AdaBoosting Text Prompts for Vision-Language Models

Seokhee Jin[1⋆], Changhwan Sung[2⋆], Sunung Mun[2], Hoyoung Kim[3], and Jungseul Ok[2†]

[1] KT Corporation, Republic of Korea
[2] Pohang University of Science and Technology (POSTECH), Republic of Korea
[3] National AI Research Lab, Republic of Korea
seokhee.jin@kt.com, {changhwan.sung, mtablo, hoyoung.kim, jungseul.ok}@postech.ac.kr
https://sung0503.github.io/TPB

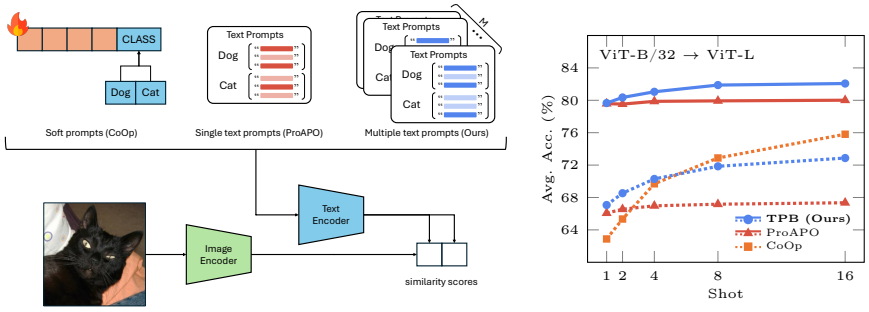
Abstract. The classification accuracy of pretrained Vision-Language Models (VLMs) relies on the quality of the text prompts. Handcrafted templates and Large Language Model (LLM)-generated descriptions not only make predictions more interpretable, but also enable reuse of the same prompts across heterogeneous VLMs. Recent works construct task-adapted text prompts with a small number of labeled images. However, existing few-shot text prompting methods do not explicitly focus on misclassified examples during prompt construction, leading to only marginal improvements even as more shots become available. To fully exploit few-shot supervision, we propose Text Prompt Boosting (TPB), an AdaBoost-inspired framework that treats each text-prompt-based classifier as a weak learner and sequentially aggregates them into a strong ensemble by explicitly targeting hard, misclassified examples. Extensive experiments show that TPB preserves task-intrinsic, model-agnostic cues in text space, enabling robust cross-model transfer. Across eleven classification benchmarks, TPB improves accuracy on the source model and preserves shot-driven gains when transferred to larger, more capable VLMs, where existing methods struggle to sustain such improvements.

Keywords: Vision-Language Models · Few-Shot Adaptation · Cross-Model Transferability

1 Introduction

Vision-Language Models (VLMs) [2, 3, 7, 14, 18, 29, 35, 36] such as CLIP [29] enable zero-shot image classification using natural-language text prompts that describe each class. In practice, the classification accuracy is sensitive to the quality of the prompts. To avoid labor-intensive manual prompt engineering, prior studies [21, 27, 30, 31, 39] have explored LLM-based automated prompt

⋆These authors contributed equally to this work.
†Corresponding author.



(a) VLM-based inference **(b)** Shot scalability

Fig. 1: Concept of TPB and its transfer-robust shot scalability. (a) While soft prompting (*e.g.*, CoOp) optimizes continuous prompt vectors via backpropagation and text-based baselines (*e.g.*, ProAPO) rely on a single prompt set, TPB iteratively builds a collection of prompt banks to form a strong classifier. (b) Performance comparison upon cross-model transfer from ViT-B/32 to five ViT-L-scale VLMs, where dashed and solid lines indicate source and transfer performance, respectively. While CoOp cannot be directly transferred and ProAPO fails to preserve shot-driven gains, TPB maintains these shot-driven gains while demonstrating superior scalability on the source model.

generation. A common practice is to query the Large Language Model with a generic instruction (*e.g.*, "*What does a {class} look like?*") [27], resulting in prompts based solely on the LLM's prior knowledge. However, such prompts often lack proper grounding in the target visual distribution and suffer from inaccuracies due to class-name ambiguity or hallucinations.

Recent studies [19, 28] address this lack of visual grounding by leveraging a small number of labeled images to facilitate the discovery of task-optimized prompts. To exploit these few-shot examples, these methods typically optimize a single aggregate metric (*e.g.*, overall validation accuracy) as their primary objective for prompt generation. However, the fundamental limitation is that such a metric can be dominated by a subset of easily recognizable samples. Once a candidate prompt correctly classifies these trivial instances, validation accuracy saturates prematurely, failing to capture the broader visual diversity of the target domain. As a result, these methods often overfit to the few-shot set and yield only marginal gains even as more labeled shots become available.

To address these limitations and effectively exploit few-shot supervision, we propose Text Prompt Boosting (TPB), drawing inspiration from the classic AdaBoost algorithm. Rather than optimizing a single prompt that quickly plateaus on easy samples, TPB iteratively builds an ensemble of natural-language prompts (as shown in Figure 1a) by explicitly redirecting focus onto "hard examples." At each boosting round, TPB reweights misclassified training examples, effectively forcing the selection of new prompts from an LLM-generated pool that resolve these specific errors. By systematically addressing failure modes, TPB maximizes the utility of few-shot examples to cover broader visual variations, successfully capturing model-agnostic task knowledge.

Extensive evaluations across diverse image classification benchmarks show that TPB consistently outperforms state-of-the-art baselines. Figure 1b previews this transfer-robust shot scalability across five ViT-L-scale VLMs. By effectively exploiting few-shot supervision, TPB exhibits superior shot scalability on the source model. Furthermore, although natural-language prompts are inherently transferable across heterogeneous VLMs through simple re-embedding, prior methods often fail to retain shot-driven gains after transfer. In contrast, TPB preserves the shot-driven improvements observed on the source model when transferred to larger, more capable VLMs, maintaining clear performance gains even at higher shot counts.

Our main contributions are summarized as follows:

- We propose a few-shot prompting framework that combines AdaBoost with natural-language text prompts, yielding a boosted ensemble with significantly improved shot scalability over prior text-based methods.
- We show that the prompt ensembles transfer seamlessly across heterogeneous VLMs by simple re-embedding, achieving superior cross-model performance compared to existing transfer baselines.
- We conduct extensive experiments across standard few-shot benchmarks and multiple VLM backbones, accompanied by detailed component analyses.

