CONTEXT-OPTIMIZED PROMPTS FOR MOBILECLIP2: EFFICIENT MULTI-MODAL ADAPTATION ON RESOURCE-CONSTRAINED MODELS

A PREPRINT

Vlad Vladutu
Department of Computer Science
Aarhus University
Nordre Ringgade 1, 8000 Aarhus Centrum
au799236@uni.au.dk

Steinar Örn Sólmundsson
Department of Computer Science
Aarhus University
Nordre Ringgade 1, 8000 Aarhus Centrum
au798559@uni.au.dk

December 5, 2025

ABSTRACT

MobileCLIP2 offers strong zero-shot performance in a compact form factor suitable for on-device inference, but practical deployment on resource-constrained platforms such as smart glasses requires efficient adaptation to everyday objects and scenes under limited supervision. We investigate context-optimized prompting as a parameter- and memory-efficient alternative to fine-tuning, learning textual and visual prompts while keeping the entire MobileCLIP2 backbone frozen.

Keywords CLIP · Context Optimization · Prompt Training · Mobile Architecture

1 Introduction

Strong zero-shot accuracy of MobileCLIP2 does not guarantee that prompt-based performance will scale in realistic deployments. MobileCLIP2 is obtained by augmenting a dataset with predictions from multiple expert models, and this sophisticated training combined with the small parameter budget makes its response to prompts hard to predict in practice. We lack a clear understanding of when MobileCLIP2 will exploit prompts effectively, when it will ignore or reject them, and how these behaviours depend on the choice of model size within the MobileCLIP2 family. In this paper we therefore study how context-optimized prompting (CoOp) performs in practice on MobileCLIP2 in resource-constrained deployments. Modern pretrained models are rarely used exactly as released, so it is essential to characterize how their accuracy and efficiency scale when they are adapted to a downstream task. Because CLIP is designed as a general-purpose model, prompts can be added or removed at runtime in a non-invasive way, enabling different on-device processes to query the same backbone in different ways while still approaching full fine-tuning accuracy for their specific needs.

2 Related Work

2.1 Learning Transferable Visual Models From Natural Language Supervision

CLIP (Contrastive Language–Image Pre-training) is a vision–language model that learns a shared embedding space for images and natural-language descriptions by training on hundreds of millions of image–text pairs from the web. Instead of relying on a fixed label set, it uses text prompts at inference time, enabling open-vocabulary, zero-shot classification and image–text retrieval without task-specific fine-tuning. This makes CLIP important because it turns free-form language into a flexible interface for visual recognition, achieves strong performance across many benchmarks out-of-the-box, and serves as a foundation for a large body of later work in vision–language and prompt-based adaptation.

2.2 Learning to Prompt for Vision-Language Models

As CLIP relies on manually designed text prompts to describe each class, its performance in practice is highly sensitive to the exact phrasing of these prompts, making "prompt engineering" both time-consuming and brittle. Context Optimization (CoOp) addresses this limitation by learning the context words in the prompt as continuous vectors while keeping the CLIP parameters frozen. In this way, CoOp automatically discovers task-specific prompts from a few labeled examples, turning CLIP into a data-efficient visual learner that consistently outperforms hand-crafted prompts and remains robust under distribution shifts.

2.3 Exploring Visual Prompts for Adapting Large-Scale Models

Beyond learning or optimizing text prompts, Bahng et al. propose visual prompting as a complementary way to adapt frozen large-scale vision and vision–language models like CLIP. Instead of only tuning the words in a textual template, they learn a small, input-agnostic perturbation in pixel space—a "visual prompt" that is added to every image—so that the unchanged backbone performs well on a new task. This approach achieves accuracy competitive with linear probing while preserving the benefits of prompt-based adaptation: the pre-trained model remains frozen, adaptation is lightweight, and the same visual prompt can even compensate for suboptimal text prompts and could improve robustness under distribution shift.

