DEFTS: DATA-EFFICIENT, TRAINING-FREE FEW-SHOT LEARNING FOR CLIP IN THE TEXT SUBSPACE

Anonymous authors
Paper under double-blind review

ABSTRACT

Contrastive Language-Image Pre-training (CLIP) achieves impressive zero-shot image classification performance by comparing image embeddings against class-specific text embeddings. While recent few-shot methods leverage labeled training samples to improve upon this zero-shot baseline, they often predominantly rely on image embeddings and underutilize the semantic structure of the text embeddings. In this paper, we introduce DEFTS, a data-efficient, training-free few-shot classification method strictly in the text subspace. We first demonstrate empirically that while raw text embeddings can be suboptimal as direct classifiers, the linear subspace they span remains highly discriminative. DEFTS therefore proposes to reduce the image-text modality gap by projecting estimated image prototypes directly into this subspace. Importantly, by expressing these prototypes as linear combinations of text embeddings, DEFTS reduces a high-dimensional estimation problem to the derivation of just a few coefficients. These coefficients are then computed via a Bayesian posterior mean, using a prior naturally centered on the original text embedding coordinates. Extensive experiments across 12 image classification benchmarks demonstrate that DEFTS significantly improves upon existing training-free methods.

1 INTRODUCTION

Contrastive Language-Image Pre-training (CLIP) (Radford et al., 2021) has gained considerable attention due to its impressive zero-shot capabilities. The model is trained to maximize the similarity between matched image-text pairs while minimizing the similarity across all incorrect pairings. By leveraging over 400 million image-text pairs, CLIP shows remarkable zero-shot image classification performance without requiring any dataset-specific fine-tuning. This is achieved by passing an image through a image encoder and class-specific text prompts (e.g., "a photo of a {class}") through a text encoder, projecting both into a shared semantic space. The image is then assigned to the class whose text embedding yields the highest cosine similarity.

Despite its powerful zero-shot performance, recent work (Wang et al., 2024; Zhang et al., 2022; Guo et al., 2023; Wang et al., 2019; Udandarao et al., 2023; Zanella & Ben Ayed, 2024; Goswami et al., 2026; Zhou et al., 2022; Zhao et al., 2024) has demonstrated that the performances of CLIP could be improved significantly by utilizing only a few labeled training samples, known as few-shot learning. In particular, training-free few-shots methods (Wang et al., 2024; Zhang et al., 2022; Udandarao et al., 2023; Goswami et al., 2026) are becoming increasingly more popular because of their computational efficiency and comparable power compared with training-required methods (Wang et al., 2024).

While significantly improving upon zero-shot baselines, most training-free methods rely heavily on the image embeddings and underutilize the semantic structure of the text embeddings. For example, Wang et al. (2024) fit a Linear Discriminant Analysis (LDA) model directly on the visual features to classify images during inference. Similarly, Zhang et al. (2022) cache the k-shot image embeddings as keys and their one-hot labels as values to compute attention weights for test images. Both approaches only utilize the text embeddings by ensembling with the original zero-shot CLIP predictions.

Therefore, in this paper, we aim to maximize the utility of the text embeddings by constraining the alignment of image and text representations strictly within the text-spanned subspace. Because our approach operates entirely within the the text space, it naturally complements existing image-centric adaptation methods.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Related Work Adapting foundational vision-language models to downstream tasks without gradient-based optimization has become highly desirable due to its computational efficiency. Early approaches focused on retrieval and attention mechanisms: Zhang et al. (2022) build a key-value cache directly from few-shot support features, Udandarao et al. (2023) impvoes CLIP performances by creating a curated support set, Guo et al. (2023) apply a parameter-free cross-modal attention module to sharpen zero-shot logits, and Zhu et al. (2024) align the extracted image and text features through two subspace projections decided by image-image and text-image attention weights.

Statistical classifiers operating directly on the extracted embeddings have also proven remarkably effective. Wang et al. (2019) demonstrate that a properly normalized Nearest Class Mean (NCM) rule acts as a competitive baseline. More recently, Wang et al. (2024) showed that a training-free Linear Discriminant Analysis (LDA) classifier yields exceptionally strong performance when ensembled with zero-shot CLIP. Several studies (Goswami et al., 2026; Zhang et al., 2026) have explored leveraging the text subspace to enhance CLIP's performance in the training-free few-shot setting. However, these methods typically rely on ad-hoc heuristics to align visual prototypes within this subspace. Specifically, TAMP (Goswami et al., 2026) projects few-shot image prototypes onto the text embedding space and linearly mixes the result with the text prototype via a scalar weight λ, which is selected on a held-out validation set. Alternatively, Zhang et al. (2026) split the image embeddings into components within the text embedding space and its orthogonal complement, subsequently reweighting them via a tuned hyperparameter α. Further discussion of related literature is provided in Appendix A.

Contributions Our primary contributions in this paper are summarized as follows:

1. We empirically demonstrate that while original zero-shot text embeddings is suboptimal for direct classification, the linear subspace they span is highly discriminative. Specifically, we show that Nearest Class Mean (NCM) classification, which uses image prototypes estimated strictly within this text-spanned subspace, achieves comparable or superior performance to full-space estimation while being significantly more sample-efficient.
2. Building on these insights, we introduce DEFTS: a **D**ata-**E**fficient, training-free **F**ew-shot learning method in the **T**ext-embedding **S**ubspace. Importantly, we notice that because any point in this space is a linear combination of the text embeddings, prototype estimation reduces to inferring a low-dimensional set of coefficients. DEFTS then determines these coefficients via a closed-form Bayesian formulation, utilizing a prior naturally anchored to the original zero-shot text coordinates.
3. We conduct extensive evaluations across 12 image classification benchmarks and 3 different model backbones. The results demonstrate that DEFTS naturally complements prior work: ensembling DEFTS with existing training-free adaptation methods yields significant and consistent performance gains.

2 METHOD

In this section, we first review CLIP for zero-shot learning. Then we present a series of observations demonstrating that the text embedding subspace is highly discriminative and data efficient. Based on these insights, we provide details of our proposed training-free method DEFTS.

