Prime Once, then Reprogram Locally: An Efficient Alternative to Black-Box Service Model Adaptation

Supplementary Materials

This appendix provides comprehensive details supporting the main paper. Section 7 elaborates on model reprogramming techniques, while Section 8 presents detailed experimental configurations, including dataset statistics and implementation specifics for all baselines and our AReS method (Section 8.3). Section 9 details the complete theoretical analysis establishing performance bounds between service and local models. Additional experimental results are reported in Section 10, covering few-shot VM experiments (Section 10.1), local model enhancement analysis (Section 10.2), evaluations on **real-world proprietary APIs** such as [1]GPT-4o and [2]Clarifai (Section 10.8), and a detailed analysis of challenging VLM scenarios (Section 10.7). Section 11 offers further discussion on our contributions and design rationale. Finally, Section 12 discusses the limitations of our approach.

7. Details of Model Reprogramming

Model Reprogramming (MR) [3, 11, 25, 71, 76] is a technique that adapts pre-trained models, such as a source model $\mathcal{F}_S$, to new tasks without altering their underlying parameters. This is particularly useful when dealing with powerful models accessed as a service (MaaS) where direct modification is not possible. The core idea involves an input transformation function g_{in} that adapts inputs x^T from the downstream task domain $\mathcal{X}^T$ to the source domain $\mathcal{X}^S$ through a learnable prompt $\mathbf{P}$. Additionally, an output or label mapping function g_{out} is employed to bridge the source label space $\mathcal{Y}^S$ and the target label space $\mathcal{Y}^T$. This approach aims to leverage the rich features learned by large pre-trained models for new, potentially resource-scarce tasks, by essentially "tricking" the model into performing a different function through these carefully crafted input and output manipulations.

7.1. Input Transformation

Input transformation [41, 57, 58, 76] in Model Reprogramming, denoted as $g_{\text{in}}(x^T|\mathbf{P})$, serves to bridge the domain gap between the downstream task and the source task the original model $\mathcal{F}_S$ was trained on. It introduces a learnable prompt or program $\mathbf{P}$ that modifies the downstream task inputs x^T before they are fed to $\mathcal{F}_S$, aiming to make the downstream data compatible with its input expectations. Common approaches in Visual Reprogramming (VR) include padding-based VR, which adds trainable noise patterns to the outer frames of an image while preserving its integrity, and watermarking-based VR, where trainable noise patterns are overlaid directly onto the input images. In the context of BAR [57], the input transformation takes the form of an "adversarial program" $P = \tanh(W \odot M)$, where W represents learnable parameters and M is a binary mask ensuring that the original embedded target data remains unchanged. This adversarial program P is universal to all target data samples and is learned to make the model $\mathcal{F}_S$ produce outputs that can be mapped to the desired target task labels. The learning of these parameters, especially in closed-box settings, often relies on zeroth-order optimization techniques.

7.2. Output Mapping

Output mapping [6, 8, 27, 57], g_{out}, is a crucial step that translates the predictions from the pre-trained model's original label space $\mathcal{Y}^S$ to the downstream task's label space $\mathcal{Y}^T$, which are often distinct. This process is frequently designed to be gradient-free, meaning it does not introduce additional trainable parameters requiring backpropagation, thus preserving efficiency, especially for tasks with large label spaces. Existing gradient-free methods like Random Label Mapping (RLM) [14], Frequent Label Mapping (FLM) [57], and Iterative Label Mapping (ILM) [8] typically establish a one-to-one correspondence. However, such one-to-one mappings can be limiting as they may overlook more complex, potentially many-to-many relationships between pre-trained and downstream labels. To address this, probabilistic and multi-label mapping strategies have been developed. Bayesian-guided Label Mapping (BLM), for example, constructs a probabilistic matrix quantifying pairwise relationships between pre-trained and downstream labels using Bayesian conditional probability, allowing for a flexible many-to-many mapping. Similarly, BAR [57] can utilize multi-label mapping where multiple source labels map to a single target label, often determined by a frequency-based scheme from initial predictions.

[1] https://platform.openai.com/docs/models/gpt-4o
[2] https://www.clarifai.com/

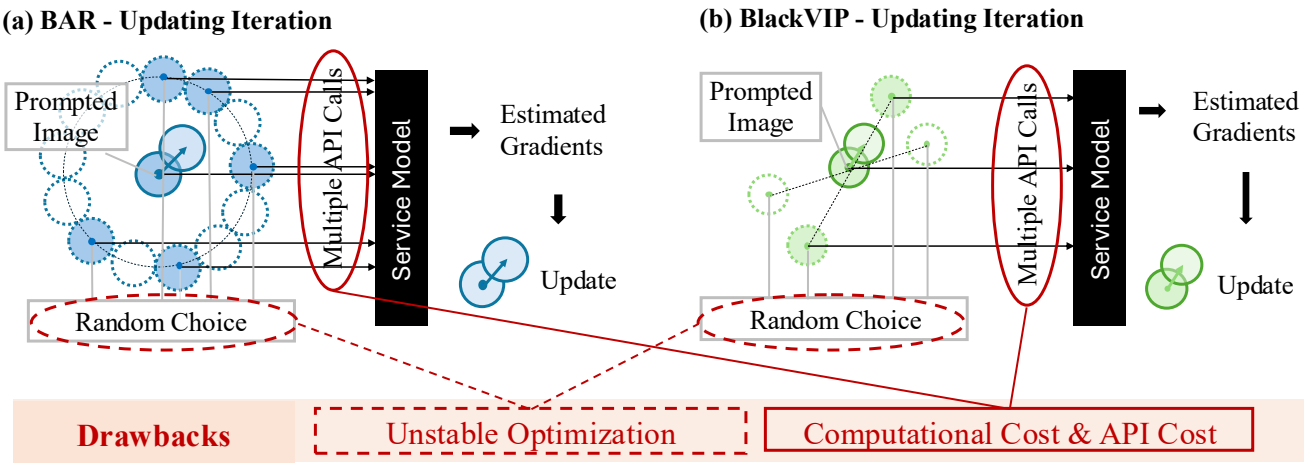

Figure 4. Illustration of Zeroth-Order Optimization (ZOO) techniques commonly used in closed-box model reprogramming. **(a)** BAR with Randomized Gradient-Free (RGF) method estimates gradients by querying the model with random directional perturbations. **(b)** BlackVIP with Simultaneous Perturbation Stochastic Approximation with Gradient Correction (SPSA-GC) approximates gradients using only two model queries with a randomly generated perturbation vector. Both approaches suffer from high query complexity and noisy gradient estimates, leading to unstable and computationally intensive optimization, especially for high-dimensional prompts.

The nature of g_{out} can differ for VMs and VLMs. For VMs, if label spaces differ, an explicit g_{out} like BLM is necessary, and in the AReS framework, priming for VMs occurs in the service model's label probability space. For VLMs, g_{out} might be an identity mapping if label spaces align, or the VLM's text encoder can naturally transform outputs to the downstream task label space.

7.3. Training

The training process in model reprogramming focuses on learning the parameters of the input transformation $g_{\text{in}}(\cdot|P)$, such as the adversarial program P, while the pre-trained source model $\mathcal{F}_S$ remains frozen. The objective is to minimize a loss function on the downstream task data using the transformed inputs and mapped outputs. In a glass-box setting, where gradients from $\mathcal{F}_S$ are accessible, standard gradient-based optimization can efficiently update the transformation parameters. Our AReS method leverages this by first priming a local, fully accessible model $\mathcal{F}_L$, then performing efficient glass-box prompt learning. Conversely, when $\mathcal{F}_S$ is a closed-box, Zeroth-Order Optimization (ZOO) techniques are employed. These methods estimate gradients using only the model's input-output responses, for instance, through random gradient-free (RGF) optimization [31, 60], Simultaneous Perturbation Stochastic Approximation with Gradient Correction (SPSA-GC) [51, 52], as shown in Fig. 4 and Fig. 5, closed-box training presents significant difficulties: optimization can be unstable due to noisy gradient estimates; it incurs high computational costs and numerous API calls; it requires continuous API dependency for both training and inference; and the performance gains are often uncertain despite the substantial investment. Our AReS approach aims to circumvent these challenges by shifting the reprogramming to a locally primed model.

