretrained CLIP ViT-B/32 image encoder. The results are shown in Table 5 and are directly comparable to the concurrent work of Bahng et al. (2022) on visual prompts (VP), from which we adapted the results of the first three rows. From Table 5, we first find that both linear finetuning, full-finetuning and our method achieve the best performances on 4 out of 12 datasets each. However, when comparing the overall averaged accuracies, we find that our method achieves 86.9%, matching the performance of the full-finetuning adaptation, and surpassing both VP and linear ft by 8-3%. In the last column of Table 5, we show the number of additional parameters required for adapting to each dataset. From this we find that our PGN method requires almost two orders of magnitude less parameters than full-finetuning, while retaining the same overall performance.

4.5 Efficient Multi-dataset PGN

Table 6: **Training a multi-dataset PGN.** Here we show the result of adapting and inferring jointly (J) over multiple datasets compared to individual (I), per-dataset training (for PGN) and evaluation. While giving more text prompts (TP) for joint inference leads to a strong decrease in accuracy, joint training of the PGN retains a strong performance while reducing the number of additional parameters by 75%.

Method	EuroSAT	UCF	Pets	RESISC	Avg.	Δ	Σ params
CLIP+TP (I)	40.0	59.9	85.9	42.4	57.1	-7.7%	-
CLIP+TP (J)	4.4	59.7	85.8	47.7	49.4		-
+ PGN (I)	98.0	77.6	91.5	92.1	89.8	-3.7%	5M
+ PGN (J)	96.9	72.7	89.0	85.7	86.1		1M

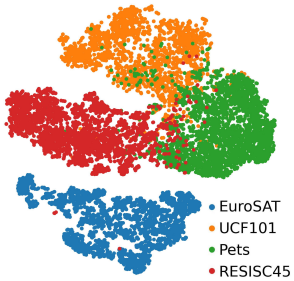

Figure 5: **Automatic domain discovery.** t-SNE visualisation of PGN outputs. PGN trained on a mixture of datasets automatically allocates the tokens in a manner that recovers the individual training domains.

Encouraged by the large-scale comparisons of the earlier section, we investigate whether PGNs can be made even more efficient. For this, we train a PGN on multiple datasets at the same time. More specifically, we retain the same setting as in Section 4.4 and train it with batches that contain samples from the four datasets in Table 6. The model is thus forced to allocate the token library items in a manner that best supports this task and the overall number of additionally adapted parameters is reduced by 75%. From Table 6, we find that despite this reduction in parameters, the overall performance only decreases by a small amount of 3.7%, despite the fact that the classification problem is now 193-way and thus much more difficult.

In Fig. 5 we show a t-SNE visualisation of the CLIP+PGN model that was trained jointly on four datasets. First, we find that the PGN learns features that are well-correlated with the domains. This is particularly interesting as the model did not receive any signal with regards to individual items' dataset origin and shows that such a procedure could be used for adapting to mixed domain or unsorted datasets. Second, we observe that some domains contain images that have similar PGN outputs, notably between UCF and Pets. These are potentially explained by overlaps in depicted objects, *e.g.*, UCF contains classes such as "walking a dog", while Pets contains multiple dog classes.

4.6 Qualitative Analysis

From Table 1, we observed that the CLIP zero-shot and the PGN backbone model's performance on their own are low with 63-64%. However, when combined, we reach performance increases of +15% yielding up to 79% on CIFAR100. In this section, we analyse how the simple mechanism behind PGN is allowing the combined model to achieve superior performances.

What do the individual Token Library items stand for? First we analyse what the individual learned items in the Token Library stand for. For this we pass the validation sets through the trained PGN model and pick individual tokens that we wish to visualize. We then pick the top four input samples that have the highest softmax values for the selected item. The result is shown in Fig. 6 for three datasets. We find that while some tokens are fairly category specific, such as those for a

Figure 6: **Token Library example items.** We show 4 samples that maximally activate one of three selected items in the token library for three datasets. Each row in the grid corresponds to one token library item. We find that the items can stand for whole objects such as apples and trees for CIFAR100, and also for lower level features such as light warmth or net structures as in UCF101.

tree or an apple, some cover much broader categories such as lighting conditions or even geometric structures. Note however that the PGN is not doing the heavy-lifting in terms of classifying the images by itself, as its output is not well-aligned with the ground-truth, as demonstrated in Fig. 2. It rather supplies the frozen transformer model with orthogonal information that helps the task.