2.1 CLIP FOR ZERO-SHOT CLASSFICATION

In the zero-shot setting, CLIP performs classification by comparing an image embedding against a set of class-specific text embeddings, assigning the image to the class that yields the highest cosine similarity. Specifically, let I denote an input image and $T_1, \ldots, T_K$ denote the synthesized text prompts (e.g., "a photo of a class") for each of the K classes. CLIP extracts the representations using a vision encoder F_v (e.g., a Vision Transformer or ResNet) and a text encoder F_t (typically a Transformer). Assuming the extracted embeddings are L_2-normalized, the predicted class $\hat{y}$ is determined by maximizing their inner product:

$$\hat{y} = \arg \max_{i \in \{1,\ldots,K\}} F_v(I)^\top F_t(T_i). \tag{1}$$

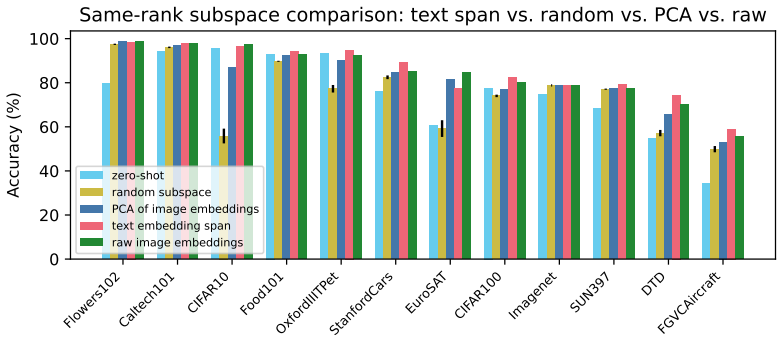

Figure 1: Nearest Class Mean (NCM) accuracy across 12 datasets when projecting image embeddings into the text-spanned subspace. This is compared against zero-shot CLIP, the full un-projected image embedding space, a random subspace and a PCA subspace of the image embeddings both with rank equivalent to the text subspace. Results for the random subspace are averaged across 10 random seeds, with error bars indicating one standard error.

2.2 Text Embedding Subspace

As shown above, the rule for deciding the class in CLIP is straightforward. However, it is less immediately obvious that the prediction results can only depend on the image embedding through its projection onto the subspace spanned by the K class text embeddings. Let $P_{\mathcal{S}}$ denotes the projection matrix into the text embedding subspace, we have

$$F_v(I) = P_{\mathcal{S}} F_v(I) + (I - P_{\mathcal{S}}) F_v(I).$$

Since $(I - P_{\mathcal{S}})F_v(I)$ is orthogonal to any text embeddings,

$$F_v(I)^\top F_t(T_i) = F_v(I)^T P_{\mathcal{S}} F_t(T_i).$$

While most prior work focuses on finding a better alternative to the text embedding $F_t(T_i)$, either by finding better representations (Zhou et al., 2022; Goswami et al., 2026) or by ignoring them entirely (Wang et al., 2024), few studies have investigated whether the linear space $\mathcal{S}$ defined by $P_{\mathcal{S}}$ is actually a suitable subspace for the image classification task.

Given that standard zero-shot classification often yields sub-optimal results, we therefore ask the question whether the text subspace is the main problem. Informally, if $\mathcal{S}$ is an inherently poor subspace, projecting the image embeddings onto it will inevitably discard critical discriminative information. In this case, classification accuracy would be strictly bottlenecked by the projection. Conversely, if $\mathcal{S}$ is highly discriminative and preserves this information, then the raw text embeddings themselves are the true bottleneck limiting performance, not the subspace they span.

In this section, we present a series of observations for investigating the the text embedding subspace. All experiments conducted in this section employ CLIP ViT-L/14 as model backbone, and utilize the hand-crafted prompt templates introduced by CoOp (Zhou et al., 2022) as shown in Appendix E.2. The experiments details for supporting our observations are provided in Appendix E.3.

Observation 1: The text-spanned subspace is highly discriminative and not the bottleneck. In our first empirical observation, we investigate the discriminative power of the linear subspace spanned by the text embeddings (as described in Section 2.3 but without centering first). It is well established that replacing text embeddings with the class means of the image embeddings, effectively applying a Nearest Class Mean (NCM) classifier (Wang et al., 2019), significantly improves classification performance. To investigate the discriminative power of the text subspace, we train an NCM classifier strictly within this subspace, and subsequently evaluate its accuracy.

As shown in Figure 1, the accuracy of this projected NCM classifier is significantly higher than that of standard zero-shot CLIP. This suggests that the text subspace is **not** the bottleneck for classification accuracy. For comparison, we also evaluate the full un-projected image embedding space, projections onto a random subspace and a PCA subspace of the image embeddings both with rank equivalent to the

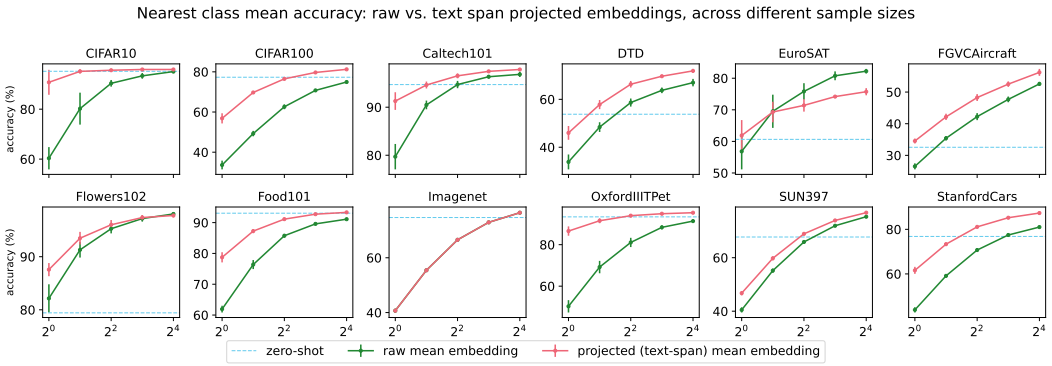

Figure 2: Nearest Class Mean (NCM) accuracy across 12 datasets, comparing image prototypes estimated within the text subspace against estimations in the un-projected embedding space. Accuracies are evaluated across varying k-shots, with all results averaged over 10 random seeds.

text subspace. Surprisingly, the text embedding subspace consistently outperforms the PCA subspace and, in several datasets, such as Food101, surpasses the full, un-projected image embeddings.

Observation 2: Class prototype estimation within the subspace is highly data-efficient. Following Observation 1, we further investigate the accuracy of the NCM classifier when only a small number of training samples per class are available to estimate the image prototypes. We compare the results of estimating these prototypes strictly within the text-spanned subspace against estimating them directly in the full embedding space. As shown in Figure 2, subspace estimation is significantly more data-efficient. It outperforms direct estimation by a large margin when the number of sample sizes are small with only two exceptions: ImageNet and EuroSAT. For ImageNet, the large number of classes ($K = 1000$) causes the text embedding subspace to span the entire feature space, so the projection is equivalent to operating in the full space. Meanwhile, EuroSAT is a well-documented outlier to the CLIP pre-training distribution (Li et al., 2025; Radford et al., 2021), which limits the performance of the text subspace.