8. Experimental Setting

8.1. Datasets

Datasets. We evaluate our method on ten diverse datasets commonly used in transfer learning literature. These datasets span various domains including fine-grained categorization (Flowers102 [40], StanfordCars [28]), texture recognition (DTD [12]), action recognition (UCF101 [50]), food classification (Food101 [4]), traffic sign recognition (GTSRB [20]), satellite imagery (EuroSAT [17]), animal recognition (OxfordPets [44]), scene classification (SUN397 [67]), and digit recognition (SVHN [38]). Following established protocols in prior work [41], we adopt a few-shot learning setup with 16 randomly selected training examples per class, using the entire test set for evaluation when the service model is VLM. And full-shot learning is applied when the service model is VM, consistent with [57]. Table 7 provides detailed statistics for each dataset.

Table 7. Detailed dataset information.

Dataset	Full-shot Training	16-shot Training	Testing Set Size	Number of Classes
Flowers102	4,093	1,632	2,463	102
DTD	2,820	752	1,692	47
UCF101	7,639	1,616	3,783	101
Food101	50,500	1,616	30,300	101
GTSRB	39,209	688	12,630	43
EuroSAT	13,500	160	8,100	10
OxfordPets	2,944	592	3,669	37
StanfordCars	6,509	3,136	8,041	196
SUN397	15,888	6,352	19,850	397
SVHN	73,257	160	26,032	10

8.2. Experimental Scope and Rationale

Our focus on image classification is a deliberate choice made to ensure a direct and rigorous comparison with the established art in closed-box Model Reprogramming (BMR) [41, 57]. The primary methods in this field, including our main baselines (BAR [57], BlackVIP [41], LLM-Opt [33]) and other related works [15, 29, 42, 64, 73] shown in Table 1, are all benchmarked on classification tasks for both standard Vision Models (VMs) and Vision-Language Models (VLMs). Adhering to this established protocol allows for a fair and robust evaluation of our framework, which is designed to provide an efficient reprogramming solution for both model types. Furthermore, classification remains a challenging and relevant testbed; recent work has shown that powerful Multimodal Large Language Models (MLLMs) e.g., LLaVA [30] often suffer from catastrophic forgetting, failing to retain the full classification performance of their underlying visual towers [75]. This highlights the non-trivial nature of adapting these models for pure classification and validates the importance of our approach.

While extending AReS to generative tasks like Visual Question Answering (VQA) is a valuable direction for future research, such tasks are outside the scope of this initial investigation. A core goal of our work is to support standard vision classifiers like ViT and ResNet, for which open-ended generative tasks are not directly applicable. By demonstrating AReS's effectiveness on the fundamental task of classification—a common ground for both VMs and VLMs—we build a strong and reliable foundation for future extensions into more complex, multimodal reasoning tasks.

8.3. Baselines and Implementation Details

Our experiments compare AReS against three primary closed-box visual reprogramming baselines: BAR, BlackVIP, and LLM-Opt.

BAR [57][3] repurposes closed-box models by learning a universal adversarial program that is added to target inputs. It relies on Randomized Gradient-Free (RGF) optimization, based on input-output responses, and employs a multi-label mapping scheme. For our VMs experiments, we utilize BLM[4] for BAR to ensure a fair comparison with other methods, including our own, which also uses BLM. We implemented BAR by referencing its official codebase and BlackVIP's reported BAR implementation, adhering to the hyperparameters detailed in BlackVIP's appendix[5] for stable convergence. BAR uses focal loss as its learning objective.

BlackVIP [41][6] enhances closed-box visual prompting by introducing a Coordinator network to generate input-dependent visual prompts and employs Simultaneous Perturbation Stochastic Approximation with Gradient Correction (SPSA-GC) for optimization. We use the official BlackVIP codebase and strictly adhere to the hyperparameter settings reported in their paper and detailed in their configuration files, which cover learning rates and SPSA-GC specific parameters. BlackVIP utilizes cross-entropy loss. For adapting VLMs, all methods, including BlackVIP and BAR, are evaluated in a 16-shot learning setting (16 training samples per class), consistent with the setup in the BlackVIP paper.

LLM-Opt [33][7] utilizes a large language model (e.g., GPT-4) as a closed-box optimizer to find effective text prompts for a

[3] https://github.com/yunyuntsai/black-box-Adversarial-Reprogramming
[4] https://github.com/tmlr-group/BayesianLM
[5] https://github.com/changdaeoh/BlackVIP/blob/main/docs/configuration.md
[6] https://github.com/changdaeoh/BlackVIP
[7] https://llm-can-optimize-vlm.github.io

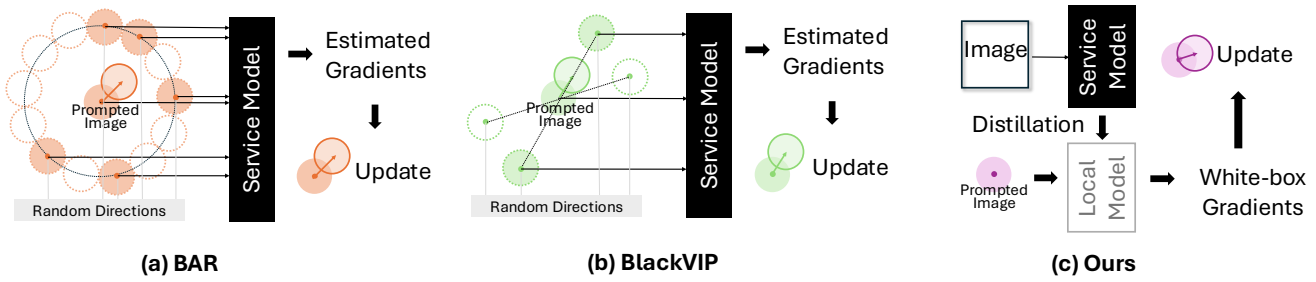

Figure 5. Comparison of closed-box visual reprogramming approaches. **(a)** BAR and **(b)** BlackVIP rely on Zeroth-Order Optimization (ZOO) by repeatedly querying the service model with perturbed inputs (e.g., using random directions) to estimate gradients for updating the visual prompt. These methods suffer from high API call costs and potentially unstable optimization. **(c)** Our AReS method performs a one-time priming from the service model to a local model. Subsequent visual prompt optimization occurs efficiently on this local model using glass-box gradients, eliminating further API calls and enabling stable, cost-effective adaptation.

target VLM. The method employs an automated "hill-climbing" procedure, where it provides the LLM optimizer with both high- and low-performing prompts as textual feedback to guide the search for better candidates [33]. However, this approach has two significant limitations. First, it introduces a second, costly API dependency for the LLM optimizer, which can amount to over $100 in API fees for a single experimental run [33]. Second, because the method exclusively optimizes the text prompt, its applicability is restricted to VLMs and it cannot be used to adapt standard VMs, which are a key focus of our work.

AReS, our proposed method, the visual reprogramming (VR) components are implemented leveraging the publicly available codebase of BLM. The priming stage, transferring from the service API to the local model's classification head, is performed for 100 epochs. The subsequent glass-box VR on the primed local model is conducted for 200 epochs. Both stages use the Adam optimizer with learning rates of 0.001 for KD and 0.01 for local VR, respectively. The local VR part of AReS employs cross-entropy loss. AReS follows the 16-shot protocol for VLM experiments and operates in a full-shot setting for VM experiments. Following [41], pre-trained Vision Transformer (ViT) models and encoders used in our experiments are sourced from `timm` [66], ResNet models are from `PyTorch` [45], and CLIP models are from the official `CLIP` repository [48].