2 Related Work

2.1 Prompt Adaptation of VLMs.

Prompting strategies for vision-language models broadly fall into soft (continuous) and hard (discrete) paradigms. While soft prompting, often referred to as prompt-learning, which attaches learnable continuous context vectors to class names for parameter-efficient adaptation [15, 16, 24, 41, 42], the learned prompts lie in a model-specific representation space, making them unreadable and difficult to transfer directly across models. In contrast, hard prompt optimization searches over discrete natural-language tokens, ensuring that the resulting prompts remain human-readable and reusable across heterogeneous CLIP-style encoders. To efficiently navigate the enormous language space, recent methods leverage LLMs to generate and refine prompts. For instance, DCLIP [21] and CuPL [27] construct task-specific descriptions, whereas further methods incorporate few-shot adaptation via conversational feedback in LLMbo [19] and evolution-based search in ProAPO [28]. Alternatively, PEZ [38] employs a gradient-based hard-prompt optimization technique. However, despite the advantages of hard prompting, most existing methods lack mechanisms to effectively leverage more than a single labeled example per class. As a result, their performance tends to saturate and improves only marginally, even when more labeled examples are provided. Our work follows the hard-prompt paradigm but introduces an AdaBoost-based framework to effectively exploit additional supervision while preserving interpretability and cross-model transfer.

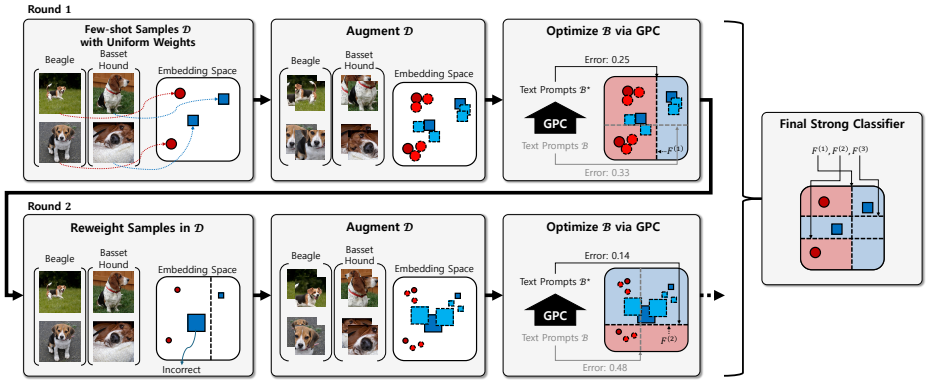


Fig. 2: Overview of our proposed TPB framework. Initially, sample weights are uniformly distributed on the few-shot set $\mathcal{D}$. At each round m, weights are updated based on the previous round's weak classifier. Then, we apply augmentation and an optimal prompt collection $\mathcal{B}^\star$ is derived via GPC; $\mathcal{B}^\star$ defines the current weak classifier. After the final round, all weak classifiers are aggregated into a single strong classifier.

2.2 Diverse Applications of AdaBoost.

Adaptive Boosting (AdaBoost) [10,11,43] sequentially combines many weak classifiers by iteratively reweighting training examples, emphasizing those misclassified by earlier classifiers. This general framework has been applied across diverse domains. In the computer vision domain, AdaBoost has been widely used in fast detection pipelines, such as cascaded detectors [37] and integral channel feature-based detectors [6]. In text classification and information retrieval, boosting aggregates weak lexical predicates into confidence-rated document classifiers [33] and optimizes pairwise orderings for ranking [9]. Unlike these applications, we address cross-modal image recognition with VLMs, treating class-wise text prompts as weak classifiers that map an image to class-wise similarity scores.

3 Method

We propose Text Prompt Boosting (TPB), a prompt-ensemble framework for shot-scalable and transfer-robust few-shot adaptation of pretrained VLMs. TPB treats a class-wise prompt collection as a text-based weak classifier and builds a strong classifier via an AdaBoost-style ensemble over multiple boosting rounds. We begin with preliminaries in Section 3.1, then introduce large prompt pools in Section 3.2 and our weak learner, Greedy Prompt Composition (GPC), in Section 3.3. Finally, we describe how GPC is integrated into the TPB boosting loop in Section 3.4. An overview of TPB is shown in Figure 2.

3.1 Preliminaries

VLM Classification with Text Prompts. We consider K-way classification for an image $\mathbf{x}$. Given a pretrained VLM with image encoder E_{img} and text encoder E_{text}, we form a CLIP-style linear classifier whose weights are text features derived from class prompts t_k. Specifically, the logit for class k is computed by cosine similarity between $E_{\text{img}}(\mathbf{x})$ and $E_{\text{text}}(t_k)$:

$$s(\mathbf{x}, t_k) := \cos\big(E_{\text{img}}(\mathbf{x}), E_{\text{text}}(t_k)\big)\,. \tag{1}$$

Instead of relying on a single prompt, multiple prompts can be grouped into a class-wise prompt bank $B_k = \{t_{k,1}, \ldots, t_{k,n_k}\}$, and compute the class score by simple averaging:

$$s(\mathbf{x}, B_k) := \frac{1}{|B_k|} \sum_{t_k \in B_k} s(\mathbf{x}, t_k)\,. \tag{2}$$

Let $\mathcal{B} := \{B_1, \ldots, B_K\}$ denote the collection of class-wise prompt banks. Inference on an image $\mathbf{x}$ is then performed using this prompt collection with a fixed temperature $\tau > 0$:

$$p(y = k \mid \mathbf{x}; \mathcal{B}) := \frac{\exp\big(s(\mathbf{x}, B_k)/\tau\big)}{\sum_{i=1}^{K} \exp\big(s(\mathbf{x}, B_i)/\tau\big)}\,. \tag{3}$$

Based on these probabilities, the predicted class label is determined by selecting the class with the highest probability:

$$\hat{y}(\mathbf{x}; \mathcal{B}) := \operatorname*{argmax}_{k \in \{1,\ldots,K\}} p(y = k \mid \mathbf{x}; \mathcal{B})\,. \tag{4}$$

While prior text-prompting studies [19, 21, 27, 28, 31] construct only a single collection $\mathcal{B}$, our TPB framework sequentially constructs multiple diverse collections $\{\mathcal{B}^{(m)}\}_{m=1}^{M}$. Each collection defines a weak classifier in an AdaBoost-style ensemble, enabling us to explicitly target hard examples while preserving the model transferability of discrete text prompts.