2.4 MobileCLIP2: Improving Multi-Modal Reinforced Training

Building on CLIP, MobileCLIP2 targets the same image–text alignment problem but under strict latency and model-size constraints, producing small models (from $75M$ to $450M$ parameters) that run in a few milliseconds on mobile and edge hardware while preserving strong zero-shot accuracy. It improves the original MobileCLIP recipe by refining the multi-modal reinforced training pipeline: MobileCLIP2 distills from stronger CLIP teacher ensembles and higher-quality caption generators trained and fine-tuned on curated DFN-based datasets, leading to state-of-the-art ImageNet-1k zero-shot results for its latency regime. For our setting, MobileCLIP2 illustrates both the promise and the challenge of small vision–language models: they are attractive for deployment but tend to overfit when fully fine-tuned on limited downstream data, which motivates using more parameter-efficient adaptation such as CoOp prompt learning rather than conventional fine-tuning.

3 Methods

3.1 Datasets

CLIP is a versatile model that can be evaluated on a variety of visual tasks. To assess its performance in realistic mobile and smart-glasses scenarios, we focus on datasets that contain common, everyday objects and diverse scene types. The selected datasets evaluate different visual understanding capabilities: fine-grained object categories, textures, general objects, and complex scenes. We therefore select four diverse datasets:

Flower102 2a A dataset of close-up flower photographs, containing 102 categories with 10 images per class. Reasoning: Suitable for testing fine-grained classification. Flowers require attention to subtle visual details.

DTD 2b (Describable Textures Dataset) A collection of "textures in the wild," featuring 47 texture categories with 120 images per class, totaling 5,640 images. Reasoning: Evaluates the model's ability to recognize diverse textures that occur frequently in real environments.

Caltech101 2c Object-centric imagery covering everyday items such as faces, airplanes, chairs, and musical instruments. The dataset consists of 101 categories with approximately 40–800 samples each, totaling over 9,000 images 1. Reasoning: Tests broad object recognition capability across varied semantic categories.

SUN397 2d A large-scale scene recognition dataset containing 397 indoor and outdoor scene categories (e.g., bedrooms, streets, forests, stores) with 108,754 total images. Reasoning: Measures scene-level understanding and environment recognition.

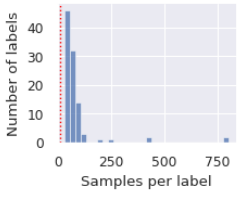
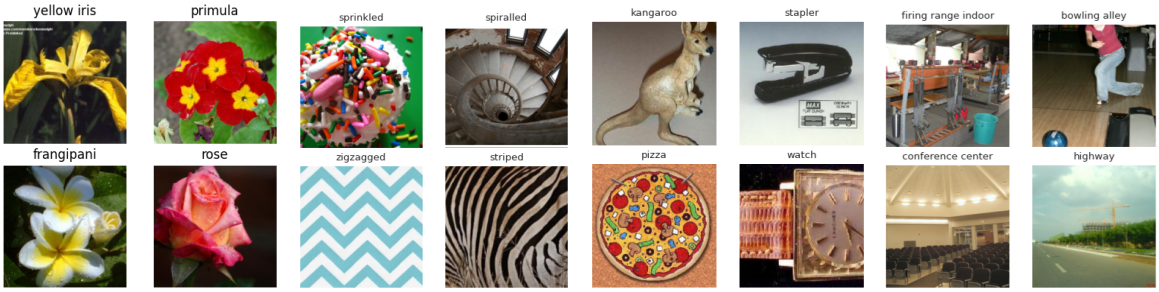


Figure 1: Caltech101 class distribution.

Although we ran experiments on all datasets, we focused primarily on Flower102 because its fine-grained visual complexity makes it the most suitable benchmark for evaluating detailed recognition performance.



(a) Flower102 (b) DTD (c) Caltech101 (d) SUN397

Figure 2: Dataset preview

3.1.1 Data splitting

Whenever available, we use the predefined training splits provided by the HuggingFace datasets. In cases where only training and test sets exist, we divide the test set evenly to form a validation subset. For datasets containing only a training split, we perform a $90\%/5\%/5\%$ partition into training, validation, and test sets. These proportions were chosen during early development, and we kept them fixed to avoid mixing up the splits later. An $80\%/10\%/10\%$ split would intuitively be preferable, as it would allocate more data to the validation and test sets, resulting in more stable estimates and a less noisy evaluation.

In the few-shot setting, we sample k images per class from the training set, with $k \in \{0, 1, 2, 4, 8, \text{all}\}$. We limit k to 8 because the Flower102 test split contains only 10 samples per class. Since this limits the effective training distribution to a constant k, training should be stable. We then evaluate performance using both plain accuracy and F1-scores, ensuring fair comparison under potential class imbalance across validation and test splits.