8.4. Clarification of VLM and VM Evaluation Setups

Our experiments distinguish between two primary evaluation setups: one for Vision-Language Models (VLMs) and another for standard Vision Models (VMs). The core differences lie in how each model type handles the output space of the downstream task and the data settings used for evaluation, which were chosen to ensure fair and direct comparisons with established baselines.

VLM Evaluation Setup. In our experiments, the VLM setup primarily involves models like CLIP, which is consistent with the standard set by our main baseline, BlackVIP. The key characteristics are:

- **Output Handling:** VLMs utilize a text encoder to perform zero-shot classification without a fixed output head. This means their output space is naturally aligned with the downstream task's text labels. Consequently, a separate or complex label mapping mechanism is not required, simplifying the adaptation process.
- **Data Setting:** To ensure a direct comparison, the evaluation follows BlackVIP's established protocol, which uses a **16-shot** learning setting. The closed-box model reprogramming for all compared methods is learned using only these few-shot labeled samples.

VM Evaluation Setup. The VM setup uses standard, pre-trained vision models with a fixed output vocabulary, such as an ImageNet-pretrained ViT-B/16 or ResNet101. This setup presents a more significant challenge:

- **The Challenge of Label Space Mismatch:** Unlike VLMs, these models face a fundamental label space mismatch. Their fixed output vocabulary (e.g., 1000 ImageNet classes) does not align with the labels of the downstream task (e.g., Flowers102). This is a "non-trivial problem" that prior work like BlackVIP explicitly avoided.
- **AReS's Solution:** Our framework directly solves this challenge by incorporating Bayesian-guided Label Mapping (BLM) to bridge the source and target label spaces. In this setup, AReS's priming stage occurs within the service model's source label space, as it has no inherent knowledge of the target class names.
- **Data Setting:** Following the standard for BMR on vision models established by BAR, these experiments are conducted in a **full-shot** setting. For completeness, a few-shot VM setting was also explored in our appendix.

9. Details of Theoretical Analysis

In this section, we provide a theoretical analysis to understand the effectiveness of our proposed method, AReS. Our framework involves two primary components affecting the final performance on the downstream task: priming from the service model $\mathcal{F}_S$ to the local encoder $\mathcal{F}_L$, and visual reprogramming (VR) applied to the local model using prompt $\mathbf{P}$. Our analysis is related to the broader study of representation transferability via task-relatedness [37], which provides formal tools for understanding when and why transferring representations between models is effective. We begin our analysis by considering the scenario where the model's output logits are directly aligned with the target labels, effectively assuming the label mapping g_{out} is an identity function due to the text encoder.

Notations: Let $\mathcal{D}^T$ represent the downstream task data distribution $p(x, y)$ over inputs $x \in \mathcal{X}^T$ and labels $y \in \mathcal{Y}^T$. Let $\mathcal{F}_S : \mathcal{X}^S \to \mathbb{R}^{K^S}$ be the closed-box service model and $\mathcal{F}_L : \mathcal{X}^S \to \mathcal{Z}$ be the local surrogate model. After priming, $\mathcal{F}_L$ includes the learned linear layer mapping inputs to the downstream task's logit space $\mathcal{Z} = \mathbb{R}^{K^T}$. The visual reprogramming involves an input transformation $g_{\text{in}}(x, p)$ parameterized by p. $\ell : \mathcal{Z} \times \mathcal{Y}^T \to \mathbb{R}_{\geq 0}$ is cross-entropy loss.

Let $p_S(x)$ and $p_L(x)$ denote the probability distributions generated by the service model $\mathcal{F}_S$ and the local model $\mathcal{F}_L$ for an input x, respectively. For the purpose of this theoretical analysis, we introduce logits. Let $z_S(x)$ and $z_L(x)$ be the logits produced by the service model $\mathcal{F}_S$ and the local model $\mathcal{F}_L$ for input x *before* visual reprogramming. It is important to note that this assumption of direct logit access from $\mathcal{F}_S$ is made for theoretical tractability and differs from the practical setting in the main paper, where only probabilities are assumed to be accessible from the service model. Let $z_S^*(x, \mathbf{Q}^*) = \mathcal{F}_S(g_{\text{in}}(x, \mathbf{Q}^*))$ and $z_L^*(x, \mathbf{P}^*) = \mathcal{F}_L(g_{\text{in}}(x, \mathbf{P}^*))$ be the logits produced by the service model and local model, respectively, for input x *after* visual reprogramming with their respective optimal prompts $\mathbf{Q}^*$ and $\mathbf{P}^*$.

The downstream risk for the local model $\mathcal{F}_L$ *without* VR is:

$$\mathcal{R}_L(\mathcal{D}^T) := \mathbb{E}_{(x,y)\sim\mathcal{D}^T}[\ell(z_L(x), y)]$$

The downstream risk for the local model $\mathcal{F}_L$ *after* applying VR with its optimal prompt $\mathbf{P}^*$ is:

$$\mathcal{R}_L(\mathcal{D}^T, \mathbf{P}^*) := \mathbb{E}_{(x,y)\sim\mathcal{D}^T}[\ell(z_L^*(x, \mathbf{P}^*), y)]$$

Similarly, for the service model $\mathcal{F}_S$, the downstream risk *without* VR is:

$$\mathcal{R}_S(\mathcal{D}^T) := \mathbb{E}_{(x,y)\sim\mathcal{D}^T}[\ell(z_S(x), y)]\,.$$

And the downstream risk for the service model $\mathcal{F}_S$ *after* applying VR with its optimal prompt $\mathbf{Q}^*$ is:

$$\mathcal{R}_S(\mathcal{D}^T, \mathbf{Q}^*) := \mathbb{E}_{(x,y)\sim\mathcal{D}^T}[\ell(z_S^*(x, \mathbf{Q}^*), y)]\,.$$

Definitions and Assumptions. We introduce the following definitions and assumptions for our theoretical analysis.

Definition 1. (ϵ-Faithful Priming). The priming from $\mathcal{F}_S$ to $\mathcal{F}_L$ is considered ϵ-faithful if the expected L_1 norm of the difference between their output logits is bounded by $\epsilon \geq 0$, both before and after applying their respective optimal visual reprogramming prompts:

$$\mathbb{E}_{(x,y)\sim\mathcal{D}^T}[\|z_S(x) - z_L(x)\|_1] \leq \epsilon$$

and

$$\mathbb{E}_{(x,y)\sim\mathcal{D}^T}[\|z_S^*(x, \mathbf{Q}^*) - z_L^*(x, \mathbf{P}^*)\|_1] \leq \epsilon\,.$$

This implies that the priming is sufficiently effective such that the logit distributions of the service and local models are closely aligned. A smaller ϵ indicates a more faithful priming.

Assumption 1. *(Service Model Superiority). The service model $\mathcal{F}_S$ is inherently more powerful or better aligned with the downstream task distribution $\mathcal{D}^T$ compared to the local model $\mathcal{F}_L$, both before and after optimal visual reprogramming. Formally:*

$$\mathcal{R}_S(\mathcal{D}^T) \leq \mathcal{R}_L(\mathcal{D}^T)$$

and

$$\mathcal{R}_S(\mathcal{D}^T, \mathbf{Q}^*) \leq \mathcal{R}_L(\mathcal{D}^T, \mathbf{P}^*)\,.$$

This is a natural assumption that motivates the use of the service model.

Assumption 2. *(ϵ-Faithful Priming). The priming process is effective, resulting in the local model $\mathcal{F}_L$ closely mimicking the logit distributions of the service model $\mathcal{F}_S$, both immediately after priming (before VR) and after both models have been optimally reprogrammed for the downstream task.*

9.1. Service and Local Model Performance Difference Bound

This section establishes the theoretical foundation for understanding the performance relationship between service and local models in our AReS framework. We begin by proving that the cross-entropy loss function exhibits Lipschitz continuity with respect to logit differences (Lemma 1), which provides the mathematical basis for our subsequent analysis. Building on this property, we derive performance difference bounds (Lemma 2) that quantify how faithful priming affects the performance gap between models. Finally, we establish tight bounds on service model performance relative to the primed local model (Theorem 3), demonstrating that faithful priming enables the local model to closely approximate the service model's capabilities both before and after optimal visual reprogramming.