Observation 3: The text-spanned subspace is tight; further dimensionality reduction degrades its discriminative power. In this observation, we investigate whether reducing the dimensionality of the text embedding subspace via PCA further degrades its discriminative capacity. We evaluate this by projecting all training samples first and then measuring the NCM accuracy within the reduced subspace. As shown in Figure 3, even a minor removal of variance, e.g. 5%, hurts the subspace's discriminative power. The only exception to this trend is ImageNet which contains 1000 classes with its text embeddings spanning the entire feature space. However, note that the image embeddings in Figure 3 are estimated using all available training samples. When the sample size used for prototype estimation is small, removing a mild amount of variance (e.g., 5%) can provide practical advantages and improve classification accuracy, as detailed in Appendix F.6.

This section investigates the text embedding subspace through a series of observations. We show that this linear subspace is highly discriminative and in-subspace prototype estimation is significantly more sample-efficient than directly estimating image embeddings in the full feature space. While all experiments in this section apply the hand-crafted prompt templates introduced by CoOp (Zhou et al., 2022), we present evidence in Appendix F.1 showing that, surprisingly, the discriminative power of the text-spanned subspace does not seem to depend on the quality of the prompt template.

2.3 DEFTS

Section 2.2 demonstrates empirically that the text embedding space is highly discriminative and data efficient. In this section, we introduce DEFTS, which performs NCM classification using image prototypes estimated strictly within the text-spanned linear subspace by exploiting two additional properties of this subspace: First, because the subspace is low-dimensional, estimating an image prototype requires inferring only a small set of coefficients. Second, the original text embeddings serve as a highly informative prior for these image prototypes. Consequently, we formulate DEFTS

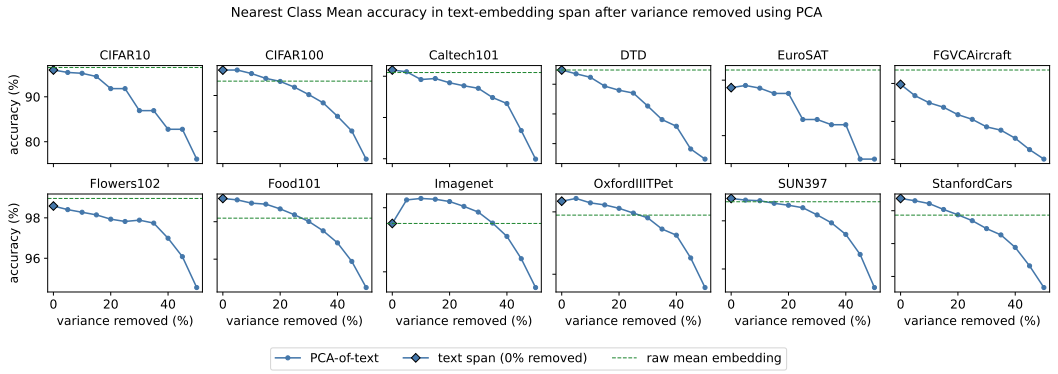

Figure 3: Nearest Class Mean (NCM) classification accuracy across 12 datasets, evaluated within the text embedding subspace after PCA-based variance reduction.

as a Bayesian estimation problem for these coefficients, utilizing a prior that naturally centered in the text embedding coordinates.

Text Embedding Subspace Projection. We first describe how we apply PCA to further reduce the dimensionality of the text embedding subspace. Let $e_1, \ldots, e_K \in \mathbb{R}^d$ denote the CLIP text embeddings for the K classes (generated from a fixed prompt template), and let $\bar{e} = \frac{1}{K}\sum_{c=1}^{K} e_c$ be their mean. We construct the centered scatter matrix:

$$S = \sum_{c=1}^{K} (e_c - \bar{e})(e_c - \bar{e})^\top \in \mathbb{R}^{d\times d}.$$

Then we eigendecompose S and retain the top r eigenvectors to form the basis matrix $Q \in \mathbb{R}^{d\times r}$, where r is chosen to capture at least 95% of the total variance. Specifically, r is the smallest integer satisfying $\sum_{i=1}^{r} \lambda_i / \sum_{i=1}^{K} \lambda_i \geq 0.95$. As demonstrated by Observation 3 in Section 2.2 and the empirical results in Appendix F.6, this 95% threshold provides practical benefits and improves classification accuracy when only a small number of training samples are available, whereas further reduction in variance decreases the discriminative power of the subspace. Note that it is also possible to project the image embeddings onto the original, uncentered text embeddings, as we have done in Section 2.2. In Appendix F.2, we compare these centered and uncentered projection spaces, and show that centering the text embeddings prior to projection yields improved classification accuracy.

In the following, we project all embeddings into this r-dimensional subspace. For each class c, the text embedding coordinate is $e'_c = Q^\top e_c \in \mathbb{R}^r$, and the k support images $x_{c,i}$ yield coordinates $b_{c,i} = Q^\top x_{c,i} \in \mathbb{R}^r$. All subsequent Bayesian estimation operates entirely within this reduced coordinate space.

Bayesian Formulation: Prior and Likelihood Let $\theta_c \in \mathbb{R}^r$ represent the unknown true prototype for class c in the projected subspace. We place a Gaussian prior on θ_c, centered exactly at the projected text embedding e'_c:

$$\theta_c \sim \mathcal{N}(e'_c, \tau^2 I_r).$$

This prior is motivated by utilizing CLIP's zero-shot capabilities. As demonstrated in Section 2.1, CLIP's classification rule depends on the image embedding strictly through its projection onto the text-spanned subspace. Because zero-shot classification is generally effective, the true image prototype naturally align with, and therefore exhibit a small angle relative to its corresponding text embedding in this space. Consequently, the projected text embedding e'_c serves as a highly informative prior for the mean of image embeddings.

The k observed support image coordinates are modeled as independent and identically distributed (i.i.d.) draws from a Gaussian likelihood parameterized by θ_c:

$$b_{c,i} \mid \theta_c \sim \mathcal{N}(\theta_c, \Sigma), \qquad i = 1, \ldots, k.$$

Note that the covariance matrix Σ does not depend on the class index c and is instead shared across all classes. We adopt this shared covariance structure to maximize data efficiency in the few-shot settings, as demonstrated in Section 3.2.3.