3.1.2 Data Preprocessing

MobileCLIP2 provides two general preprocessing transformations for images, one for training and one for inference, specifically designed for optimal CLIP usage, and we apply them accordingly. These preprocessing pipelines also vary across models in the family, resizing images to either 224 or 256 pixels depending on the underlying vision transformer backbone.

3.2 Implementation

We created the overview diagram in Figure 3 to illustrate the overall experimental setup, which we detail in the following sections. Briefly, the orange components denote the only trainable parameters, the text and image prompts. The prompt-augmented text is fed to the text encoder, the prompt-augmented image to the image encoder, and their resulting embeddings are compared to compute similarity.

3.2.1 Text prompting for classification

Assumptions: Based on the results reported in the original CoOp work, we anticipate strong performance when applying the method to MobileCLIP2, as both rely on similar ViT-based architectures. Furthermore, due to the low latency and efficiency of MobileCLIP2, we expect training to be relatively lightweight.

Procedure: CoOp constructs prompts by introducing M learnable context tokens $[\text{ctx}]_1, \ldots, [\text{ctx}]_M$, which are prepended to each category names, forming the prompt sequence:

$$p_i = [\text{ctx}]_1\, [\text{ctx}]_2\, \ldots\, [\text{ctx}]_M\, [\text{CLASS}_\text{i}].$$

We set $M = 16$ based on the following considerations. The MobileCLIP2 text encoder accepts at most 77 tokens (including start- and end-of-sentence), and class names can take several tokens, so the effective search range is about $[1, 64]$. Very short prompts (1–4 tokens) store little information (worse performance according to our findings), while lengths above 16 risk truncating important parts of the description during more complex tasks. Both 8 and 16 worked well, and for simplicity we use 16 in all experiments.