Lemma 1. *(Lipschitz Continuity of Cross-Entropy Loss with respect to Logits). The cross-entropy loss function* $\ell(z, y) = -\sum_{j=1}^{K^T} y_j \log(p_j(z))$*, where* $p_j(z)$ *are softmax probabilities derived from logits* $z \in \mathbb{R}^{K^T}$ *and* y *is a one-hot true label vector, is Lipschitz continuous with respect to the logits* z*. The specific Lipschitz constant depends on the norm used to measure the difference between logit vectors. Specifically:* $\ell(z, y)$ *is 1-Lipschitz with respect to the* L_1 *norm of the logits:*

$$|\ell(z_1, y) - \ell(z_2, y)| \leq \|z_1 - z_2\|_1 ,$$

for any two logit vectors z_1, z_2*. These constants do not explicitly depend on the number of classes* K^T *(for* $K^T \geq 2$*).*

Proof. The gradient of the cross-entropy loss $\ell(z, y)$ with respect to the logits z is given by:

$$\nabla_z \ell(z, y) = p(z) - y ,$$

where $p(z)$ is the vector of softmax probabilities derived from z, and y is the one-hot true label vector. A differentiable function $f(x)$ is L-Lipschitz with respect to a norm $\|\cdot\|_p$ if the dual norm $\|\cdot\|_q$ of its gradient is bounded by L (i.e., $\|\nabla f(x)\|_q \leq L$), by Hölder's inequality. For the L_1 norm of logits ($p = 1$, dual norm $q = \infty$): We examine the L_∞ norm of the gradient:

$$\|\nabla_z \ell(z, y)\|_\infty = \|p(z) - y\|_\infty = \max_{j=1,\ldots,K^T} |p_j(z) - y_j| .$$

Let k be the index of the true class ($y_k = 1$, and $y_j = 0$ for $j \neq k$). Then $|p_k(z) - 1| = 1 - p_k(z) \leq 1$. For $j \neq k$, $|p_j(z) - 0| = p_j(z) \leq 1$. Thus, $\|\nabla_z \ell(z, y)\|_\infty \leq 1$. This establishes $L = 1$.

□

Lemma 2. *(Performance Difference Bound due to Priming Faithfulness). Under Assumption 2 (*ϵ*-Faithful Priming), the difference in performance between the local model* $\mathcal{F}_L$ *and the service model* $\mathcal{F}_S$ *is bounded by* ϵ *as follows:*

1. *Before visual reprogramming:*
$$\mathcal{R}_L(\mathcal{D}^T) - \mathcal{R}_S(\mathcal{D}^T) \leq \epsilon ;$$
2. *After optimal visual reprogramming with prompts* $\mathbf{P}^*$ *for* $\mathcal{F}_L$ *and* $\mathbf{Q}^*$ *for* $\mathcal{F}_S$*:*
$$\mathcal{R}_L(\mathcal{D}^T, \mathbf{P}^*) - \mathcal{R}_S(\mathcal{D}^T, \mathbf{Q}^*) \leq \epsilon .$$

Proof. We will prove the first part of the lemma regarding performance before visual reprogramming. The proof for the second part (after optimal visual reprogramming) follows an identical structure, applying the arguments to $z_L^*(x, \mathbf{P}^*)$ and $z_S^*(x, \mathbf{Q}^*)$ and using the corresponding condition from Assumption 2.

The difference in downstream risks between the local model $\mathcal{F}_L$ and the service model $\mathcal{F}_S$ is:

$$\begin{aligned}\mathcal{R}_L(\mathcal{D}^T) - \mathcal{R}_S(\mathcal{D}^T) &= \mathbb{E}_{(x,y)\sim\mathcal{D}^T}[\ell(z_L(x), y)] - \mathbb{E}_{(x,y)\sim\mathcal{D}^T}[\ell(z_S(x), y)] \\ &= \mathbb{E}_{(x,y)\sim\mathcal{D}^T}[\ell(z_L(x), y) - \ell(z_S(x), y)] .\end{aligned}$$

By Lemma 1, the cross-entropy loss $\ell(z, y)$ is 1-Lipschitz continuous with respect to the L_1 norm of its logit inputs. Therefore, for any specific sample (x, y):

$$\ell(z_L(x), y) - \ell(z_S(x), y) \leq |\ell(z_L(x), y) - \ell(z_S(x), y)| \leq \|z_L(x) - z_S(x)\|_1 .$$

Taking the expectation over the data distribution $\mathcal{D}^T$:

$$\mathbb{E}_{(x,y)\sim\mathcal{D}^T}[\ell(z_L(x), y) - \ell(z_S(x), y)] \leq \mathbb{E}_{(x,y)\sim\mathcal{D}^T}[\|z_L(x) - z_S(x)\|_1] .$$

Table 8. Accuracy and Efficiency comparison using RN101 (Service) in Full-shot setting.

Method	Flowers	DTD	UCF	Food	GTSRB	EuroSAT	Pets	Cars	SUN	SVHN	Avg.	#API (M)	Time (h)
VR (glass-box)	42.3	44.4	34.4	25.4	53.1	85.0	73.9	5.0	20.5	75.1	45.9	43.4	21.9
BAR	16.9	25.3	30.2	13.9	22.5	45.7	28.7	1.5	10.7	42.1	23.8	1,724.2	313.5
BlackVIP (RN50)	21.7	39.6	31.4	14.9	26.8	68.3	61.4	3.5	12.3	42.6	32.3	2,586.2	320.5
AReS (RN50)	**41.3**	**42.3**	**37.2**	**25.3**	**48.4**	**84.3**	**73.7**	**5.1**	**19.7**	**72.9**	**45.0**	**0.2**	**8.7**
BlackVIP (ViT-B/32)	26.7	37.7	30.2	13.3	33.8	67.3	57.9	3.0	14.5	34.5	31.9	2,586.2	418.0
AReS (ViT-B/32)	**51.3**	**45.8**	**41.5**	**30.2**	**57.9**	**94.9**	**63.3**	**5.1**	**25.6**	**83.6**	**49.9**	**0.2**	**22.3**

From Assumption 2 (ϵ-Faithful Priming), we have the condition that $\mathbb{E}_{(x,y)\sim\mathcal{D}^T}[\|z_S(x) - z_L(x)\|_1] \leq \epsilon$. Substituting this into the inequality:

$$\mathcal{R}_L(\mathcal{D}^T) - \mathcal{R}_S(\mathcal{D}^T) \leq \epsilon\,.$$

□

Theorem 3. *(Bounding Service Model Performance). Combining Lemma 2 with Assumption 1 (Service Model Superiority), we can establish bounds on the performance of the service model $\mathcal{F}_S$ relative to the primed local model $\mathcal{F}_L$:*

1. *Before visual reprogramming:*

$$\mathcal{R}_L(\mathcal{D}^T) - \epsilon \leq \mathcal{R}_S(\mathcal{D}^T) \leq \mathcal{R}_L(\mathcal{D}^T)\,; \tag{4}$$

2. *After optimal visual reprogramming:*

$$\mathcal{R}_L(\mathcal{D}^T, \mathbf{P}^*) - \epsilon \leq \mathcal{R}_S(\mathcal{D}^T, \mathbf{Q}^*) \leq \mathcal{R}_L(\mathcal{D}^T, \mathbf{P}^*)\,. \tag{5}$$

Proof. By Lemma 2, we have $\mathcal{R}_L(\mathcal{D}^T) - \mathcal{R}_S(\mathcal{D}^T) \leq \epsilon$, which rearranges to $\mathcal{R}_L(\mathcal{D}^T) - \epsilon \leq \mathcal{R}_S(\mathcal{D}^T)$. From Assumption 1, we have $\mathcal{R}_S(\mathcal{D}^T) \leq \mathcal{R}_L(\mathcal{D}^T)$. Combining these two inequalities gives:

$$\mathcal{R}_L(\mathcal{D}^T) - \epsilon \leq \mathcal{R}_S(\mathcal{D}^T) \leq \mathcal{R}_L(\mathcal{D}^T)\,.$$

The reprogrammed case can be shown similarly. □

The above theoretical analysis provides solid justification for AReS's effectiveness by showing that faithful knowledge priming (ϵ-faithful) directly translates to bounded performance differences between service and local models. Theorem 3's key insight reveals that traditional closed-box methods expend significant computational resources attempting to optimize the service model's performance directly, while AReS transforms this challenge by first achieving faithful priming (small ϵ) and then efficiently optimizing the local model using stable first-order methods. This theoretical framework validates our empirical findings and explains why AReS can achieve competitive performance with dramatically reduced API calls and computational costs.