Empirical Bayes Hyperparameter Estimation To estimate the hyperparameters of the prior and likelihood, we employ the Empirical Bayes method. Let $\bar{b}_c = \frac{1}{k}\sum_{i=1}^{k} b_{c,i}$ denote the sample mean of the projected images for class c, and let $\delta_c = \bar{b}_c - e'_c$ be its deviation from the text prior. For $k \geq 2$, we first estimate the observation noise by computing the pooled sample variance across all K classes:

$$\hat{\sigma}^2 = \frac{1}{rK(k-1)} \sum_{c=1}^{K} \sum_{i=1}^{k} \|b_{c,i} - \bar{b}_c\|^2. \tag{2}$$

Then we estimate τ^2 as:

$$\hat{\tau}^2 = \frac{1}{r}\overline{\|\delta\|^2} - \frac{\hat{\sigma}^2}{k}, \tag{3}$$

where $\frac{1}{r}\overline{\|\delta\|^2} = \frac{1}{rK}\sum_{c=1}^{K}\left(\|\delta_c\|^2\right)$ is the averaged deviation of the image mean from the text prior to approximate $\frac{1}{r}\mathbb{E}\left[\|\bar{b}_c - e_c\|^2\right]$. Appendix B.1 shows the derivation for the estimator.

In the one-shot setting ($k = 1$), the within-class variance $\hat{\sigma}^2$ cannot be computed from the data. Consequently, σ^2 and τ^2 become unidentifiable, as only their sum, $\frac{1}{r}\overline{\|\delta\|^2} = \tau^2 + \sigma^2$, is observed. To resolve this problem, we partition the total observed spread using empirical results when $k > 1$ as shown in Appendix F.3. These results indicate that $\hat{\sigma}^2$ often accounts for roughly 10% of the averaged empirical deviation $\frac{1}{r}\overline{\|\delta\|^2}$. Thus, we set $\hat{\sigma}^2 = 0.1\frac{1}{r}\overline{\|\delta\|^2}$ and $\hat{\tau}^2 = 0.9\frac{1}{r}\overline{\|\delta\|^2}$ throughout the whole paper.

For the likelihood covariance, we apply a ridge-type shrinkage estimator analogous to the formulation in Wang et al. (2024), adapted here for the r-dimensional subspace. Let $\hat{\Sigma} = \frac{1}{K(k-1)}\sum_{c=1}^{K}\sum_{i=1}^{k}(b_{c,i} - \bar{b}_c)(b_{c,i} - \bar{b}_c)^\top$ be the pooled empirical sample covariance. The regularized precision matrix is defined globally for all classes as:

$$\Sigma^{-1} = \beta(K(k-1) + 2r + 1)\left[K(k-1)\hat{\Sigma} + \mathrm{tr}(\hat{\Sigma})I_r\right]^{-1}, \qquad k \geq 2, \tag{4}$$

where $\beta \in (0, 1]$ is a covariance inflation parameter. Because empirical sample covariances derived from extremely small support sets (e.g. $k = 2$) could be highly unstable as shown in Appendix F.4, the implicit factor $\frac{1}{\beta}$ explicitly inflates the estimated variance to prevent the Bayesian update from becoming overconfident. We investigate the impact of this hyperparameter in detail in Appendix F.5. Empirically, we observe that small β (e.g., 0.1 or 0.2) yields better performance than $\beta = 1$ especially when the number of shots are small, such as when $k = 2$. Thus, throughout the paper, we will fix $\beta = 0.1$. For $k = 1$, we default to $\Sigma^{-1} = \beta\hat{\sigma}^{-2}I_r$. The full derivation for obtaining the estimation is provided in Appendix B.2.

Closed-Form Posterior Inference Denote $\Omega_0 = \hat{\tau}^{-2}I_r$. Given the Gaussian prior and likelihood, the posterior distribution is also Gaussian, and its posterior mean is (Bishop & Nasrabadi, 2006):

$$\hat{\theta}_c = (\Omega_0 + k\Sigma^{-1})^{-1}(\Omega_0 e'_c + k\Sigma^{-1}\bar{b}_c). \tag{5}$$

The complete procedure for performing DEFTS is summarized in Algorithm 1 in Appendix C. The results from DEFTS can be used directly for cosine-similarity classification, or ensembled with existing methods as described in Appendix G.1.

3 Experiments

3.1 DEFTS Complements Existing Training-Free Methods

In this section, we conduct extensive experiments to evaluate the performance of DEFTS, and existing training-free methods after complemented by DEFTS through ensembling.

Datasets. Following prior relevant work (Wang et al., 2024; Radford et al., 2021; Zhou et al., 2022), we evaluate DEFTS on 12 standard few-shot classification datasets. The datasets span generic object recognition (CIFAR-10/100 (Krizhevsky et al., 2009), Caltech101 (Fei-Fei et al., 2004), ImageNet (Deng et al., 2009)), fine-grained recognition (StanfordCars (Krause et al., 2013), FGVCAircraft (Maji et al., 2013), Flowers102 (Nilsback & Zisserman, 2008), OxfordIIITPet (Parkhi et al., 2012), Food101 (Bossard et al., 2014)), texture recognition (DTD (Cimpoi et al., 2014)), scene recognition (SUN397 (Xiao et al., 2010)), and remote-sensing imagery (EuroSAT (Helber et al., 2019)). In Appendix E.1, we provide more information about these datasets.

Table 1: Classification accuracy (%) and performance improvements across datasets. The values in parentheses denote the absolute performance change after ensembling with DEFTS and the 90% confidence interval. Positive gains are highlighted in green, and negative changes in red.