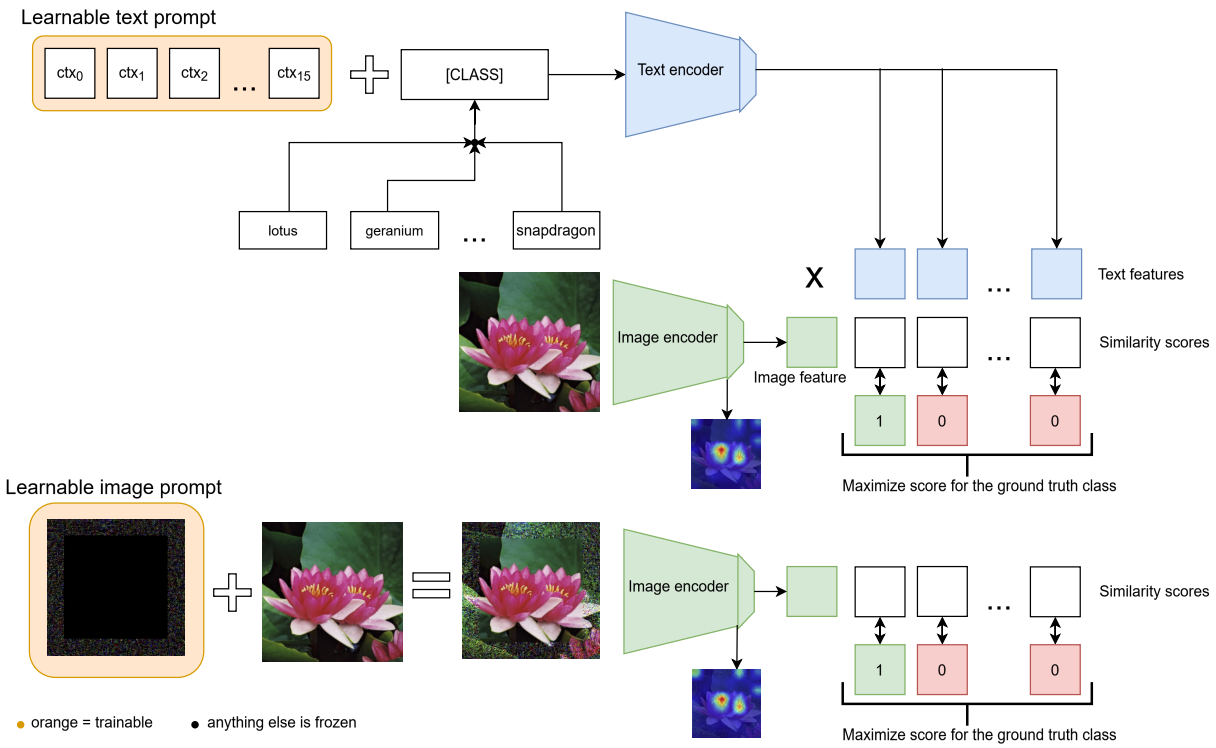


Figure 3: Overview diagram of CLIP Model with trainable prompts

We use the *unified context* variant proposed in the paper, where the same context tokens are shared across all classes (as opposed to *class-specific context*).

The procedure of classification is:

1. For each class i, we use the prompt sequence p_i and feed it to CLIP's text encoder.
2. We pass the image to CLIP's image encoder.
3. We normalize both feature vectors and compare them using cosine similarity.

The prediction errors are minimized using the cross-entropy loss with respect to the learnable context vectors. The entire pre-trained parameters are frozen so only the unfrozen prompt is updated with back-propagated gradients all the way through the text encoder.

Optimization: Training with limited data typically increases the risk of overfitting. As mentioned above CoOp provides two variants for prompt learning: unified context and class-specific context. The latter introduces substantially more parameters (50–400x), making it highly prone to overfitting in few-shot settings. For this reason, we adopt the unified context approach. By learning a single shared prompt across all classes, the model is encouraged to capture a generalizable representation rather than memorizing individual samples, reducing the likelihood of overfitting.

We employ the Adam optimizer, as its adaptive momentum properties typically offer more stable convergence than SGD with momentum in low-data, few-shot settings. Batch size is selected proportionally to the number of shots - larger shot counts allow larger batches and improved training throughput. The learning rate was determined experimentally across datasets, with 3×10^{-4} providing consistently fast convergence without instability.

We observed that training accuracy rapidly converged to 100%, while validation accuracy increased much more slowly, a clear indication of overfitting. To address this, we introduced weight decay into our custom pipeline and visualized the per-dimension amplitude of the soft words in the prompt to tune this parameter. Figure 4 shows the distribution of the amplitude of each dimension of the soft words before applying weight decay. On the right, several noisy bars deviate from the main curve, which can be smoothed by increasing the weight decay. Empirically, a small weight decay of 0.01 yields a smoother distribution and improves generalization stability across datasets.

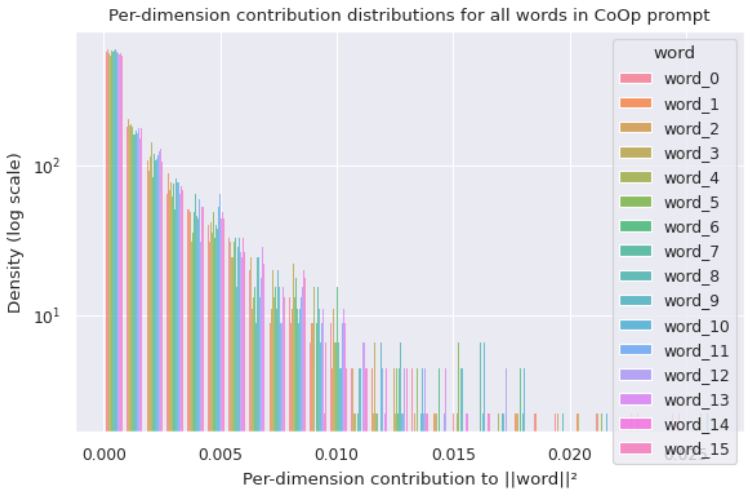


Figure 4: Word embedding dim amplitude

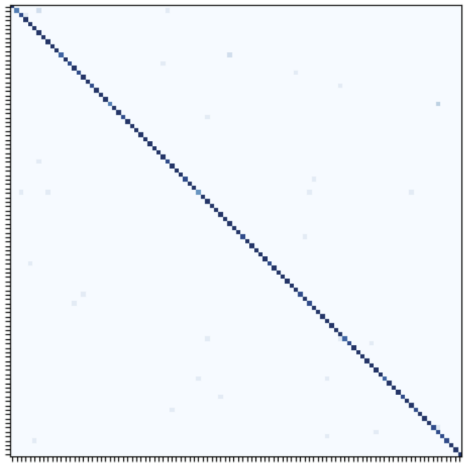


Figure 5: Confusion matrix for Flower102, using MobileCLIP2-S0 with $k = \text{all}$