10. Additional Experimental Results

10.1. VM as a Service Model in Few-Shot Setting

Table 9 details the performance of VM adaptation using ViT-B/16 as the service model and ViT-B/32 as the local model within a challenging 16-shot per class setting. This contrasts with the full-shot experiments for VMs presented in the main paper (Tables 8 and 3), which are generally preferred for VMs due to their more limited generalization compared to VLMs like CLIP. These few-shot results, however, provide a valuable benchmark for performance in data-scarce conditions. As anticipated, all evaluated methods show a decline in performance relative to the full-shot scenario. Notably, the closed-box reprogramming methods BAR and BlackVIP achieve average accuracies of only 14.6% and 25.9%, respectively. Despite this challenging low-data regime, our AReS method demonstrates superior performance, achieving an average accuracy of 32.8%. This represents a significant average improvement of +18.2 percentage points over BAR and +6.9 percentage points over BlackVIP. This consistent outperformance, even with extremely limited training data, underscores AReS's effectiveness in robustly transferring knowledge from closed-box service models and significantly enhancing local model reprogramming capabilities across diverse datasets and varying data availability.

Table 9. Accuracy comparison using ViT-B/16 (Service) and ViT-B/32 (Local) in 16-shot setting.

Method	Flowers	DTD	UCF	Food	GTSRB	EuroSAT	Pets	Cars	SUN	SVHN	Avg.
VR (glass-box)	55.0	44.7	42.0	18.0	17.4	70.6	70.7	5.7	29.6	28.4	38.2
BAR	9.7	16.3	23.5	8.1	4.8	34.2	21.5	1.0	7.6	19.6	14.6
BlackVIP	16.1	**36.1**	33.8	12.5	9.1	54.9	54.5	3.9	22.1	16.2	25.9
AReS (Ours)	**46.0**	35.5	**34.3**	**12.7**	**19.6**	**71.1**	**54.3**	**4.2**	**23.0**	**27.3**	**32.8**

Table 10. Accuracy comparison between Local VR (ViT-B/16) and our method on VLMs using CLIP ViT-B/16 (Service) and ViT-B/16 (Local) in a 16-shot setting.

Method	Flowers	DTD	UCF	Food	GTSRB	EuroSAT	Pets	Cars	SUN	SVHN	Avg.
Zero-shot	71.3	43.9	66.9	**85.9**	21.0	47.9	**89.1**	**65.2**	62.6	17.9	57.2
Local VR	55.0	44.7	42.0	18.0	17.4	70.6	70.7	5.7	29.6	28.4	38.2
AReS (Ours)	**86.6**	**48.2**	**67.1**	68.8	**39.4**	**85.7**	88.9	43.2	**62.8**	**63.2**	**65.4**

Table 11. Accuracy comparison between Local VR and our method on VMs in Full-shot setting.

Service	**Local**	**Local VR**	**Ours**
ViT-B/16	ViT-B/32	45.3	**50.4**
	RN50	43.9	**45.9**
RN101	ViT-B/32	45.3	**49.9**
	RN50	43.9	**45.0**

Table 12. Comparison of trainable parameters and accuracy on the EuroSAT dataset. More parameters do not correlate with better performance for ZOO-based methods.

Method	**# Trainable Params**	**Accuracy (%)**
VP w/ SPSA-GC	69K	70.9
BAR	37K	77.3
BlackVIP	9K	73.3
AReS (Ours)	**21K**	**85.7**

10.2. AReS's Impact on Local Model Performance

To quantify AReS's enhancement of local model capabilities, we compare it against directly performing glass-box VR on the local model (termed "Local VR"). For AReS, we assume a pre-trained local encoder, consistent with methods like BlackVIP. For this evaluation of the local encoder only, we additionally assume a pre-trained linear layer on the source domain. As shown in Table 10 for VLMs (CLIP ViT-B/16 service, ViT-B/16 local, 16-shot), AReS achieves a 65.4% average accuracy, a substantial $+27.2\%$ improvement over Local VR's 38.2%. This highlights that AReS's priming significantly boosts the local ViT-B/16's reprogrammability beyond its standalone capacity.

A similar advantage for AReS is observed with VMs in the full-shot setting, as detailed in Table 11. For instance, with a ViT-B/16 service model, AReS improves upon Local VR by $+5.1\%$ (50.4% vs. 45.3%) when using a ViT-B/32 local model, and by $+2.0\%$ (45.9% vs. 43.9%) with an RN50 local model. Consistent gains are also seen with an RN101 service model. These results across both VLM and VM configurations underscore the critical role of AReS's initial priming phase. By effectively transferring knowledge from the more powerful service model, AReS significantly elevates the local model's baseline performance and its potential for reprogramming, demonstrating a more effective utilization of combined model strengths.

10.3. Dissecting the Source of Performance Gain

The superior performance of AReS stems not from a higher parameter count or the mere inclusion of local tuning, but from its novel two-stage framework. We demonstrate that the initial priming stage is an indispensable component and that the overall design is highly parameter-efficient.

Table 13. Comparison with an enhanced BlackVIP baseline on the EuroSAT dataset. The results confirm that AReS's performance gain is primarily from its superior two-stage framework.

Method	Accuracy (%)
Local VR	70.6
BlackVIP	73.3
BlackVIP w/ Local Tuning	74.1
AReS (Ours)	**85.7**

Table 14. Accuracy (%) comparison on the EuroSAT dataset across different few-shot settings. AReS consistently outperforms the strong local baseline (Local VR+LP) as data availability increases.

Method	4-shot	8-shot	16-shot	32-shot
BlackVIP	69.3	71.7	73.3	72.9
Local VR+LP	59.7	68.1	80.1	86.5
AReS (Ours)	**66.1**	**73.4**	**85.7**	**91.6**

First, an analysis of parameter efficiency reveals that for ZOO-based methods, a larger number of trainable parameters does not guarantee better performance and can even be detrimental. As shown in Table 12, a baseline using SPSA-GC with a large visual prompt (69K parameters) achieves a lower accuracy than more compact models. This is likely because noisy gradient estimates are less stable in higher-dimensional spaces. In contrast, AReS's stable, glass-box optimization allows for the effective tuning of a compact prompt, confirming its effectiveness is due to a superior optimization strategy, not parameter quantity.

Second, the performance gain is critically dependent on the initial priming stage. Our component analysis on EuroSAT (Table 6 in the main paper) shows that local visual reprogramming alone (Local VR) achieves 70.6% accuracy. The full AReS method, which includes priming, elevates this to 85.7%, a substantial +15.1% improvement demonstrating that the knowledge transferred during priming is essential for unlocking the local model's full potential.To further validate this, we created an enhanced version of BlackVIP that incorporates a local pre-training stage. To mimic a local pre-training stage, we trained its prompt-generating Coordinator decoder with a reconstruction loss, as direct supervision is not possible without a ground-truth prompt. As shown in Table 13, this local optimization provides only a marginal improvement to BlackVIP (+0.8%). AReS still significantly outperforms this enhanced baseline by +11.6%, confirming that the performance gain originates from our superior two-stage framework, where priming makes the local model significantly more amenable to reprogramming.