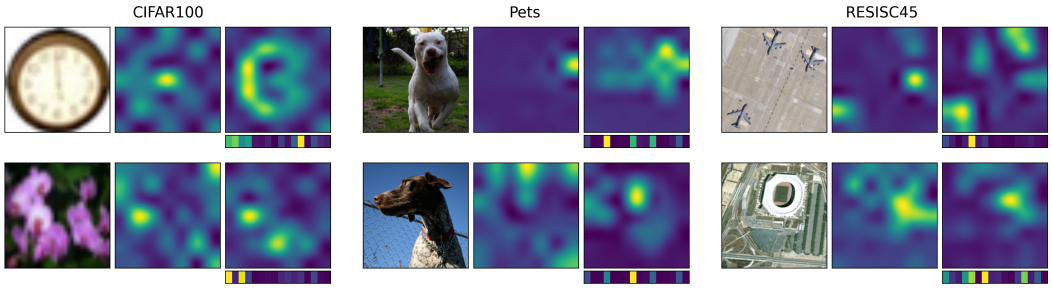

Figure 7: **Modification of `CLS` attention maps.** We show the attention map of the `CLS` token for various inputs (left) with the spatial patches for both the original CLIP model's (middle) and the PGN-modified CLIP's final layer (right). Below the PGN attention map, we show the attention to PGN's additional prompts. We observe both the modification of the attention map as well as the diverse activation patterns of the supplied tokens.

How is the computation changed by PGN prompts? Next, we analyse the effect of the PGN prompts to the internal computations of the frozen vision transformer. In Fig. 7, we visualize the `CLS` token's attention map at the final layer of the transformer with or without our PGN. Even though we only show the effect of the prompts on the last layer's attention map, we still find a strong effect of the PGNs additionally supplied prompts. While the effect is not interpretable for low-resolution datasets such as CIFAR, for Pets and Resisc we observe a strengthened focus on the object vs the background. We also show the attention values of the `CLS` to the 16 supplied prompts below the PGN-CLIP attention maps. A strong variance between images is seen, demonstrating that the method indeed learns and leverages the input-dependency of the prompts that are supplied to the frozen model.

5 Discussion and Conclusion

We propose Prompt Generator Network (PGN), a simple and effective framework for learning input-dependent visual prompts. Our method is parameter-efficient by leveraging a Token Library from which tokens are combined to yield the prompts using a lightweight ResNet. We demonstrate that PGN can be used to adapt CLIP, surpassing previous methods and even matching and exceeding the results of full-finetuning of the visual encoder, despite requiring two orders of magnitude less number of adapted weights. Finally, we have demonstrated that PGN can be a scalable method for generic adaptation of frozen visual transformers by training them with a mixture of datasets.

REFERENCES

Yuki M. Asano, Christian Rupprecht, and Andrea Vedaldi. Self-labelling via simultaneous clustering and representation learning. In *ICLR*, 2020.

Hyojin Bahng, Ali Jahanian, Swami Sankaranarayanan, and Phillip Isola. Exploring visual prompts for adapting large-scale models. *arXiv:2203.17274*, 2022.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *ECCV*, 2014.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *NeurIPS*, 33:1877–1901, 2020.

Han Cai, Chuang Gan, Ligeng Zhu, and Song Han. Tinytl: Reduce memory, not parameters for efficient on-device learning. In *NeurIPS*, pp. 11285–11297, 2020.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *ECCV*, pp. 132–149, 2018.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, pp. 9650–9660, 2021.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *ICML*, 2020.

Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883, 2017.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *CVPR*, pp. 3606–3613, 2014.

Virginia de Sa. Learning classification with unlabeled data. In *NeurIPS*, 1993.

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. *ICLR*, 2020.