To assess potential class imbalance effects during validation, we compute both accuracy and macro F1-score. A substantial discrepancy between these metrics would indicate that minority classes are underperforming or overperforming; however, the scores remained closely aligned across experiments, attributable to the balanced sampling enforced by the fixed shot count k. This observation is further supported by the confusion matrix in Figure 5, computed from the `MobileCLIP2-S0` model with a text prompt trained on Flower102 using all available shots ($k = \text{all}$). The matrix is predominantly diagonal, with only sparse, uniformly distributed off-diagonal errors, indicating no systematic bias toward specific classes.

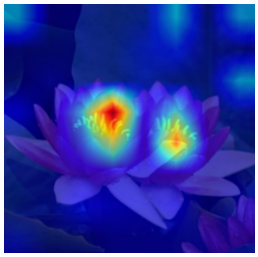
3.2.2 Visual Prompting

Assumptions: Enhancing the text prompt alone does not influence the behavior of the image encoder, as the visual processing pipeline remains unchanged. Because of that, we hypothesized that incorporating a visual prompt could further improve performance by guiding the model's attention toward salient regions of the input image.

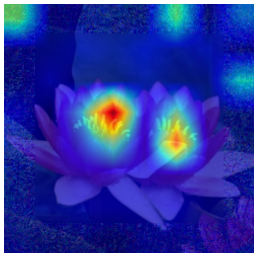
Procedure: Validating this assumption requires examining the model's internal attention patterns to determine whether visual prompting effectively shifts focus toward semantically meaningful regions. Our inspection of attention maps suggests the opposite effect: as training progresses, attention increasingly concentrates near the image borders rather than on the flower itself. Figure 6 illustrates this phenomenon by visualizing the attention distribution of *head* #14 in `trunk.stages.3.blocks.1.token_mixer`, selected due to its strong responsiveness to salient content. Despite being one of the more informative heads, attention remains disproportionately drawn to peripheral regions, indicating that the visual prompt may not consistently enhance object-focused representation as intended, but overfit to the border instead.

In an attempt to address this, reducing the border width to lower the number of parameters rendered it almost entirely ineffective, and increasing weight decay did not meaningfully reduce the overfitting.

A second limitation concerns computational overhead. When applying a visual prompt, all images must be re-encoded to produce new visual embeddings for classification, retrieval, or database matching. Unlike text prompting—where only the text encoder needs to be rerun—visual prompting affects the image stream directly, making any precomputed embeddings unusable. As we move to larger MobileCLIP2 variants, the image encoder increasingly dominates latency, so repeatedly reprocessing entire image collections can become a practical bottleneck.



(a) Without visual prompt



(b) With visual prompt while training

Figure 6: Attention weights heatmap comparison

3.2.3 Multitask training

Assumptions: We hypothesized that if prompt tuning improves classification by better aligning text and image embeddings, then applying the same training strategy to descriptive text should also improve text-to-image retrieval. In this view, a prompt that also enhances class discrimination should yield more informative text embeddings for retrieval. Our experiments did not support this assumption.

Procedure: Retrieval uses cosine similarity between a query text embedding and all image embeddings, selecting the maximum. For multitask experiments, each batch contains B matched text–image pairs. We extract image and text features and form the similarity matrix $\mathbf{S} = \mathbf{I}\mathbf{T}^\top$, whose columns correspond to text-to-image (t2i) and rows to image-to-text (i2t) retrieval. With samples ordered so that the diagonal contains the correct matches, we use labels $\mathbf{y} = (0, 1, \ldots, B-1)$ and define $\mathcal{L}_{\text{t2i}} = \text{CE}(\mathbf{S}, \mathbf{y})$, $\mathcal{L}_{\text{i2t}} = \text{CE}(\mathbf{S}^\top, \mathbf{y})$, and $\mathcal{L}_{\text{ret}} = (\mathcal{L}_{\text{i2t}} + \mathcal{L}_{\text{t2i}})/2$. Because the model only learns to retrieve within a batch, batches must be resampled aggressively, and in-batch accuracy tends to overestimate performance, as retrieval among a small candidate set is comparatively easy.