10.4. Performance Scaling with Data Availability

A key question is whether AReS's advantage over a strong, locally-trained baseline persists as more training data becomes available. To investigate this, we conducted an ablation study on the EuroSAT dataset, varying the number of training samples from 4 to 32 shots. The results in Table 14 show that AReS consistently outperforms both the ZOO-based baseline (BlackVIP) and a strong local baseline (Local VR+LP) across all data settings.

While the local baseline's performance improves steadily with more data, it never surpasses AReS. Our method is not only competitive in low-data regimes but also scales more effectively as data increases. For instance, at 32 shots, AReS achieves 91.6% accuracy, maintaining a significant +5.1% lead over the 86.5% from the local baseline. This confirms that the initial priming stage provides a durable advantage that local training alone cannot replicate, equipping the local model with a superior foundation that enables a higher performance ceiling.

10.5. Synergistic Effect of Combining Model Knowledge

In certain configurations, AReS's performance can surpass that of a glass-box reprogramming approach on the service model itself. This outcome stems from a synergistic effect, where our framework effectively combines the distinct knowledge of the powerful service model with the unique inductive biases of the local model. This process of navigating differing predictive behaviors to harness more reliable learning outcomes shares conceptual similarities with recent advances in trustworthy multi-view classification [34]. Unlike methods [41] that use the local model as a simple feature generator, AReS's two-stage design first primes and then leverages the local model's capabilities during reprogramming. This process allows the final model to correctly classify samples that neither the service model nor the standalone local model could handle individually.

This synergistic effect is demonstrated quantitatively in our analysis of the Flowers102 dataset (Table 15), where we compare scenarios with service models of varying strength. When the service model (RN101) has capabilities comparable to

Table 15. Illustrative Breakdown of AReS's Performance Composition on Flowers102. The local model for both scenarios is ViT-B/32 (glass-box Acc: 38.7%). "Kept correct" refers to samples the local model classified correctly that AReS also classifies correctly. "Newly learned" refers to samples the local model misclassified that AReS now classifies correctly.

Initial State	**Service: RN101 (42.3%)**		**Service: ViT-B/16 (69.6%)**	
	% of Dataset	**How AReS Performed**	**% of Dataset**	**How AReS Performed**
Both Service & Local Correct	30.0%	29.5% kept correct	35.0%	34.0% kept correct
Only Local Correct	8.7%	7.0% kept correct	3.7%	0.5% kept correct
Only Service Correct	12.3%	10.0% newly learned	34.6%	19.0% newly learned
Both Incorrect	49.0%	4.8% newly learned	26.7%	0.1% newly learned
AReS Final Accuracy	**51.3%**		**53.6%**	

Table 16. Accuracy (%) comparison on ImageNet in a 16-shot setting with CLIP ViT-B/16 as the service model. AReS significantly outperforms all baselines.

Method	**ImageNet Acc (%)**	**Gain over Zero-shot (%)**
VR (glass-box)	67.4	+0.7
Zero-shot	66.7	-
BAR	64.6	-2.1
VP w/ SPSA-GC	62.3	-4.4
BlackVIP	67.1	+0.4
AReS (Ours)	**80.1**	**+13.4**

the local model, AReS's final accuracy surpasses both, largely by learning to classify an additional 4.8% of samples that were incorrect for both models initially. Conversely, when the service model (ViT-B/16) is significantly stronger, AReS's role shifts to highly effective knowledge transfer, yielding a remarkable +14.9% absolute improvement over reprogramming the local model alone. This analysis confirms that AReS facilitates a potent combination of knowledge, creating synergistic outcomes not possible with other approaches.

10.6. Performance on Large-Scale Benchmarks: ImageNet

To further validate our method's effectiveness on a large-scale, stable benchmark, we conducted experiments on ImageNet [49] in a 16-shot setting, using CLIP ViT-B/16 as the service model. The results, presented in Table 16, demonstrate AReS's significant superiority. While prior ZOO-based methods like BAR and VP w/ SPSA-GC degrade performance compared to the zero-shot baseline, and the SOTA BlackVIP provides only a marginal +0.4% gain, AReS achieves a remarkable 80.1% accuracy. This represents a substantial +13.4% improvement over the zero-shot baseline and a +13.0% gain over the next best competitor, BlackVIP. Notably, AReS even surpasses the glass-box reprogramming baseline, further highlighting the powerful synergistic effect of combining the service model's knowledge with the local model's inductive biases. This is particularly notable as it demonstrates that when a strong local model is available, AReS can effectively employ this advantage to achieve performance superior to both the service model and other local-model-based methods. In contrast, the BlackVIP baseline, despite utilizing the same local encoder architecture, fails to leverage its capabilities, providing only a negligible improvement. These compelling results on a challenging, large-scale dataset confirm that our conceptual shift away from the ZOO-based paradigm is a more robust and effective strategy for closed-box model adaptation.

10.7. Performance Analysis on Challenging VLM Scenarios

While AReS demonstrates considerable average performance improvements with unparalleled efficiency, its efficacy with VLMs like CLIP as a service can exhibit variability on particularly challenging datasets, notably Food101 and Cars, as indicated in Table 2. The inherent difficulty of these datasets for reprogramming is underscored by the performance of even highly capable baselines. For instance, glass-box VR performed directly on the CLIP ViT-B/16 service model achieves 81.6% on Food101, which is below the zero-shot performance of 85.9%. On Cars, glass-box VR obtains 66.2%, only a marginal improvement over the 65.2% zero-shot accuracy. This suggests a ceiling for reprogramming efficacy on these complex domains, even with full model access. Consequently, ZOO-based closed-box methods like BAR and BlackVIP also struggle; on Food101, BAR (84.4%) and BlackVIP (85.9%) offer negligible to no improvement over zero-shot performance. Similarly,

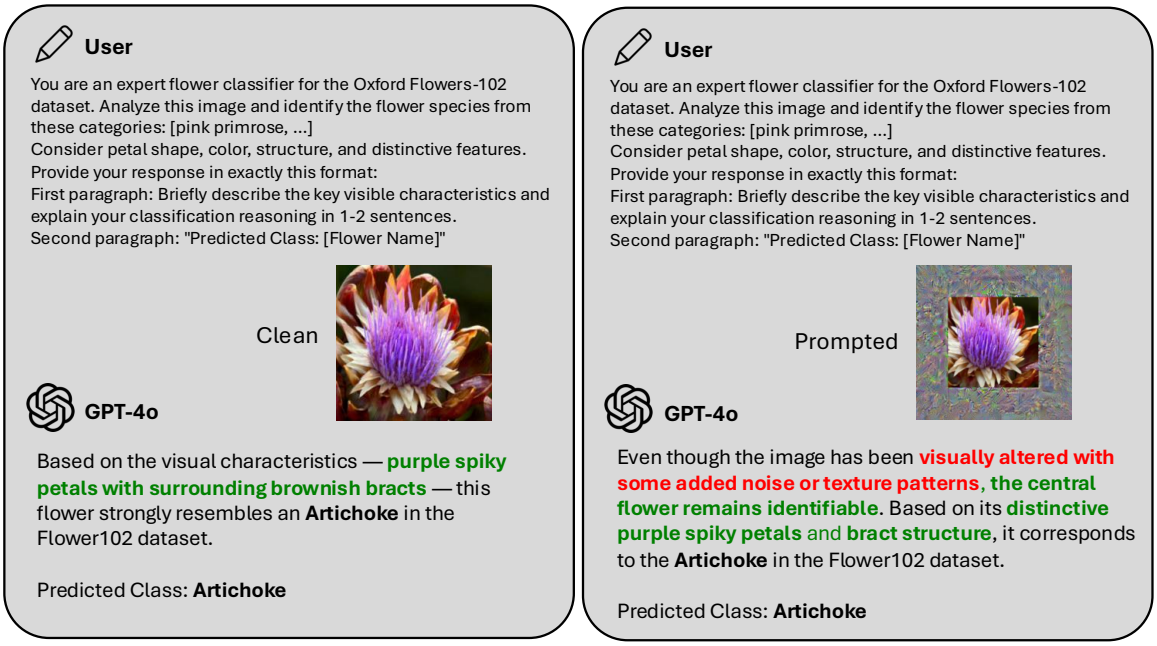

Figure 6. An example of GPT-4o's robustness to the input perturbations used in reprogramming. When presented with a clean image (left) and a prompted image (right), GPT-4o's textual reasoning explicitly acknowledges the visual alteration but correctly identifies the flower's key features in both cases. This demonstrates that the model is often invariant to the input noise that ZOO-based methods rely on, motivating a strategic shift in adaptation methods.

on Cars, BAR (63.0%) underperforms the zero-shot baseline, and BlackVIP (65.4%) provides only a minimal gain. These results highlight that substantial API call volumes and computational efforts by existing closed-box methods do not reliably translate into significant performance gains on such challenging datasets.