AdaBoost Overview. AdaBoost [10] is a prominent ensemble algorithm that combines weak learners into a strong learner by iteratively training weak classifiers on reweighted data. Unlike standard AdaBoost, which typically uses decision trees as weak learners, our framework uses the VLM classifier induced by a prompt collection $\mathcal{B}$ as the weak learner. At each boosting round $m \in \{1, \ldots, M\}$, we construct a prompt collection $\mathcal{B}^{(m)}$. Since the weak classifier is fully specified by the VLM predictions induced by $\mathcal{B}^{(m)}$, the objective at round m is to minimize the weighted classification error on the training set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$:

$$\mathcal{L}^{(m)} := \sum_{i=1}^{n} w_i^{(m)}\, \mathbb{I}\Big[y_i \neq \hat{y}(\mathbf{x}_i; \mathcal{B}^{(m)})\Big]\,, \tag{5}$$

where $w_i^{(m)}$ is the weight assigned to the i-th image at round m, indicating how much the current prompt collection should focus on it, and $\hat{y}$ denotes the predicted class defined in Equation (4).

After obtaining $\mathcal{B}^{(m)}$, we update the sample weights for the next round by increasing the weights of misclassified (hard) samples and decreasing those of correctly classified ones. This fitting and reweighting process repeats, encouraging subsequent prompt collections to prioritize the remaining difficult examples. The final prediction is obtained by aggregating the outputs from all M rounds. For example, aggregation can be performed via weighted majority voting:

$$\hat{y}_{\text{final}} = \arg\max_k \sum_{m=1}^{M} \alpha^{(m)} \cdot \mathbb{I}\Big[\hat{y}(\mathbf{x}; \mathcal{B}^{(m)}) = k\Big] \ , \tag{6}$$

where $\alpha^{(m)}$ is the weight for the m-th weak learner. The specific reweighting and aggregation rules depend on the AdaBoost variant; we adopt SAMME.R [43], a multi-class real-valued extension, which we refer to simply as AdaBoost in the rest of this paper.

3.2 Large Prompt Pools

Before initiating the boosting loop, we first construct a comprehensive prompt pool $\mathcal{P} = \{P_1, \ldots, P_K\}$ for each class $k \in \{1, \ldots, K\}$. To ensure sufficient expressive power, these prompt pools encompass both simple template-based prompts (*e.g.*, "*a photo of a beagle.*") and richer, more detailed sentences generated by LLMs (*e.g.*, "*beagles have large, floppy ears that hang down to the sides of their face.*"). Concretely, we utilize the 80 hand-crafted templates provided by CLIP [29], alongside LLM-generated sentences released by several prior studies [21, 27, 32, 39, 44]. To further expand the discrete hypothesis space and enrich semantic diversity, we also append concatenations of these templates and descriptive sentences to each pool.

3.3 Greedy Prompt Composition

At each boosting round $m \in \{1, \ldots, M\}$, our framework constructs a collection $\mathcal{B}^{(m)} = \{B_k^{(m)}\}_{k=1}^K$ of class-specific prompt banks, where each $B_k^{(m)}$ is selected from the class-specific prompt pool P_k. Exhaustively evaluating all joint prompt combinations over the K classes is computationally infeasible; thus, we adopt a two-stage greedy strategy, which we refer to as *Greedy Prompt Composition (GPC)*. For notational simplicity, we omit the round index in this subsection and write B_k and w_i instead of $B_k^{(m)}$ and $w_i^{(m)}$.

Stage 1: Single-Template Initialization. We initialize the prompt collection $\mathcal{B} = \{B_k\}_{k=1}^K$ with a single shared template. Among the 80 standard templates $\Phi_{\text{template}} = \{\phi_1, \ldots, \phi_{80}\}$, we select $\phi^\star$ that minimizes the weighted classification

error. Concretely, for each candidate template $\phi \in \Phi_{\text{template}}$, we form a template-induced collection $\mathcal{B}(\phi) = \{B_k(\phi)\}_{k=1}^{K}$ by setting $B_k(\phi) = [\,\phi(c_k)\,]$, where c_k denotes the label text of class k. We then compute the weighted error of the resulting classifier (under the current example weights) and choose the best template. If multiple templates attain the same minimum error, we break ties randomly. Accordingly, we define

$$\mathcal{B}_{\text{init}} := \{B_k\}_{k=1}^{K}, \quad B_k := [\,\phi^\star(c_k)\,] \;\; \forall k \in \{1, \ldots, K\}. \tag{7}$$

We adopt this single-template initialization for two reasons. First, any classifier requires at least one prompt per class, and a shared template provides this with minimal overhead by simply inserting the class name. Second, using one template consistently across classes already yields a reasonably strong baseline, which serves as a stable starting point for greedy refinement and promotes fast convergence.

Stage 2: Class-wise Greedy Prompt Insertion. Starting from the single-template initialization $\mathcal{B}_{\text{init}}$, we greedily expand the prompt banks by iteratively adding prompts to each class. Specifically, given the current prompt bank collection $\mathcal{B}$ and class k, we evaluate each candidate prompt $t \in P_k$ by temporarily appending it to the bank, *i.e.*, $\widetilde{B}_k(t) = B_k \| [t]$, and computing the resulting weighted error. Note that the class score is computed as an unweighted average over the prompts in a bank as defined in Equation (2). Under this formulation, adding a single prompt to an already populated bank may induce only a small change in the class score, which can be insufficient to resolve the remaining hard examples. Rather than introducing learnable continuous weights, which would require gradient updates, we implicitly approximate continuous weighting within a strictly discrete space by permitting duplicate selections of the same prompt; since the score is an unweighted average, duplicating a prompt increases its effective contribution. Ultimately, we select the candidate $t_k^\star$ that yields the largest reduction in the weighted error, updating $B_k \leftarrow B_k \| [t_k^\star]$ if such a candidate exists, or leaving B_k unchanged otherwise. We repeat this procedure cyclically across all classes $k = 1, \ldots, K$ until a full pass over all classes yields no further updates, resulting in the optimized bank collection $\mathcal{B}^\star$.