Optimization: We report top-1 retrieval accuracy. A randomly initialized retrieval prompt reduced Flickr30k accuracy by about 10 percentage points relative to the 74.7% baseline. After training, improvements over the non-prompted model were negligible ($+0.8$ on validation, $+0.0$ on test), effectively matching the no-prompt setting. To avoid the prompt "hiding" behind the descriptive text, we jointly optimized classification and retrieval using

$$\mathcal{L}_{\text{multitask}} = (1-\lambda)\,\mathcal{L}_{\text{ret}} + \lambda\,\mathcal{L}_{\text{cls}}.$$

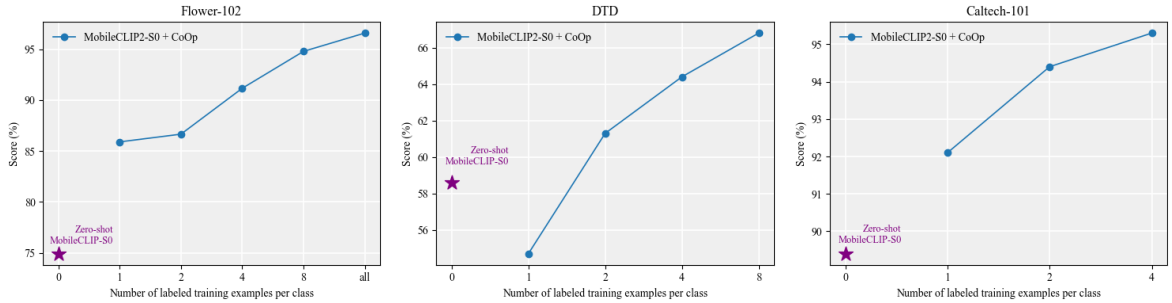
Under this objective, the learned prompt matched the Flower102-only prompt within 0.5 percentage points on classification while still yielding $+0.0$ improvement on Flickr30k retrieval, indicating that it can improve classification without meaningfully affecting retrieval.

4 Results

4.1 Experiments

- Evaluate the model's zero-shot classification performance across a range of benchmarks.
- Keep the model parameters frozen while learning text prompts in order to improve classification accuracy.
- Modify the visual prompt to investigate how the image encoder allocates attention across different regions of the input.
- Assess the scalability of the CoOp method across the entire MobileCLIP2 model family.
- Explore alternative approaches for visualizing and interpreting learned prompts (e.g., by analyzing their proximity to hard tokens and their semantic similarity to hand-crafted prompts).

All experiments were first conducted on the validation set; only after finalizing the setup did we evaluate on the held-out test split. For the smallest model in the family ($S0$), the resulting accuracy improvements on the three datasets are reported below 7.

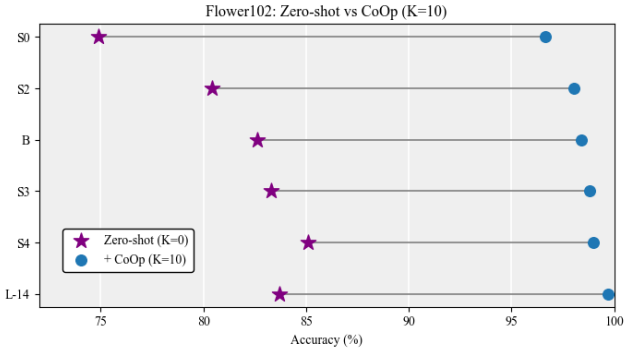


(a) Flower102 (b) DTD (c) Caltech101

Figure 7: MobileCLIP2-S0 improvements with CoOp

When training the larger models, the training time increased substantially. Consequently, we evaluated ($k = all = 10$) for all model sizes, but omitted some other values of k, and for certain models also skipped some datasets.

Our findings on the scalability of the CoOp method with MobileCLIP2 across the model family are summarized in the Figures below 8, where model size increases from top to bottom.