Dataset	ZS	SimpleShot	Linear Probe	Tip-Adapter	APE	TIP-X	TAMP+ZS	LDA+ZS	DEFTS
C10	94.89 (+0.21 ±0.17)	60.57 (+30.13 ±2.99)	70.54 (+23.49 ±4.18)	95.01 (+0.12 ±0.14)	95.05 (+0.10 ±0.14)	95.03 (+0.11 ±0.13)	94.81 (+0.26 ±0.14)	94.90 (+0.21 ±0.15)	95.14
C100	77.27 (+1.27 ±0.18)	33.04 (+29.25 ±1.04)	38.88 (+26.85 ±1.05)	77.41 (+1.17 ±0.16)	77.85 (+0.79 ±0.11)	77.58 (+0.99 ±0.15)	77.67 (+0.60 ±0.12)	77.29 (+1.27 ±0.19)	77.28
Cal101	94.62 (+0.86 ±0.27)	79.69 (+11.04 ±1.14)	85.78 (+7.03 ±0.84)	94.88 (+0.62 ±0.24)	95.30 (+0.24 ±0.27)	94.99 (+0.51 ±0.23)	95.37 (-0.01 ±0.19)	94.65 (+0.83 ±0.27)	94.73
DTD	53.03 (+6.50 ±0.63)	34.44 (+16.09 ±0.92)	38.20 (+14.78 ±1.06)	53.45 (+6.17 ±0.52)	54.08 (+5.73 ±0.58)	53.99 (+5.60 ±0.58)	56.69 (+2.73 ±0.49)	53.04 (+6.48 ±0.62)	60.31
Euro	61.17 (+2.13 ±0.47)	60.79 (+11.39 ±1.77)	60.83 (+11.91 ±2.01)	62.40 (+1.35 ±0.43)	64.31 (+0.19 ±0.44)	63.03 (+0.94 ±0.43)	60.53 (+2.25 ±0.43)	61.31 (+2.06 ±0.48)	64.62
Air.	32.18 (+7.16 ±0.60)	25.63 (+9.27 ±0.68)	26.85 (+8.92 ±0.62)	33.28 (+6.17 ±0.60)	34.85 (+4.79 ±0.83)	33.71 (+5.80 ±0.63)	37.09 (+2.36 ±0.37)	32.18 (+7.16 ±0.59)	40.45
Flow.	78.40 (+9.38 ±0.36)	82.19 (+8.85 ±0.89)	84.71 (+7.20 ±0.55)	79.11 (+8.77 ±0.33)	81.01 (+7.26 ±0.37)	79.53 (+8.45 ±0.33)	84.87 (+3.50 ±0.27)	78.52 (+9.27 ±0.36)	91.19
Food	93.11 (+0.09 ±0.10)	59.40 (+23.15 ±0.47)	64.31 (+21.04 ±0.43)	93.12 (+0.05 ±0.10)	93.18 (-0.04 ±0.11)	93.14 (+0.02 ±0.10)	93.19 (-0.18 ±0.08)	93.11 (+0.07 ±0.10)	92.01
INet	74.35 (+1.54 ±0.13)	40.94 (+16.19 ±0.29)	44.25 (+14.78 ±0.16)	74.56 (+1.27 ±0.13)	75.11 (+0.62 ±0.13)	74.68 (+1.12 ±0.12)	74.43 (-0.70 ±0.09)	74.41 (+1.46 ±0.13)	70.92
Pet	92.82 (+0.58 ±0.30)	49.77 (+31.63 ±1.10)	57.06 (+29.04 ±1.25)	92.97 (+0.44 ±0.27)	93.14 (+0.28 ±0.31)	93.01 (+0.40 ±0.28)	92.97 (+0.28 ±0.26)	92.86 (+0.54 ±0.30)	92.85
SUN	67.35 (+3.64 ±0.28)	40.37 (+14.28 ±0.23)	44.74 (+12.95 ±0.28)	67.64 (+3.35 ±0.31)	68.23 (+2.68 ±0.30)	67.90 (+3.09 ±0.31)	69.67 (+0.13 ±0.16)	67.38 (+3.60 ±0.27)	67.86
Cars	76.61 (+3.05 ±0.43)	43.79 (+19.20 ±0.66)	47.07 (+19.19 ±0.40)	76.80 (+2.85 ±0.36)	77.37 (+2.26 ±0.39)	76.92 (+2.76 ±0.37)	77.29 (+1.02 ±0.30)	76.65 (+3.01 ±0.42)	78.10
AVG	**74.65** (+3.03 ±0.46)	**50.88** (+18.37 ±1.25)	**55.27** (+16.43 ±1.15)	**75.05** (+2.69 ±0.43)	**75.79** (+2.07 ±0.38)	**75.29** (+2.48 ±0.41)	**76.21** (+1.02 ±0.21)	**74.69** (+3.00 ±0.45)	**77.12**

Existing methods. We consider completing the following training-free few-shot classification methods using DEFTS: **Linear Probe** (Radford et al., 2021): a linear classier on top of image embeddings; **SimpleShot** (Wang et al., 2019), a nearest class mean classifier; **Tip-Adapter** (Zhang et al., 2022), which caches the k-shot features as keys and one-hot labels as values to compute a weighted cache vote added to the zero-shot logits; **TIP-X** (Udandarao et al., 2023), which extends Tip-Adapter with a KL-divergence affinity term between the support and test images' text-class distributions; **APE** (Zhu et al., 2023), which refines Tip-Adapter by selecting class-discriminative channels from text embeddings and reweighting cache votes; **LDA** (Wang et al., 2024), a Gaussian discriminant classifier utilizing a single empirical Bayes shrunk covariance pooled across all classes; and **TAMP** (Goswami et al., 2026), which mixes image and text prototypes in an un-truncated text embeddings subspace. As the standalone LDA classifier typically yields sub-optimal performance, we ensemble it with zero-shot CLIP, as suggested by Wang et al. (2024). Similarly, we evaluate the performances of TAMP after ensembling with both zero-shot CLIP.

Implementation Details. All embeddings are extracted using a frozen CLIP ViT-L/14 (Radford et al., 2021) and prompt templates from CoOp (Zhou et al., 2022) as detailed in Appendix E.2. We evaluate $k \in \{1, 2, 4, 8, 16\}$ support images per class, with results averaged over 10 random seeds. Ensembling methods require choosing weights for each method, e.g. the weight between LDA and zero-shot (ZS). In this section, all methods strictly adhere to the k-shot budget without utilizing external held-out validation. Thus, when $k = 1$, we apply fixed weights for each method. When $k \geq 2$, we search optimal hyperparameters for each method using leave-one-out cross-validation. The details for choosing and tuning these hyperparameters is reported in Appendix G.