3.4 Integration into the TPB Framework

Boosting-Loop Image Augmentation. To induce informative variations and prevent overfitting, which occurs when the ensemble quickly memorizes the limited samples, we apply image augmentation to the few-shot dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$ before executing the GPC algorithm. With an augmentation factor $a \in \mathbb{N}$, we generate randomly transformed copies for each training image $\mathbf{x}_i$ using simple operations (*e.g.*, random resized crop and horizontal flip), resulting in an augmented dataset $\mathcal{D}_{\text{aug}}$ of size an. We replicate the original sample weight w_i for each copy and renormalize the weight vector $\mathbf{w}_{\text{aug}}$ to sum to one. Consequently, the weighted errors evaluated during the GPC stages are computed over this augmented dataset $\mathcal{D}_{\text{aug}}$ and its corresponding weights $\mathbf{w}_{\text{aug}}$.

Iterative Weight Update and Final Ensemble. In the overall TPB framework, the prompt bank collection $\mathcal{B}^{\star(m)}$ obtained from GPC serves as a weak classifier for boosting round m. We then apply this weak classifier to the original (non-augmented) few-shot dataset $\mathcal{D}$ to update the sample weights $\mathbf{w}^{(m)}$ to $\mathbf{w}^{(m+1)}$, increasing the weights of misclassified examples and decreasing those of correctly classified ones. Using these updated weights, the next boosting round constructs a new weak classifier that focuses more intensely on the previously misclassified samples. Note that we initialize the weights uniformly in the first round, i.e., $w_i^{(1)} = 1/|\mathcal{D}|$ for all $(\mathbf{x}_i, y_i) \in \mathcal{D}$. After the final round M, all weak classifiers are aggregated into a strong classifier. For the reweighting and aggregation rules, we adopt SAMME.R [43], a multi-class variant of AdaBoost. Detailed pseudocodes are provided in the supplementary material.

4 Experiments

4.1 Experimental Setup

Datasets. Following prior text prompting [19, 21, 27, 28] and prompt-learning studies [16, 42], we consider eleven image classification datasets that cover a wide range of domains and granularity: ImageNet-1K [5] and Caltech101 [8] for generic object classification, OxfordPets [25], StanfordCars [17], Flowers102 [23], Food101 [1], FGVCAircraft [20] for fine-grained image classification, SUN397 [40] for scene recognition, DTD [4] for texture classification, EuroSAT [13] for satellite image classification, and UCF101 [34] for action classification.

Implementation Details. The number of boosting rounds M is set to 50, except for ImageNet, where it is set to 30, and the augmentation factor a is set to 4. Our framework is implemented based on the AdaBoost implementation in scikit-learn [26]. For the temperature parameter τ in Equation (3), we use a fixed value $\tau = 1$. Note that during inference, the text embeddings of the ensembled prompts can be pre-computed. Thus, for any new image, TPB only requires a single forward pass, ensuring that our method does not incur significant latency overhead compared to other baselines. All results, except for LLMbo [19] as described in Section 4.2, are averaged over three random seeds; full shot-wise and target-wise tables with standard deviations are in the supplementary material.

4.2 Few-Shot Adaptation and Shot Scalability

Text-based Baselines. For zero-shot baselines, we employ the standard template "*a photo of a {class}.*" (denoted as CLIP), along with DCLIP [21] and CuPL [27]. For few-shot baselines, we compare our method against LLMbo [19], ProAPO [28], and a gradient-based hard-prompt optimization method, PEZ [38]. Except for LLMbo, all baseline results are reproduced using the authors' official implementations. Since PEZ was not originally designed for our task, we reimplement it for image classification; implementation details are provided in the supplementary material.

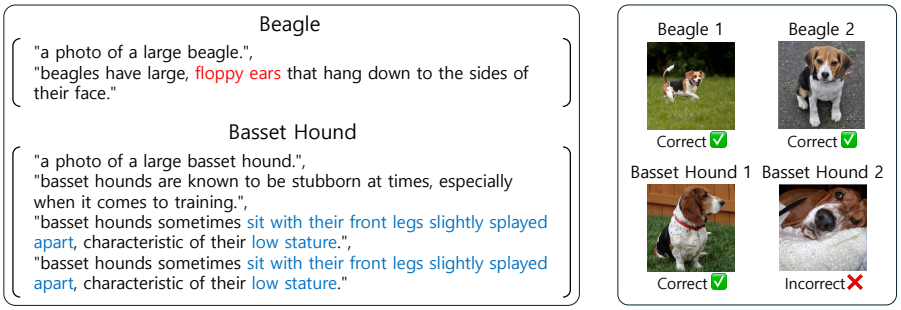
Table 1: Comparison of shot scalability among text-based methods. Top-1 accuracy (%) on eleven datasets using OpenAI CLIP RN50. TPB outperforms all text-based baselines in both one-shot and sixteen-shot scenarios.

Shot	Method	IN-1K	Caltech	Pets	Cars	Flowers	Food	Aircraft	SUN	DTD	ESAT	UCF	Avg.
Zero-shot scenario													
ZS	CLIP	58.3	85.8	83.7	55.9	61.2	75.2	14.5	58.5	40.0	24.2	58.4	56.0
ZS	DCLIP	59.6	88.6	83.1	53.9	66.3	76.6	16.8	61.0	41.7	37.6	60.8	58.7
ZS	CuPL	61.5	88.0	87.6	56.1	68.1	77.1	18.3	61.9	47.6	36.3	62.1	60.4
Few-shot scenario													
1 shot	CoOp	55.5	88.0	86.3	55.6	68.2	74.2	8.4	60.2	43.3	51.1	61.9	59.3
1 shot	PEZ	34.2	66.5	68.8	38.2	57.8	51.9	12.8	35.6	28.8	29.4	41.8	42.4
1 shot	LLMbo	59.6	**89.1**	88.1	56.2	67.2	**78.3**	18.1	61.0	44.8	49.0	60.2	61.1
1 shot	ProAPO	**61.1**	89.0	**89.0**	**57.6**	68.9	78.3	18.2	61.9	48.3	**53.3**	63.6	62.7
1 shot	**TPB (Ours)**	60.7	88.8	87.2	57.4	**77.0**	76.4	**19.7**	**63.7**	**53.1**	51.5	**66.2**	**63.8**
16 shot	CoOp	62.9	91.9	86.3	72.9	94.7	74.4	31.5	68.5	63.4	83.0	75.7	73.2
16 shot	PEZ	54.5	85.6	82.3	56.9	75.5	70.5	20.6	57.0	50.6	58.1	60.4	61.1
16 shot	LLMbo	59.9	89.5	88.3	56.8	67.4	78.3	18.1	60.8	44.9	51.4	60.5	61.4
16 shot	ProAPO	60.1	90.1	**89.1**	58.5	73.3	78.8	18.7	63.0	52.5	58.6	65.5	64.4
16 shot	**TPB (Ours)**	**64.7**	**91.8**	89.0	**65.0**	**85.9**	**79.0**	**24.8**	**70.7**	**62.8**	**66.9**	**73.0**	**70.3**