(a) MobileCLIP2-S0 accuracy improvements per model size

MobileCLIP2 Size	Params (M) (img + txt)	Latency (ms) (img + txt)
S0	11.4 + 63.4	1.5 + 3.3
S2	35.7 + 63.4	3.6 + 3.3
B	86.3 + 63.4	10.4 + 3.3
S3	125.1 + 123.6	8.0 + 6.6
S4	321.6 + 123.6	19.6 + 6.6
L/14	304.3 + 123.6	57.9 + 6.6

(b) Params and Latency

Figure 8: MobileCLIP2 family size comparison

5 Discussion

Analyzing the data from the Results section, we observe that the small MobileCLIP2-S0 model can achieve a remarkable increase in accuracy even with only a few training examples when using CoOp. The performance differences between model sizes can inform the choice of an appropriate model for mobile hardware. For instance, it may be worthwhile to choose S2 over S0, as the model size increases by only 32% and latency by 44%, while S2 clearly outperforms S0. A possible explanation is that the S0 image encoder is by far the smallest relative to its text encoder compared with the other models.

Analyzing Figure 8, we observe that although S4 and L-14 have the same model size, the zero-shot accuracy of L-14 is lower than that of S4, whereas with 10 shots L-14 substantially surpasses S4, achieving 99.7

Examining the three misclassified examples, we focus on the unusual case in Figure 9. The dataset labels the image on the right (Figure 9c) as a corn poppy, but compared with the other two images and based on our own assessment, this appears incorrect. Everything indicates that the flower is a Texas bluebell instead.

(a) Corn poppy #1 (b) Corn poppy #2 (c) Sample labeled as corn poppy

Figure 9: Corn poppy class examples, with one atypical instance

Comparing the CoOp accuracy gains of MobileCLIP2-S0 with those of the ViT-B/16-based CLIP reported in the 2024 paper [5], MobileCLIP2 initially appears clearly stronger: it achieves higher zero-shot accuracy while being roughly twice as small and offering lower latency 10. However, once CoOp context is added, it no longer outperforms that model. This suggests either that MobileCLIP2 is particularly effective at exploiting its own internal knowledge in the pure zero-shot setting, or that it is simply better aligned with public benchmark datasets. More broadly, we observe that even when one CLIP model surpasses another in zero-shot accuracy, their relative performance in k-shot regimes cannot be reliably predicted in advance.

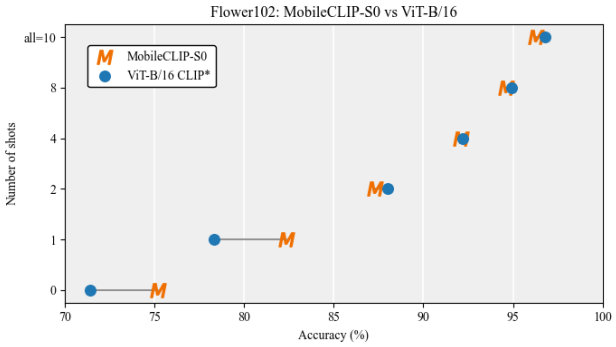


Figure 10: MobileCLIP-S0 vs. ViT-B/16

Furthermore, for a prompt length of 16 soft tokens, we obtain a $+21.6\%$ accuracy improvement on Flower102 (from 74.9% to 96.3%) in the 32 kB setting, with very high robustness. For a prompt of length 1 soft token (2 kB), we achieve a $+19.5\%$ improvement, although the results are less consistent. This suggests that substantial information can in fact be distilled into even smaller prompts. We have not yet investigated the robustness metric in detail so we cannot draw conclusions.

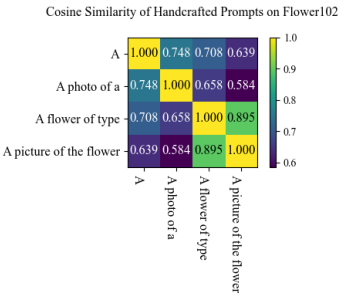
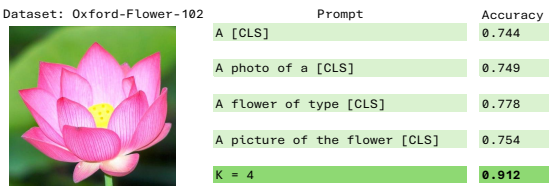


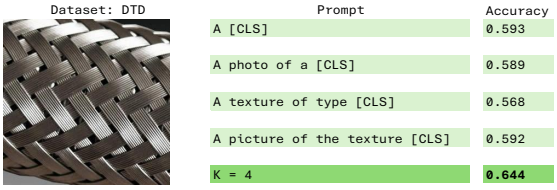
Figure 11: Cosine similarity of handcrafted prompts on Flower102

Naturally, one may ask how these learned prompts translate into actual words. Unfortunately, their nearest interpretations are far from human-readable (see Figure 12). Furthermore, when comparing the meaning of a prompt trained on Flower102 with the handcrafted prompt "A photo of a", by passing both through the text encoder, the cosine similarity is only 0.14. This is quite small; for reference, Figure 11 shows the similarities among several handcrafted prompts.