Gamaleldin F Elsayed, Ian Goodfellow, and Jascha Sohl-Dickstein. Adversarial reprogramming of neural networks. *arXiv preprint arXiv:1806.11146*, 2018.

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. *arXiv:2110.04544*, 2021.

Tengda Han, Weidi Xie, and Andrew Zisserman. Memory-augmented dense predictive coding for video representation learning. In *ECCV*, pp. 312–329. Springer, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pp. 770–778, 2016.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pp. 9729–9738, 2020.

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.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. *ICML*, 2019.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*, pp. 4904–4916. PMLR, 2021.

Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. *ECCV*, 2022.

Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, pp. 2901–2910, 2017.

Chen Ju, Tengda Han, Kunhao Zheng, Ya Zhang, and Weidi Xie. Prompting visual-language models for efficient video understanding. In *ECCV*, 2022.

James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *PNAS*, 114(13):3521–3526, 2017.

Ryan Kiros, Ruslan Salakhutdinov, and Richard S Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *NeurIPS*, 2014.

Eliska Kloberdanz, Jin Tian, and Wei Le. An improved (adversarial) reprogramming technique for neural networks. In *International Conference on Artificial Neural Networks*, pp. 3–15. Springer, 2021.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Technical Report*, 2009.

Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. Ask me anything: Dynamic memory networks for natural language processing. In *ICML*, pp. 1378–1387. PMLR, 2016.

Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. The open images dataset v4. *IJCV*, pp. 1956–1981, 2020.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *EMNLP*, 2021.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *ACL*, 2021.

Huaishao Luo, Lei Ji, Ming Zhong, Yang Chen, Wen Lei, Nan Duan, and Tianrui Li. Clip4clip: An empirical study of clip for end to end video clip retrieval and captioning. *Neurocomputing*, 508: 293–304, 2022.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. *NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pp. 722–729. IEEE, 2008.

Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *CVPR*, pp. 3498–3505. IEEE, 2012.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. Adapterhub: A framework for adapting transformers. *EMNLP*, 2020.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Technical Report*, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *ICML*, 2021.

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

Nitish Srivastava and Ruslan Salakhutdinov. Multimodal learning with deep boltzmann machines. *NeurIPS*, 2012.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. *NeurIPS*, 28, 2015.

Ashish Vaswani, Google Brain, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *ICML*, pp. 23965–23998. PMLR, 2022a.

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, et al. Robust fine-tuning of zero-shot models. In *CVPR*, pp. 7959–7971, 2022b.

Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, pp. 3733–3742, 2018.

Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, pp. 3485–3492. IEEE, 2010.

Yuan Yao, Ao Zhang, Zhengyan Zhang, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. Cpt: Colorful prompt tuning for pre-trained vision-language models. *arXiv:2109.11797*, 2021.

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *ACL*, 2021.

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*, pp. 698–714. Springer, 2020.

Renrui Zhang, Rongyao Fang, Peng Gao, Wei Zhang, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. Tip-adapter: Training-free clip-adapter for better vision-language modeling. *arXiv:2111.03930*, 2021.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *IJCV*, 2022a.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. *CVPR*, 2022b.

APPENDICES

A ADDITIONAL DETAILS

Our code and models will be published online.

A.1 DATASETS

Table 7 gives an overview of the downstream datasets used for the evaluation of our method, including the text prompt templates used to generate classifiers for CLIP.

Table 7: Description of the datasets and the corresponding text prompt used for CLIP. Table adapted from Bahng et al. (2022).

Dataset	Train Size	Validation Size	Test Size	Classes	Text Prompt
CIFAR100	50,000	-	10,000	100	"This is a photo of a { }"
CIFAR10	50,000	-	10,000	10	"This is a photo of a { }"
Flowers102	4,093	1,633	2,463	102	"This is a photo of a { }"
Food101	50,500	20,200	30,300	101	"This is a photo of a { }"
EuroSAT	13,500	5,400	8,100	10	"This is a photo of a { }"
SUN397	15,888	3,970	19,850	397	"This is a photo of a { }"
UCF101	7,639	1,898	3,783	101	"This is a photo of a { }"
SVHN	73,257	-	26,032	10	"This is a photo of a { }"
OxfordPets	2,944	736	3,669	37	"This is a photo of a { }"
DTD	2,820	1,128	1,692	47	"This is a photo of a { }"
Resisc45	18,900	6,300	6,300	45	"This is a photo of a { }"
CLEVR/count	70,000	-	15,000	8	"This is a photo of { } objects"

A.2 EXPERIMENTAL SETTINGS

Table 8: Experimental settings for each of our tables and figures. (*) In Table 1, the reported numbers for VP do not come from our own experimentation, hence our settings do not apply.