Shot Scalability. Table 1 compares our TPB with existing text-based methods on eleven datasets using OpenAI CLIP [29] with a ResNet-50 backbone [12]. While other text-based baselines show only marginal or no gains as more shots become available, our TPB achieves both strong few-shot performance and clear improvements with additional labeled data. At one shot, TPB slightly outperforms ProAPO by about 1.1 percentage points (pp) in average accuracy. When the number of shots increases from one to sixteen, TPB gains 6.5 pp and outperforms ProAPO and other text-based baselines by a substantial margin. PEZ benefits from additional shots and improves by 18.7 pp between one and sixteen shots, but it remains clearly worse than other text-based baselines at one shot and only approaches their performance at sixteen shots.

We also compare TPB with the continuous prompt-learning method CoOp [42]. While CoOp achieves higher accuracy at larger shot counts, this comes at the cost of being coupled to the model's specific representation space. In contrast, TPB preserves the universal nature of language, unlocking robust cross-model transferability where continuous prompts fail as further discussed in Section 4.3.

Qualitative Analysis: Targeting Hard Samples. To visualize how TPB iteratively builds an ensemble by explicitly targeting misclassified images, we present a qualitative example of the GPC algorithm on a two-class task distinguishing *beagle* from *basset hound*. As shown in Figure 3a, GPC constructs weak



(a) Text Prompts **(b)** Classification Results

Fig. 3: Qualitative example of a weak classifier. (a) Text prompts for *beagle* and *basset hound*. (b) The weak classifier defined by these prompts correctly classifies three images, but it misclassifies the image of a basset hound lying down, whose ears and legs are mostly occluded.

classifiers by selecting prompts from the pool that capture dominant, easily recognizable features. For the *beagle*, it highlights "large", "floppy ears," while for the *basset hound*, it emphasizes "low stature" and "front legs slightly splayed apart."

While this prompt bank correctly classifies standard images as depicted in Figure 3b, it fails on a challenging *hard sample*, a basset hound lying down where primary discriminative visual cues like legs and overall stature are heavily occluded. In a standard text-prompting paradigm, this error would likely remain unresolved. However, within the TPB framework, this specific misclassification is clearly penalized, assigning the image a higher sample weight in the subsequent boosting round. Consequently, the next iteration of GPC is explicitly forced to search for alternative textual cues, such as facial wrinkles or distinct snout shapes, specifically to correct this unresolved error. This highlights the underlying mechanism of TPB: rather than relying on a single static validation objective, the framework adapts to challenging samples by incrementally assembling diverse natural-language prompts.

4.3 Transfer Robustness Across Heterogeneous VLMs

Source and Target VLMs. In our cross-model transfer experiments, we distinguish between 'source models' (used for prompt optimization) and 'target models' (used for evaluation). Note that the concept of a source model is not applicable to zero-shot methods, as they do not involve model-specific optimization. As sources, we use OpenAI CLIP [29] with two backbones: ResNet-50 and ViT-B/32. For target models, we use OpenCLIP [2], EVA-02-CLIP [35], CLIPA-v2 [18], DFN [7], SigLIP2 [36], and MetaCLIP2 [3]. These VLMs are released with various backbones, which we group by their size (*e.g.*, ViT-L, ViT-H) and report averaged results for each group. This setup covers a range of architectures, scales, and training recipes, allowing us to assess model-transfer performance across heterogeneous VLMs.

Table 2: Cross-model transferability of TPB. Average Top-1 accuracy (%) across eleven datasets. All methods are optimized on OpenAI CLIP ViT-B/32 and directly evaluated on larger, heterogeneous target models (ViT-L and ViT-H). TPB preserves shot-driven gains where other text-based baselines saturate, demonstrating superior robustness and scalability during model transfer.

Target Model	Method	Shot				
		1	2	4	8	16
Source Model: OpenAI CLIP, ViT-B/32						
ViT-L: OpenCLIP, SigLIP2, DFN, EVA-02-CLIP, CLIPA-v2	ZS-CLIP			76.76		
	CoOp+EFT	63.81	65.98	69.45	71.78	74.05
	PromptSRC+EFT	66.79	68.07	71.65	73.45	75.20
	PEZ	49.05	53.21	55.56	57.84	58.93
	ProAPO	79.59	79.53	79.87	79.93	80.01
	TPB (Ours)	**79.66**	**80.35**	**81.05**	**81.88**	**82.07**
ViT-H: OpenCLIP, DFN, CLIPA-v2, MetaCLIP2	ZS-CLIP			78.75		
	CoOp+EFT	62.69	64.63	68.17	70.40	72.73
	PromptSRC+EFT	65.34	67.02	70.52	72.28	74.16
	PEZ	46.97	51.50	54.59	56.47	57.60
	ProAPO	**82.07**	81.90	82.26	82.25	82.39
	TPB (Ours)	81.73	**82.12**	**83.24**	**83.84**	**84.24**

Transfer Baselines. For model-transfer experiments, we evaluate both the text-based methods and prompt-learning baselines. As a representative prompt-learning method, we use PromptSRC [16], which adapts VLMs by learning visual and textual prompt tokens while keeping the backbone parameters frozen. Since these learned vectors cannot be directly reused in other VLMs with different architectures or representation spaces, we additionally employ emulated fine-tuning (EFT) [22] proposed in the language domain. EFT combines a large pretrained model with a smaller fine-tuned model by reweighting logits, effectively emulating the result of fine-tuning the large model without updating its parameters. This allows us to transfer the adaptation learned by PromptSRC to other VLMs in a training-free manner, but the inference overhead increases due to multiple forward passes (see the supplementary material for details).