Results In Table 1, we present the classification results for the one-shot learning setting ($k = 1$). The results demonstrate that ensembling with DEFTS improves the average accuracy of all evaluated methods across the 12 datasets. For weaker baselines, such as SimpleShot and Linear Probe, the

Table 2: Average classification accuracy (%) and performance improvements across 12 datasets for different numbers of shots k. The values in parentheses denote the absolute performance change after ensembling with DEFTS and the 90% confidence interval. Positive gains are highlighted in green. The best baseline accuracy for each k-shot setting (before adding DEFTS) is highlighted in **bold**.

k-shot	ZS+DEFTS	SimpleShot	Linear Probe	Tip-Adapter	APE	TIP-X	TAMP+ZS	LDA+ZS	DEFTS alone
$k=1$	74.65 (+3.03 ±0.46)	50.88 (+18.37 ±1.25)	55.27 (+16.43 ±1.15)	75.05 (+2.69 ±0.43)	75.79 (+2.07 ±0.38)	75.29 (+2.48 ±0.41)	76.21 (+1.02 ±0.21)	74.69 (+3.00 ±0.45)	**77.12**
$k=2$	74.65 (+2.36 ±0.69)	64.46 (+15.30 ±1.30)	65.74 (+14.07 ±1.07)	76.57 (+0.76 ±0.31)	77.34 (+0.41 ±0.25)	75.19 (+1.27 ±0.18)	76.65 (+0.94 ±0.32)	76.37 (+1.12 ±0.37)	**79.50**
$k=4$	74.65 (+6.33 ±0.91)	73.83 (+8.48 ±0.73)	75.84 (+6.35 ±0.66)	79.05 (+2.16 ±0.45)	79.78 (+1.64 ±0.36)	76.93 (+0.52 ±0.13)	78.80 (+2.20 ±0.46)	**82.12** (+0.68 ±0.26)	81.66
$k=8$	74.65 (+8.90 ±1.10)	78.80 (+5.54 ±0.47)	81.44 (+2.61 ±0.36)	81.37 (+2.79 ±0.35)	82.06 (+2.29 ±0.38)	78.84 (+0.03 ±0.06)	80.00 (+3.30 ±0.54)	**85.24** (+0.49 ±0.13)	83.51
$k=16$	74.65 (+10.04 ±1.20)	81.56 (+4.00 ±0.37)	85.12 (+0.37 ±0.27)	82.85 (+2.69 ±0.33)	83.84 (+2.11 ±0.35)	80.58 (+0.43 ±0.20)	80.89 (+3.78 ±0.59)	**86.70** (+0.70 ±0.16)	84.59

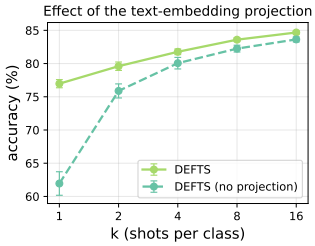

Figure 4: DEFTS' accuracy with and without projecting into the text subspaces.

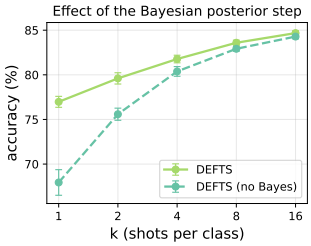

Figure 5: DEFTS' accuracy with and without applying the Bayesian estimation.

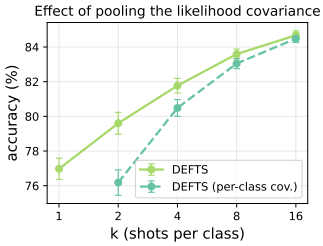

Figure 6: DEFTS' accuracy with and without applying the pooled covariance estimation .

performance gains yielded by DEFTS are substantial. Meanwhile, for strong baselines like APE, we still observe statistically significant improvements. We also notice that the magnitude of these improvements varies across datasets. For example, on the Flowers102 dataset, DEFTS provides strong improvements ($> 3\%$) for all considered methods. In contrast, on the Food101 dataset, DEFTS does not yield further improvements for the already strong baselines. Notably, the standalone DEFTS framework achieves the highest overall average accuracy in the one-shot case. In Appendix F.7, we provide detailed results analogous to Table 1 for varying numbers of shots k.

Table 2 further evaluates the performance of each method across varying numbers of shots k. The results show that DEFTS consistently complements existing baselines. In extremely low-data regimes ($k \in \{1, 2\}$), the standalone DEFTS framework achieves the highest overall accuracy while significantly improving the performance of all other methods upon ensembling. For larger support sets ($k \in \{4, 8, 16\}$), LDA+ZS emerges as the strongest initial baseline; crucially, ensembling it with DEFTS continues to yield statistically significant improvements. Finally, in Appendix F.8, we replicate these simulations using alternative model backbones, demonstrating that the performance gains provided by DEFTS are robust across different models.

3.2 Ablation Study

3.2.1 Estimation in subspace

DEFTS proposes projecting features into a low-rank, variance-reduced text embedding subspace prior to performing Bayesian estimation for the class-conditional image means. We first investigate the impact of this projection by comparing it against an ablation where Bayesian estimation is applied directly in the original, unprojected embedding space. The results Figure 4 demonstrate that projecting into the subspace yields a substantial advantage especially in the low-data regimes ($k \in \{1, 2\}$). As the number of shots k increases, the performance of DEFTS with and without the subspace projection converges.

Table 3: Comparison of average classification accuracy (%) between TAMP and DEFTS across different numbers of shots k. The values in parentheses denote the 90% confidence interval. Positive differences (DEFTS − TAMP) are highlighted in green.

Method	$k = 1$	$k = 2$	$k = 4$	$k = 8$	$k = 16$
TAMP alone	74.47 (±2.65)	76.52 (±2.72)	78.81 (±2.44)	79.96 (±2.33)	80.89 (±2.21)
DEFTS alone	77.12 (±2.49)	79.50 (±2.34)	81.66 (±2.16)	83.51 (±1.95)	84.59 (±1.83)
DEFTS − TAMP	+2.66 (±0.35)	+2.97 (±0.56)	+2.85 (±0.51)	+3.55 (±0.61)	+3.70 (±0.61)

3.2.2 Bayesian formulations

We further investigate the contribution of the Bayesian estimation component within the DEFTS framework. In Figure 5, we compare the proposed Bayesian approach against a simple average of the image embeddings. The results demonstrate that Bayesian estimation yields substantial performance gains, although these benefits decreases as the number of shots increases.

3.2.3 Pooled covariance estimation

As discussed in Section 2.3, DEFTS relies on a pooled sample covariance matrix (Equation 4) to estimate the distribution of the image coordinates. To justify this design choice, we conduct an ablation study comparing our pooled covariance strategy against independent per-class covariance estimation. The results are shown in Figure 6. Note that in the one-shot setting ($k = 1$), computing a per-class covariance is mathematically intractable. When $k \in \{2, 4\}$, pooling the variance across all classes yields superior performance. Conversely, as the support set size increases to $k = 16$, per-class covariance estimation outperforms the pooled approach.

3.2.4 Comparison with closest baseline

In previous ablation studies, we isolate each component of DEFTS and investigates their contributions for DEFTS. In this section, we compare DEFTS directly with TAMP, serving as the closest baseline to DEFTS. TAMP projects image embeddings into uncentered text embedding spaces and estimates the final representations via a weighted average of the mean image embeddings and text embeddings. This averaging weight is then tuned via leave-one-out cross-validation in this paper, as described in Appendix G. The detailed theoretical comparison between DEFTS and TAMP can be seen in Appendix D. As shown in Table 3, the differences between DEFTS and TAMP matters practically as well: DEFTS significantly outperforms TAMP across all evaluated k-shot settings.