Table/Figure	Epochs	Number of prompts	TL Size	PGN resolution
Table 1	1000*	16	256	224×224
Table 2	500	8	64	64×64
Table 3	500	-	-	-
Table 4	500	16	256	224×224
Table 5	1000	16	256	224×224
Table 6	500	16	256	224×224
Figure 2	1000	16	256	224×224
Figure 3	500	8	64	64×64
Figure 4	1000	16	256	224×224
Figure 5	1000	16	256	224×224
Figure 6	1000	16	256	224×224
Figure 7	1000	16	256	224×224

A.3 ARCHITECTURES

In Table 9, we show the details of ResNet10 architectures.

A.4 FEATURE SIMILARITIES COMPUTATION

For Fig. 2, we embed the validation set of CIFAR-100 using the three visual encoders of PGN (only), CLIP, and PGN+CLIP. For this we cluster the features into 100 clusters using k-means. After this, the representations can be easily compared with each other using the normalised mutual information score.

Table 9: The structure of ResNet10, which is modified from ResNet18 to be more light-weight. Modifications are marked in red. Note that the final classification layer is omitted.

stage	specification	output sizes $H \times W \times C$
input data	-	$224^2 \times 3$
conv_1	$7 \times 7, 16$ stride $2, 2$	$112^2 \times 16$
pool_1	$3 \times 3, 16$ stride $2, 2$	$56^2 \times 16$
res_2	$\begin{bmatrix} 3 \times 3, 16 \\ 3 \times 3, 16 \end{bmatrix} \times 1$	$56^2 \times 16$
res_3	$\begin{bmatrix} 3 \times 3, 32 \\ 3 \times 3, 32 \end{bmatrix} \times 1$	$28^2 \times 32$
res_4	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 1$	$14^2 \times 64$
res_5	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 1$	$7^2 \times 128$
pool_2	$7 \times 7, 128$ stride $1, 1$	$1^2 \times 128$

B Details of Feature Similarity Analysis

In the NMI analysis in Section 4.2 and Fig. 2, we measure the pairwise alignment between the outputs of the visual encoders we use and the ground truth. These are: the frozen CLIP model's visual encoder that outputs `CLS` embedding, the trained PGN model that outputs prompts (the $\hat{\mathbf{h}}_V$ in Eq. (5)), and the combined CLIP+PGN model which uses PGN prompts to modify CLIP's visual encoder's outputs (that outputs `CLS` embedding after CLIP). For this, we apply k-means clustering to the set of embeddings generated by each encoder individually, setting k equal to the number of ground-truth classes. For our experiment, we use the full CIFAR100 test split. This yields a set of 3 pseudo labelings of the dataset. After combination with the ground-truth labels, we can make 6 pairwise comparisons and calculate the normalised mutual information, which measures a generalized correlation between labelings. The results are shown in Table 10.

Table 10: Normalized Mutual Information (NMI) score in %.

NMI	**GT**	**PGN**	**CLIP**	**PGN+CLIP**
GT	100	29.5	58.1	**70.1**
PGN		100	27.5	33.8
CLIP			100	61.2
PGN+CLIP				100

C Additional Experiments

Table 11: Feature projection layer type.

Type	CIFAR100	SUN397
Linear	77.9	**70.5**
MLP	**78.2**	70.4

We evaluate replacing the final linear layer of g_θ with a MLP with 1 hidden layer, which allows for a nonlinear mapping between image features and the logits that give rise to the combination coefficients in Eq. (4). No significant performance gain is observed.