Transfer-robust Shot Scalability. Table 2 reports cross-model transfer results when all methods operate on OpenAI CLIP ViT-B/32 with one to sixteen shot supervision and then evaluated on larger target models (ViT-L and ViT-H). On ViT-L targets, TPB improves the average accuracy from 76.76% (zero-shot CLIP) to 82.07%; on ViT-H targets, it improves from 78.75% to 84.24%. TPB also outperforms ProAPO, the strongest text-based baseline, by 2.06 pp on ViT-L targets and 1.85 pp on ViT-H targets. Notably, TPB also maintains a performance margin over CoOp+EFT and PromptSRC+EFT. While continuous methods like CoOp can achieve higher accuracy on the source model by over-

Table 3: Effect of gradient-optimized text prompts on model transfer. Transfer accuracy (%) over eleven datasets, adapted on RN50 and transferred to ViT-L/14.

Backbone	Methods		
	ZS-CLIP	PEZ	TPB (ours)
RN50 (source)	55.98	<u>61.09</u>$\pm$0.79	**70.32**$\pm$0.51
ViT-L/14 (target)	<u>71.07</u>	53.02$\pm$1.75	**78.46**$\pm$0.36

fitting to its specific representation space, they fail to preserve this advantage during transfer. In contrast, TPB achieves superior transferability and scalability while requiring only a single forward pass during inference, whereas EFT-based transfer necessitates multiple inference passes. Moreover, TPB exhibits superior transfer-robust shot scalability. While ProAPO's performance gains from one-shot to sixteen-shot are limited to 0.42 pp on ViT-L and 0.32 pp on ViT-H, TPB achieves significantly higher improvements of 2.41 pp and 2.51 pp, respectively. This contrast demonstrates that TPB consistently preserves shot-driven gains where other text-based baselines saturate, capturing architecture-agnostic task knowledge that remains robust during cross-model transfer. Conversely, PEZ, which optimizes discrete prompts via gradients, transfers poorly and falls even below zero-shot performance, as we discuss in the following analysis.

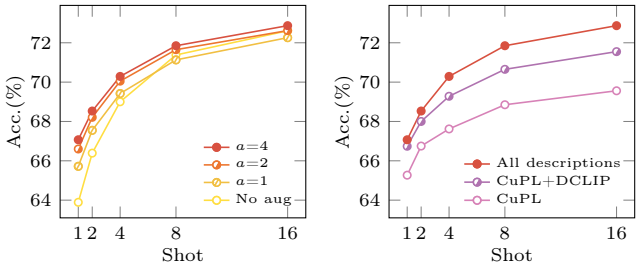
Additional shot-wise transfer results on larger target VLMs (*e.g.*, ViT-E and ViT-G) are provided in the supplementary material. Consistent with Table 2, they show that TPB preserves shot-driven gains after transfer.

Failure Analysis of Gradient-optimized Prompts. Table 3 reports the sixteen-shot average accuracy over eleven datasets when adapting OpenAI CLIP RN50 and transferring the resulting prompts to OpenAI CLIP ViT-L/14. On the source model, PEZ improves zero-shot CLIP from 55.98% to 61.09%, indicating that it successfully adapts the RN50 backbone to the training tasks. However, the learned prompts transfer poorly: on the ViT-L/14 target, accuracy drops from 71.07% (zero-shot CLIP) to 53.02%, well below even the zero-shot baseline. To understand why PEZ transfers so poorly despite using text prompts, we inspect the optimized context tokens. For example, for the *pug* class in OxfordPets, PEZ yields the following prompts:

"hey darby pls violets kissestc accept liza any,,
adorable gorgeous ♡♡ :-) life behaved *pug*"

Here, every token except the final class name "*pug*" is obtained by gradient updates during optimization. The resulting context is a sequence of seemingly arbitrary words, symbols, and non-word fragments with no coherent natural-language meaning. Such prompts behave like model-specific codes in the RN50 representation space, rather than semantically meaningful descriptions that a human could interpret.

This observation suggests that the ability of a prompt to transfer across models does not come simply from it being text. Instead, prompts are more



(a) Augmentation factor a **(b)** Prompt pool ablation

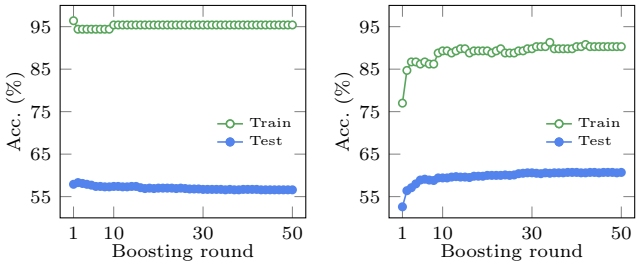
Fig. 4: Ablation of each component of TPB. Average accuracy over eleven datasets when using OpenAI CLIP ViT-B/32 as both source and target model. (a) Performance for different augmentation factors a (b) Performance for different constructions of the prompt pool.

transferable when they stay close to fluent, human-readable natural-language. Prompts that drift too far from human-interpretable language may overfit a particular model and therefore fail to transfer.

4.4 Ablation Studies

Effect of Augmentation and Prompt Pool Diversity. In Figure 4a, increasing the augmentation factor a consistently improves accuracy over using no augmentation, with the largest gains in the low-shot setting. Here, a controls the number of random transformations per training example, and $a = 0$ corresponds to using only the fixed few-shot set without any augmentation. Without augmentation, AdaBoost quickly fits the small training set within a few rounds, leaving little signal for subsequent boosting rounds, whereas randomly transformed views of the same images keep introducing informative errors, enabling later weak classifiers to correct them and yielding a stronger ensemble. Figure 4b then examines the impact of the diversity of the prompt pool, constructed by combining several groups of LLM-generated descriptions. Using both CuPL and DCLIP descriptions yields better performance than using CuPL alone, even though DCLIP is individually weaker, as reflected by its lower accuracy in Table 1. Incorporating all available descriptions further improves accuracy, especially at higher shot counts, suggesting that a more diverse prompt pool helps TPB better exploit additional labeled examples as they become available.