In this context, AReS's performance (68.8% on Food101 and 43.2% on Cars in Table 2) is logically influenced by the capabilities of the local ViT-B/16 model. Given that our approach entirely avoids API calls during inference and subsequent reprogramming, its success is intrinsically tied to how well the primed knowledge empowers the local model for the specific downstream task. On these particularly demanding datasets, the primed local model may not fully capture the intricate features that the larger service model leverages for its strong zero-shot performance, leading to a performance gap. However, it is crucial to evaluate AReS's contribution in terms of enhancing the local model's standalone reprogrammability. As shown in Table 10, direct Local VR on the ViT-B/16 achieves only 18.0% on Food101 and a mere 5.7% on Cars. AReS elevates these figures to 68.8% and 43.2% respectively, marking substantial improvements of $+50.8\%$ and $+37.5\%$. This significant enhancement of the local model's utility, achieved with minimal, one-time API interaction and drastically reduced computation, is of immense practical value, particularly in scenarios where continuous and costly reliance on service APIs is untenable. Addressing the remaining performance gap on these specific challenging datasets, possibly through advancements in primed techniques or by employing more capable local architectures, presents an interesting direction for future research.

10.8. Results on MLLMs and Real-world APIs

To validate AReS's practical effectiveness on modern models, we conducted a targeted evaluation using the EuroSAT 16-shot benchmark on three distinct services: the open-source MLLM LLaVA, the proprietary VLM GPT-4o, and the commercial VM API [8]Clarifai. Since MLLMs do not produce output probabilities by default, we instructed the models to act as a classifier and return a confidence score for each class (e.g., "`For the classes [...], provide a confidence score from 0 to 1 for each.`"). Our evaluation reveals a critical limitation of traditional BMR on these services. As illustrated in Figure 6, powerful models like GPT-4o are highly robust to the input perturbations central to ZOO-based reprogramming. The model's textual reasoning shows it can "see past" the visual noise to the underlying image content, rendering the ZOO process that relies on these perturbations ineffective. This finding, supported by quantitative results where

[8]https://www.clarifai.com/

ZOO fails to improve over the zero-shot baseline (Table 4), requires a strategic shift away from perturbation-based adaptation for modern APIs.

In contrast, AReS's two-stage framework is architecturally immune to this issue. By performing a one-time priming step and shifting all subsequent reprogramming to a local model, AReS's success is not dependent on the service model's sensitivity to input noise. This allows it to achieve a remarkable **+27.8%** gain over the zero-shot baseline on GPT-4o, a task where other methods stagnate. Furthermore, on the commercial Clarifai API—a closed-vocabulary setting where zero-shot is not an option—AReS again achieves the highest accuracy (83.2%) at a fraction of the cost ($0.20), making it over **300x cheaper** than BlackVIP ($67.30). These results confirm that AReS's strategy provides a more robust, effective, and economically viable solution for adapting modern closed-box models.

10.9. Computational Efficiency Analysis

To evaluate AReS's local resource trade-offs, we report peak GPU memory and per-image inference latency measured on an NVIDIA RTX 3090 for the EuroSAT benchmark (Table 17). AReS requires comparable GPU memory to existing closed-box baselines such as BlackVIP while achieving +12.4% higher accuracy. Notably, AReS achieves the fastest inference latency (60ms) by running entirely locally after the one-time priming stage, whereas API-based methods incur additional network latency on top of their local computation. This modest local computational cost represents a reasonable trade-off for eliminating API dependency and enabling cost-free, low-latency inference at deployment time.

Table 17. GPU memory and inference latency comparison on EuroSAT with NVIDIA RTX 3090. AReS achieves the highest accuracy and fastest inference by running entirely locally after priming.

Method	GPU Mem. (MB)	Latency (ms)	Acc. (%)
VR (glass-box)	11,937	-	90.9
Zero-shot	0	120	47.9
BAR	1,649	122	77.3
BlackVIP	2,428	170	73.3
AReS (Ours)	**2,781**	**60**	**85.7**

10.10. Robustness to Domain Gap Between Service and Local Models

To evaluate AReS's robustness when there is a substantial domain gap between the service model and the local model, we conduct an experiment using a domain-specific dermatology service model (DermCLIP, pretrained on the Derm1M dataset [72]) paired with a generic local encoder (ViT-B/16 pretrained on natural images) on the HAM10000 skin lesion dataset [59] under the 16-shot setting. This setup represents a challenging scenario where the service model possesses highly specialized domain knowledge (medical dermatology), while the local model has only been exposed to generic natural image features. Recent works have highlighted the importance of domain-specific datasets in medical imaging [62, 63], yet deploying such specialized models often requires costly API access. AReS addresses this by transferring specialized knowledge to a generic local model through one-time priming.

Despite the substantial domain mismatch (medical vs. natural images), AReS outperforms both the zero-shot baseline and BlackVIP (Table 18), confirming that it is not intrinsically tied to domain-matched encoders. The priming mechanism successfully aligns the generic local feature space to the specialized service model even without domain-specific pretraining. The smaller gains compared to domain-aligned settings are consistent with our theoretical analysis: extreme domain gaps increase the priming bound ε (Assumption 2), and performance is ultimately constrained by the local encoder's capacity to represent specialized features (e.g., skin lesion patterns).

Table 18. Domain gap experiment: DermCLIP (pretrained on Derm1M) $\rightarrow$ ViT-B/16 (pretrained on ImageNet) on HAM10000 under the 16-shot setting.

Method	Zero-shot	BlackVIP	**AReS (Ours)**
Acc. (%)	62.5	63.2	**63.6**

10.11. Robustness to Partial API Outputs

Many real-world APIs return only partial outputs, such as top-k predictions rather than the full probability distribution over all classes. To evaluate AReS's robustness under such constraints, we conduct an ablation study restricting the service model's output to: (i) *top-k soft labels* (top-k class names with their confidence scores) and (ii) *top-k hard labels* (only an ordered top-k list without confidence scores). For both settings, we convert the partial output into a usable training target by keeping the revealed top-k information and distributing the remaining probability mass uniformly across the unrevealed classes. For hard labels, we assign a total mass of $c(k) = \frac{k}{k+1}$ to the top-k set and split it across the k classes by rank using a decay weight $w_r \propto \frac{1}{r}$, assigning higher mass to higher-ranked classes.

Results on EuroSAT (10 classes) are shown in Table 19. AReS demonstrates graceful degradation as k decreases, while remaining close to full-output performance for moderate k values (e.g., $k = 5$ achieves 85.0% vs. 85.7% with full output). This constraint equally affects all methods relying on output probabilities (including BAR and BlackVIP), as their optimization objectives similarly depend on full prediction distributions.

Table 19. Robustness to partial API outputs (top-k) on EuroSAT. "Soft" uses top-k class names with confidence scores; "Hard" uses only the ordered top-k list.

Top-k	1	2	5	all (10)
Soft	80.1	82.3	85.0	85.7
Hard	70.7	74.6	78.9	80.6

11. Further Discussion on Contributions and Design Rationale

We further clarifies the core contributions of AReS, its design rationale, and its positioning relative to alternative paradigms.