4 Conclusion and Limitations

While existing training-free methods effectively enhance the performance of zero-shot CLIP, they often underutilize the semantic meaning from text embeddings. Motivated by this, we introduce DEFTS: a data-efficient, training-free few-shot learning method strictly in the text subspace, and show that DEFTS complements current image-centric methods. We first demonstrate empirically that the text embedding space is highly discriminative and data-efficient. DEFTS therefore performs Nearest Class Mean (NCM) estimation directly within this subspace. By expressing image embeddings as linear combinations of text embeddings, DEFTS reduces a high-dimensional estimation problem down to just a few coefficients. These coefficients are computed as a Bayesian posterior mean, utilizing a prior naturally centered on the text embedding coordinates. Extensive experiments demonstrate the effectiveness of DEFTS: by ensembling with other methods, DEFTS can often improve their accuracy over different datasets.

Limitations While DEFTS significantly complements existing baselines, the magnitude of these improvements varies across datasets, with some datasets showing only marginal gains. Additionally, the relative performance boost provided by DEFTS decreases as the number of shots k increases. This is likely because the baseline methods themselves become substantially stronger with larger support sets. However, even in these higher-shot scenarios, DEFTS demonstrates that it can significantly complement strong baselines.

REFERENCES

Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *European conference on computer vision*, pp. 446–461. Springer, 2014.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3606–3613, 2014.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Bradley Efron and Carl Morris. Data analysis using stein's estimator and its generalizations. *Journal of the American Statistical Association*, 70(350):311–319, 1975.

Andre Esteva, Katherine Chou, Serena Yeung, Nikhil Naik, Ali Madani, Ali Mottaghi, Yun Liu, Eric Topol, Jeff Dean, and Richard Socher. Deep learning-enabled medical computer vision. *NPJ digital medicine*, 4(1):5, 2021.

Kawin Ethayarajh. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 55–65, 2019.

Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *2004 conference on computer vision and pattern recognition workshop*, pp. 178–178. IEEE, 2004.

Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot visual imitation learning via meta-learning. In *Conference on robot learning*, pp. 357–368. PMLR, 2017.

Dipam Goswami, Simone Magistri, Gido M van de Ven, Bartłomiej Twardowski, Andrew D Bagdanov, Tinne Tuytelaars, and Joost van de Weijer. Cross-modal prototype alignment and mixing for training-free few-shot classification. *arXiv preprint arXiv:2603.24528*, 2026.

Ziyu Guo, Renrui Zhang, Longtian Qiu, Xianzheng Ma, Xupeng Miao, Xuming He, and Bin Cui. Calip: Zero-shot enhancement of clip with parameter-free attention. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 746–754, 2023.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

William James, Charles Stein, et al. Estimation with quadratic loss. In *Proceedings of the fourth Berkeley symposium on mathematical statistics and probability*, volume 1, pp. 361–379. University of California Press, 1961.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pp. 554–561, 2013.

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

Olivier Ledoit and Michael Wolf. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of multivariate analysis*, 88(2):365–411, 2004.

Chaohua Li, Enhao Zhang, Chuanxing Geng, and Songcan Chen. Recent advances in out-of-distribution detection with clip-like models: A survey. *arXiv preprint arXiv:2505.02448*, 2025.

Victor Weixin Liang, Yuhui Zhang, Yongchan Kwon, Serena Yeung, and James Y Zou. Mind the gap: Understanding the modality gap in multi-modal contrastive representation learning. *Advances in neural information processing systems*, 35:17612–17625, 2022.

Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *2019 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 2532–2541. IEEE, 2019.

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

Jiaqi Mu, Suma Bhat, and Pramod Viswanath. All-but-the-top: Simple and effective postprocessing for word representations. *arXiv preprint arXiv:1702.01417*, 2017.

Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

Maria-Elena Nilsback and Andrew Zisserman. 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.

Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pp. 3498–3505. IEEE, 2012.

VS Parupudi. Anisotropy decides cosine vs. rank metrics for text embeddings. *arXiv preprint arXiv:2606.29571*, 2026.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on robot learning*, pp. 894–906. pmlr, 2022.

Harald Steck, Chaitanya Ekanadham, and Nathan Kallus. Is cosine-similarity of embeddings really about similarity? In *Companion Proceedings of the ACM Web Conference 2024*, pp. 887–890, 2024.

Vishaal Udandarao, Ankush Gupta, and Samuel Albanie. Sus-x: Training-free name-only transfer of vision-language models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2725–2736. IEEE, 2023.

Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *2018 IEEE/CVF conference on computer vision and pattern recognition*, pp. 8769–8778. IEEE, 2018.

Yan Wang, Wei-Lun Chao, Kilian Q Weinberger, and Laurens Van Der Maaten. Simpleshot: Revisiting nearest-neighbor classification for few-shot learning. *arXiv preprint arXiv:1911.04623*, 2019.

Zhengbo Wang, Jian Liang, Lijun Sheng, Ran He, Zilei Wang, and Tieniu Tan. A hard-to-beat baseline for training-free clip-based adaptation. In *International Conference on Learning Representations*, volume 2024, pp. 28841–28859, 2024.

Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. 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.

Kisung You. Semantics at an angle: When cosine similarity works until it doesn't. *arXiv preprint arXiv:2504.16318*, 2025.

Maxime Zanella and Ismail Ben Ayed. Low-rank few-shot adaptation of vision-language models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1593–1603, 2024.

Renrui Zhang, Wei Zhang, Rongyao Fang, Peng Gao, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. Tip-adapter: Training-free adaption of clip for few-shot classification. In *European conference on computer vision*, pp. 493–510. Springer, 2022.

Yi Zhang, Yu Mao, Hui Chen, Jinyu Gao, Ziqiang Huang, and Yaojin Lin. Multi-bias logit fusion in a pseudo-text space for clip few-shot learning. *Knowledge-Based Systems*, pp. 116834, 2026.

Shuai Zhao, Xiaohan Wang, Linchao Zhu, and Yi Yang. Test-time adaptation with clip reward for zero-shot generalization in vision-language models. In *International Conference on Learning Representations*, volume 2024, pp. 3597–3613, 2024.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International journal of computer vision*, 130(9):2337–2348, 2022.