Dynamics of Augmentation Across Boosting Rounds. Figure 5 compares AdaBoost with and without image augmentation. Without augmentation (Figure 5a), training the ensemble rapidly memorizes the few available examples, pushing training accuracy close to 100% within only a few rounds. Once training accuracy saturates, the reweighted distribution no longer highlights informative mistakes, preventing AdaBoost from meaningfully updating its weak



(a) No augmentation (a=0) **(b)** With augmentation (a=4)

Fig. 5: Effects of augmentation over boosting rounds. Train and test accuracy on StanfordCars in the one-shot setting using OpenAI CLIP ViT-B/32.

Table 4: Ablation on the number of weak classifiers. Results (%) averaged over 10 datasets excluding ImageNet-1K using OpenAI CLIP ViT-B/32.

Shots	ProAPO	TPB (ours)			
		$M=1$	$M=10$	$M=30$	$M=50$
1	66.27±0.46	61.80±0.74	66.77±0.44	67.28±0.77	**67.39**±0.81
16	67.74±0.39	70.13±0.26	72.84±0.20	73.21±0.17	**73.36**±0.25

learners; consequently, test accuracy remains almost unchanged across subsequent rounds. In contrast, when applying image augmentation with a factor of a=4 (Figure 5b), each boosting round observes newly transformed views of the same images. These variations prevent the model from perfectly fitting the training set, keeping training accuracy below saturation and ensuring that each round still produces informative errors for reweighting. This continual supply of hard examples stabilizes the boosting process and allows test accuracy to improve steadily over many rounds, demonstrating that augmentation is essential for effective boosting under low-shot supervision.

Effect of the Number of Weak Classifiers. Table 4 analyzes how the performance of TPB varies with the number of weak classifiers M. With only a single weak classifier (M=1), TPB is slightly weaker than ProAPO in the one-shot setting. However, once M reaches 10, TPB already matches or exceeds ProAPO at both one-shot and sixteen-shot. Increasing M further to 30 or 50 provides additional but diminishing improvements, suggesting that performance effectively saturates in the range of M=30-50.

Boosting rounds *vs*. augmentation. To isolate boosting from image augmentation, we fix the total exposure budget $M{\cdot}a$=200 and sweep (M, a) as shown in Table 5. This sweep is conducted in the 16-shot setting over ten datasets excluding ImageNet, using OpenAI CLIP ViT-B/32 as the source model and averaging transfer accuracy over the same target groups as in Table 2: five ViT-L-scale and four ViT-H-scale VLMs. Under this fixed exposure budget, increasing

Table 5: Boosting *vs*. augmentation. Top-1 accuracy (%) in the 16-shot setting, averaged over 10 datasets excluding ImageNet with fixed exposure $M \cdot a = 200$.

Backbones		ZS-CLIP	M/a			
			1/200	5/40	10/20	50/4
Source	ViT-B/32	60.1	**74.42**$\pm$0.17	72.94$\pm$0.10	<u>73.32</u>$\pm$0.26	73.36$\pm$0.47
Target	ViT-L	76.53	80.77$\pm$0.15	81.13$\pm$0.08	81.54$\pm$0.07	**82.24**$\pm$0.39
	ViT-H	78.60	83.12$\pm$0.26	83.72$\pm$0.19	83.98$\pm$0.25	**84.49**$\pm$0.53

a while reducing M improves source accuracy but degrades transfer accuracy. This comparison shows that transfer robustness is driven by iterative boosting rather than by augmented exposure alone.

5 Conclusion

We introduce Text Prompt Boosting (TPB), a boosting-based framework that ensembles natural-language prompts for shot-scalable and transfer-robust few-shot adaptation of vision-language models. By treating a class-wise prompt bank collection as a weak classifier and employing the Greedy Prompt Composition (GPC) procedure as the weak learner, TPB incrementally builds an ensemble of prompt-based classifiers that explicitly focus on hard examples while preserving interpretability and transferability. Consequently, the resulting strong classifier exhibits improved shot scalability on the source model, while simultaneously achieving transfer robustness, successfully preserving its shot-driven performance gains even when evaluated on various larger heterogeneous VLMs.

Limitations: Despite its effectiveness, TPB has limitations. First, its construction cost can be non-negligible on large-scale datasets, where the candidate space is substantially larger. Second, AdaBoost-style reweighting may overemphasize noisy or atypical few-shot samples. Finally, TPB selects prompts from a fixed pool, which may limit its flexibility for fine-grained or ambiguous classes.

Acknowledgements

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (Nos. RS-2019-II191906, Artificial Intelligence Graduate School Program; RS-2024-00457882, AI Research Hub Project; RS-2026-25511821, Development of Personalized Media Service Recommendation and Generative Technology), by a grant (RS-2025-00564342) from the Korea Institute for Advancement of Technology (KIAT), funded by the Ministry of Trade, Industry and Energy (MOTIE), and by Seoul R&D Program (SP240008) through the Seoul Business Agency (SBA) funded by the Seoul Metropolitan Government.

References

1. Bossard, L., Guillaumin, M., Van Gool, L.: Food-101–mining discriminative components with random forests. In: European conference on computer vision. pp. 446–461. Springer (2014)
2. Cherti, M., Beaumont, R., Wightman, R., Wortsman, M., Ilharco, G., Gordon, C., Schuhmann, C., Schmidt, L., Jitsev, J.: Reproducible scaling laws for contrastive language-image learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2818–2829 (2023)
3. Chuang, Y.S., Li, Y., Wang, D., Yeh, C.F., Lyu, K., Raghavendra, R., Glass, J., Huang, L., Weston, J., Zettlemoyer, L., et al.: Meta clip 2: A worldwide scaling recipe. arXiv preprint arXiv:2507.22062 (2025)
4. Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., Vedaldi, A.: Describing textures in the wild. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3606–3613 (2014)
5. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
6. Dollár, P., Tu, Z., Perona, P., Belongie, S.J.: Integral channel features. In: Bmvc. vol. 2, p. 5. London, UK (2009)
7. Fang, A., Jose, A.M., Jain, A., Schmidt, L., Toshev, A., Shankar, V.: Data filtering networks. arXiv preprint arXiv:2309.17425 (2023)
8. Fei-Fei, L., Fergus, R., Perona, P.: Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In: 2004 conference on computer vision and pattern recognition workshop. pp. 178–178. IEEE (2004)
9. Freund, Y., Iyer, R., Schapire, R.E., Singer, Y.: An efficient boosting algorithm for combining preferences. Journal of machine learning research **4**(Nov), 933–969 (2003)
10. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and system sciences **55**(1), 119–139 (1997)
11. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). The annals of statistics **28**(2), 337–407 (2000)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
13. Helber, P., Bischke, B., Dengel, A., Borth, D.: 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 **12**(7), 2217–2226 (2019)
14. Jia, C., Yang, Y., Xia, Y., Chen, Y.T., Parekh, Z., Pham, H., Le, Q., Sung, Y.H., Li, Z., Duerig, T.: Scaling up visual and vision-language representation learning with noisy text supervision. In: International conference on machine learning. pp. 4904–4916. PMLR (2021)
15. Khattak, M.U., Rasheed, H., Maaz, M., Khan, S., Khan, F.S.: Maple: Multi-modal prompt learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 19113–19122 (2023)