Given the results above, and as illustrated in Figure 13, a small number of in-domain images appears to be highly valuable, with CoOp providing substantial accuracy gains. Combined with the fact that the learned prompt is compact and can be applied directly at inference time, this makes CoOp an attractive choice for a wide range of applications.



(a) Flower102 prompt comparison



(b) DTD prompt comparison

Figure 13: Handwritten prompts vs. learned

6 Conclusion and Further work

In conclusion, our CoOp-style prompt tuning experiments show that MobileCLIP2 scales effectively with the number of shots used to learn a textual prompt, with all model sizes delivering strong gains relative to their parameter budgets and latency. However, these gains are not fully predictable in advance, which highlights the importance of understanding how each variant scales for a given downstream task. From a practical standpoint, MobileCLIP2 becomes especially compelling for on-device scenarios: a single general-purpose model can be adapted to highly specific tasks with only a handful of labeled examples.

Further work:

One of the main practical challenges was the lack of a local GPU, which forced us to rely on a cloud platform with limited session durations; frequent disconnects made resuming experiments cumbersome and time-consuming. Since prompt training is relatively lightweight, even a modest local GPU would have greatly simplified and accelerated our workflow.

There seems to be considerable room for improving the image prompts and the retrieval, which we did not have time to fully explore, as well as for extending the evaluation to multiple datasets and reporting mean statistics instead. Given the capabilities of MobileCLIP2, these directions appear especially promising for future work.

The current code is highly modular and allows hot-swapping prompts for different experiments. The main source files, in particular the `CoOp_v0_1_1.ipynb` notebook, together with some of the trained prompts, are available on GitHub (`dlvr-28/MobileCLIP2_CoOp`).

Token index	Flower102	DTD
0	kidlit (0.464346)	♫♫ (0.473543)
1	bbcra (0.452249)	♥♡ (0.423275)
2	😑 (0.479293)	(0.458322)
3	travelchat (0.479343)	(0.480258)
4	😱😍 (0.493299)	....@ (0.498091)
5	😊😊 (0.471579)	(0.465939)
6	(0.482176)	swachhb (0.471143)
7	(0.446037)	akshaykumar (0.452874)
8	(0.453129)	(0.452783)
9	(0.478094)	epiconetsy (0.448630)
10	indiegame (0.442174)	(0.474607)
11	ramapho (0.463884)	(0.452654)
12	homeitems (0.471440)	(0.441079)
13	womenintech (0.442392)	□ (0.469392)
14	😉😉 (0.484803)	thismorning (0.467391)
15	ramapho (0.460279)	(0.469835)

Figure 12: Prompts to closest tokens table

References

[1] 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. *arXiv preprint arXiv:2103.00020*, 2021.

[2] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, Ziwei Liu Learning to Prompt for Vision-Language Models *arXiv preprint arXiv:2109.01134*, 2021.

[3] Hyojin Bahng, Ali Jahanian, Swami Sankaranarayanan, Phillip Isola Exploring Visual Prompts for Adapting Large-Scale Models *arXiv preprint arXiv:2203.17274*, 2022.

[4] Fartash Faghri, Pavan Kumar Anasosalu Vasu, Cem Koc, Vaishaal Shankar, Alexander Toshev, Oncel Tuzel, Hadi Pouransari MobileCLIP2: Improving Multi-Modal Reinforced Training *arXiv preprint arXiv:2508.20691*, 2025.

[5] Maxime Zanella, Ismail Ben Ayed Low-Rank Few-Shot Adaptation of Vision-Language Models *arXiv preprint arXiv:2508.20691*, 2025.

[6] Diederik P. Kingma, Jimmy Lei Ba Adam: A Method for Stochastic Optimization *arXiv preprint arXiv:1412.6980*, 2017.
Low-Rank Few-Shot Adaptation of Vision-Language Models