11.1. On Novelty: Priming for Reprogrammability vs. Traditional Distillation

A key contribution of AReS is the high-level 'prime-then-reprogram' conceptual framework. This framework is flexible, and the priming stage is not constrained to a single implementation. While this work successfully uses an objective function inspired by knowledge distillation (KL divergence), the core idea is to enhance the local model's reprogrammability. Any future technique that can effectively prime a local model's amenability to reprogramming from a closed-box source could be integrated into this framework. This focus on "**priming for reprogrammability**" is what fundamentally distinguishes AReS from traditional distillation (also in Sec. 4):

1. Traditional KD aims to create a final, high-performance student model. This approach often fails in BMR scenarios, particularly when adapting standard Vision Models (VMs). For instance, if the service model is a VM (e.g., pre-trained on ImageNet) and the target is Flowers102, their **label spaces are disjoint**. One cannot directly distill a Flowers102 classifier from an ImageNet classifier.
2. AReS's Priming is **solely a preparatory step**. It is not intended to solve the final task. Instead, it operates in the service model's label space (e.g., ImageNet) to make the local model API-aware when given downstream data (e.g., Flowers102 images).

This novel preparatory focus is what enables the second, local reprogramming stage to be highly effective, even in challenging disjoint-label VM scenarios where *traditional distillation is not applicable*.

We note that knowledge distillation has been explored extensively in both vision and language domains, including step-by-step distillation from large language models [19, 21], multi-chain-of-thought consistent distillation [10], and black-box few-shot knowledge distillation [39]. While these methods have shown success in their respective settings, they fundamentally aim to produce a student model that solves the same task as the teacher. In contrast, AReS's priming stage operates across disjoint label spaces and serves only as a preparatory mechanism for subsequent reprogramming, representing a fundamentally different use of the distillation objective.

11.2. On Practicality: Cost-Free Inference as a Core Design Goal

A core design choice of AReS is to perform a single-pass knowledge transfer, which then **intentionally eliminates all subsequent API dependency** during inference. This is not a shortcoming but the central advantage of our framework, enabling practical deployment in on-device, edge, offline, or cost-sensitive scenarios. The practical motivation for this new paradigm is threefold:

1. **ZOO-based Methods are Less Effective for Modern APIs:** We find that the ZOO-based paradigm is becoming ineffective on modern, robust APIs. Powerful models like GPT-4o are often less sensitive to the input perturbations that ZOO methods rely on . Our experiments (Table 4) confirm this: ZOO-based methods provide little to no improvement over the zero-shot baseline on GPT-4o. In contrast, AReS achieves a **+27.8% gain**, succeeding precisely where the previous paradigm fails. This provides a strong practical reason to shift to a local model primed by the service API.
2. **Synergistic Performance:** The AReS framework can unlock synergistic effects by combining the knowledge of the powerful service model with the inductive biases of the local model. As shown in our analysis (Table 15), this allows AReS to correctly classify samples that neither the service model nor the standalone local model could handle individually. This synergy is so effective that AReS can even **outperform the glass-box VR performance of the service model**—the theoretical performance ceiling for any ZOO-based method.
3. **Enabling Real-World Deployment:** This design unlocks a wide range of practical scenarios where perpetual API access is not feasible or desirable, such as on-device or edge-computing applications, scenarios requiring real-time or offline adaptation, and cost-sensitive applications where a $>$99.99% reduction in API calls is a primary requirement.

Our real-world API experiments (Table 4) confirm this total practical value. On the commercial Clarifai API, AReS achieves superior accuracy (83.2%) for just \$0.20, while BlackVIP costs \$67.30 for lower accuracy (72.1%).

11.3. On Efficacy: AReS vs. Standalone Local Model Reprogramming

To isolate the contribution of our priming stage, we conducted extensive component analyses comparing AReS to a Local VR baseline. This baseline represents the naive approach of simply reprogramming the local model without any priming.

The results conclusively demonstrate that our performance gain is not merely from "just training a small network," but from the synergistic "**prime-then-reprogram**" framework.

1. On EuroSAT (Table 6): Local VR (baseline) achieves 70.6% accuracy. The full AReS framework achieves 85.7%—a +15.1% gain directly attributable to the priming stage.
2. On VLMs (Table 9): AReS achieves a 65.4% average, a +27.2% improvement over the 38.2% from Local VR.
3. On VMs (Table 11): AReS (50.4%) improves upon Local VR (45.3%) by +5.1%.

This empirically proves that our priming stage successfully transfers knowledge from the service model, making the local model significantly more amenable to effective reprogramming.

11.4. On the Theoretical Framework

Our theoretical analysis provides the formal justification for why our two-stage approach is effective. The assumption of ϵ-Faithful Priming (Assumption 2) is not a given; it is the explicit goal of our Priming stage. Our framework is constructive:

1. Principle: We posit that if ϵ-faithful priming can be achieved, the unstable, query-heavy closed-box optimization on the service model $\mathcal{F}_S$ can be provably bounded by an efficient, stable, first-order optimization on the local model $\mathcal{F}_L$ (Theorem 3).
2. Mechanism: The Primeing stage is the practical mechanism we designed to achieve this faithful priming (i.e., a small ϵ). We also empirically validate the robustness of this mechanism in our ablation study (Fig. 3c). The results show that our practical, probability-only (closed-box) method achieves the same strong performance as a variant using logits (translucent-box), confirming that our theoretical analysis is well-grounded and its assumptions do not create a gap with our practical implementation.
3. Result: Our extensive empirical results—especially the +27.8% gain on GPT-4o where ZOO methods fail —demonstrate that this mechanism succeeds in practice.

The theory, therefore, explains why our practical approach of shifting optimization to a local model is a sound and effective strategy for leveraging the service model's capabilities. To the best of our knowledge, this is **the first work to provide a formal theoretical analysis for model reprogramming** in the context of closed-box service models.

11.5. Positioning Within the Broader Visual Prompting Landscape

AReS is situated within a rapidly growing landscape of visual prompting and parameter-efficient fine-tuning (PEFT) methods [25, 36, 61, 65, 68–70]. Furthermore, this push towards highly efficient adaptation and data utilization conceptually parallels recent breakthroughs in broader efficient learning paradigms, such as dataset distillation [81, 82] and efficient generative modeling [83], and LLM-driven active learning [47]. These works have advanced prompt design in diverse directions, including instance-aware prompting, intermediate-representation-based transfer, and systematic studies of efficiency-performance trade-offs in visual recognition. Beyond standard white-box settings, recent efforts have also explored visual prompting under

distribution shift and restricted model access [43, 78, 79], extending the prompting paradigm to scenarios where gradient information is unavailable or the model must adapt continually.

While these methods primarily operate in settings where the model is either fully accessible or adapted at test time, AReS addresses a distinct challenge: efficiently transferring knowledge from a closed-box service model to a local model through one-time priming, followed by local glass-box reprogramming. This positions AReS as complementary to these approaches, offering a practical pathway for leveraging powerful but inaccessible service models.

12. Limitations

Our AReS method, while offering benefits in efficiency and local model enhancement, has certain limitations. In extreme data scarcity scenarios, such as 1/2-shot learning, the one-time priming phase may be constrained (see Fig. 3b). This can challenge the ϵ-faithful primed assumption in our theoretical analysis, potentially affecting the optimality of subsequent local VR and leading to suboptimal performance. Furthermore, AReS's performance is inherently affected by the representational capacity of the chosen local model. If the local model, even after priming, cannot capture the complexities of a particularly challenging downstream task, AReS may not achieve the performance levels of methods that continuously leverage the full computational power and feature richness of the larger service model throughout adaptation (discussed in Sec. 10.7). An interesting future direction is to explore connections between AReS and black-box test-time adaptation [35, 80], where models must adapt to distribution shifts at test time without access to the model's internals. Combining the strengths of one-time priming with online adaptation strategies could further enhance the robustness and flexibility of closed-box model adaptation.