16. Khattak, M.U., Wasim, S.T., Naseer, M., Khan, S., Yang, M.H., Khan, F.S.: Self-regulating prompts: Foundational model adaptation without forgetting. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 15190–15200 (2023)
17. Krause, J., Stark, M., Deng, J., Fei-Fei, L.: 3d object representations for fine-grained categorization. In: Proceedings of the IEEE international conference on computer vision workshops. pp. 554–561 (2013)
18. Li, X., Wang, Z., Xie, C.: Clipa-v2: Scaling clip training with 81.1% zero-shot imagenet accuracy within a $10,000 budget. In: R0-FoMo: Robustness of Few-shot and Zero-shot Learning in Large Foundation Models (2023)
19. Liu, S., Yu, S., Lin, Z., Pathak, D., Ramanan, D.: Language models as black-box optimizers for vision-language models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12687–12697 (2024)
20. Maji, S., Rahtu, E., Kannala, J., Blaschko, M., Vedaldi, A.: Fine-grained visual classification of aircraft. arXiv preprint arXiv:1306.5151 (2013)
21. Menon, S., Vondrick, C.: Visual classification via description from large language models. arXiv preprint arXiv:2210.07183 (2022)
22. Mitchell, E., Rafailov, R., Sharma, A., Finn, C., Manning, C.D.: An emulator for fine-tuning large language models using small language models. arXiv preprint arXiv:2310.12962 (2023)
23. Nilsback, M.E., Zisserman, A.: Automated flower classification over a large number of classes. In: 2008 Sixth Indian conference on computer vision, graphics & image processing. pp. 722–729. IEEE (2008)
24. Park, J., Ko, J., Kim, H.J.: Prompt learning via meta-regularization. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 26940–26950 (2024)
25. Parkhi, O.M., Vedaldi, A., Zisserman, A., Jawahar, C.: Cats and dogs. In: 2012 IEEE conference on computer vision and pattern recognition. pp. 3498–3505. IEEE (2012)
26. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
27. Pratt, S., Covert, I., Liu, R., Farhadi, A.: What does a platypus look like? generating customized prompts for zero-shot image classification. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 15691–15701 (2023)
28. Qu, X., Gou, G., Zhuang, J., Yu, J., Song, K., Wang, Q., Li, Y., Xiong, G.: Proapo: Progressively automatic prompt optimization for visual classification. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 25145–25155 (2025)
29. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PmLR (2021)
30. Ren, Z., Su, Y., Liu, X.: Chatgpt-powered hierarchical comparisons for image classification. Advances in neural information processing systems **36**, 69706–69718 (2023)
31. Roth, K., Kim, J.M., Koepke, A., Vinyals, O., Schmid, C., Akata, Z.: Waffling around for performance: Visual classification with random words and broad concepts. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 15746–15757 (2023)

32. Saha, O., Van Horn, G., Maji, S.: Improved zero-shot classification by adapting vlms with text descriptions. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 17542–17552 (2024)
33. Schapire, R.E., Singer, Y.: Boostexter: A boosting-based system for text categorization. Machine learning **39**(2), 135–168 (2000)
34. Soomro, K., Zamir, A.R., Shah, M.: Ucf101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402 (2012)
35. Sun, Q., Fang, Y., Wu, L., Wang, X., Cao, Y.: Eva-clip: Improved training techniques for clip at scale. arXiv preprint arXiv:2303.15389 (2023)
36. Tschannen, M., Gritsenko, A., Wang, X., Naeem, M.F., Alabdulmohsin, I., Parthasarathy, N., Evans, T., Beyer, L., Xia, Y., Mustafa, B., et al.: Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. arXiv preprint arXiv:2502.14786 (2025)
37. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001. vol. 1, pp. I–I. Ieee (2001)
38. Wen, Y., Jain, N., Kirchenbauer, J., Goldblum, M., Geiping, J., Goldstein, T.: Hard prompts made easy: Gradient-based discrete optimization for prompt tuning and discovery. Advances in Neural Information Processing Systems **36**, 51008–51025 (2023)
39. Wu, W., Yao, H., Zhang, M., Song, Y., Ouyang, W., Wang, J.: Gpt4vis: What can gpt-4 do for zero-shot visual recognition? arXiv preprint arXiv:2311.15732 (2023)
40. Xiao, J., Hays, J., Ehinger, K.A., Oliva, A., Torralba, A.: Sun database: Large-scale scene recognition from abbey to zoo. In: 2010 IEEE computer society conference on computer vision and pattern recognition. pp. 3485–3492. IEEE (2010)
41. Zhou, K., Yang, J., Loy, C.C., Liu, Z.: Conditional prompt learning for vision-language models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 16816–16825 (2022)
42. Zhou, K., Yang, J., Loy, C.C., Liu, Z.: Learning to prompt for vision-language models. International Journal of Computer Vision **130**(9), 2337–2348 (2022)
43. Zhu, J., Zou, H., Rosset, S., Hastie, T., et al.: Multi-class adaboost. Tech. Rep. 430, Department of Statistics, University of Michigan (2005)
44. Zhu, Y., Ji, Y., Zhao, Z., Wu, G., Wang, L.: Awt: Transferring vision-language models via augmentation, weighting, and transportation. Advances in Neural Information Processing Systems **37**, 25561–25591 (2024)