Xiangyang Zhu, Renrui Zhang, Bowei He, Aojun Zhou, Dong Wang, Bin Zhao, and Peng Gao. Not all features matter: Enhancing few-shot clip with adaptive prior refinement. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2605–2615. IEEE, 2023.

Xingyu Zhu, Beier Zhu, Yi Tan, Shuo Wang, Yanbin Hao, and Hanwang Zhang. Selective vision-language subspace projection for few-shot clip. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 3848–3857, 2024.

A More Related Work

Study of the Shared Embedding Space. Standard zero-shot CLIP classification relies on cosine similarity, assuming implicitly that image and text embeddings occupy a unified semantic space. However, Liang et al. (2022) demonstrate a measurable modality gap by showing that image and text embeddings occupy distinctly separated regions on the embedding sphere. This phenomenon mirrors the anisotropy documented in contextual language models (Ethayarajh, 2019; Mu et al., 2017), where representations concentrate their variance into narrow cones. Recent surveys (You, 2025) and empirical studies (Parupudi, 2026; Steck et al., 2024) reveal that this geometric distortion causes the performances decrease of cosine similarity.

Bayesian Estimation with Small Sample Size. Estimating a highly parameterized class mean from only a handful of samples is notoriously unstable. Classical statistical theory dictates that the naive per-group sample mean is suboptimal in this regime; instead, shrinking every group's sample mean toward a common point achieves uniformly lower total squared error (James et al., 1961). Efron & Morris (1975) operationalized this shrinkage via empirical Bayes, where the strength of the shrinkage is estimated directly from the data's variance rather than heuristically tuned.

Few-shot Learning. Few shot learning is highly desirable in practice where the available training sample size is often limited. First, real-world data frequently exhibits long-tailed distributions, where a small number of common classes dominate the dataset while the vast majority of categories has only a few training samples. For example, in the diagnosis of rare medical conditions, the detection of novel manufacturing defects, or the classification of endangered species, a class may only be represented by only one or two examples (Liu et al., 2019; Van Horn et al., 2018). Second, even when abundant data exists in principle, the cost of expert annotation imposes a practical challenge on the volume of data that can be labeled for deployment (Esteva et al., 2021). Third, rapid on-the-fly personalization, such as teaching a robotic assistant to recognize a user's personal belongings from a single demonstration, demands models that generalize instantly from extreme few-shot interactions (Finn et al., 2017). This capacity for immediate vision-language grounding is becoming increasingly central to the deployment of real-world robotic manipulation systems (Shridhar et al., 2022).

B HYPERPARAMETER ESTIMATION

B.1 DERIVATION FOR THE VARIANCE OF THE PRIOR

To estimate the hyperparameters without relying on held-out validation data, we employ the Empirical Bayes method. Let $\bar{b}_c = \frac{1}{k}\sum_{i=1}^{k} b_{c,i}$ denote the sample mean of the projected images for class c, and let $\delta_c = \bar{b}_c - e_c$ be its deviation from the text prior. For $k \geq 2$, we first isolate the observation noise by computing the pooled within-class sample variance across all K classes:

$$\hat{\sigma}^2 = \frac{1}{rK(k-1)} \sum_{c=1}^{K} \sum_{i=1}^{k} \|b_{c,i} - \bar{b}_c\|^2.$$

In order to estimate τ^2, we note that the true prior covariance is defined by the unobservable true class mean vector $\mu_c \in \mathbb{R}^r$:

$$\text{Cov}(\mu_c - e'_c) = \tau^2 I_r.$$

Because we can only measure this distance using the empirical sample mean $\bar{b}_c$, we must account for sampling noise. Assume that one sample equals the true mean plus a random sampling error vector ϵ. Then we have

$$\bar{b}_c = \mu_c + \frac{1}{k}\sum_{i=1}^{k} \epsilon_i.$$

Since the true semantic shift and the sampling error are independent, their covariance matrices add linearly:

$$\text{Cov}(\bar{b}_c - e'_c) = \text{Cov}\left(\mu_c + \frac{1}{k}\sum_{i=1}^{k}\epsilon_i - e'_c\right) = \text{Cov}(\mu_c - e'_c) + \text{Cov}\left(\frac{1}{k}\sum_{i=1}^{k}\epsilon_i\right) = \left(\tau^2 + \frac{\sigma^2}{k}\right) I_r.$$

Assuming that $\bar{b}_c$ is an unbiased estimation of e'_c, then taking the expected squared L_2 norm equals the trace of the covariance matrix. Dividing the expected squared L_2 norm by the dimensionality r, we obtain the per-dimension scalar relationship:

$$\frac{1}{r}\mathbb{E}\left[\|\bar{b}_c - e'_c\|^2\right] = \tau^2 + \frac{\sigma^2}{k}.$$

Rearranging this equation isolates the true prior variance:

$$\tau^2 = \frac{1}{r}\mathbb{E}\left[\|\bar{b}_c - e'_c\|^2\right] - \frac{\sigma^2}{k}.$$

Using the observed average squared spread

$$\frac{1}{r}\overline{\|\delta\|^2} = \frac{1}{rK}\sum_{c=1}^{K}\left(\left\|\frac{1}{k}\sum_{i=1}^{k} b_{c,i} - e'_c\right\|^2\right)$$

to approximate $\frac{1}{r}\mathbb{E}\left[\|\bar{b}_c - e'_c\|^2\right]$, we derive an estimator of τ^2 by subtracting the expected sampling noise:

$$\hat{\tau}^2 = \frac{1}{r}\overline{\|\delta\|^2} - \frac{\hat{\sigma}^2}{k}$$

where the overline denotes the average across all K classes.

Remark for the prior assumption. The derivation above assumes that the text embedding e'_c serves as an unbiased prior for the true image class mean μ_c (i.e., $\mathbb{E}[\mu_c - e'_c] = 0$). While recent literature has identified a systematic "modality gap" between image and text embeddings in contrastive models (Liang et al., 2022), explicitly modeling this shift requires estimating a high-dimensional bias vector. In few-shot regimes (e.g., $k \in \{1, 2, 4\}$), such estimation could be prone to overfitting. We therefore deliberately enforce this zero-bias assumption as a regularization constraint.

B.2 Derivation for the shrunk covariance estimation

In this section, we show the derivation of the Bayesian shrinkage estimator for the covariance matrix.

Let $\hat{\Sigma}$ denote the pooled empirical sample covariance computed from K classes and k support shots. The total degrees of freedom for this empirical estimate is

$$n = K(k-1).$$

We define the empirical scatter matrix S as the unnormalized sum of squared errors: $S = n\hat{\Sigma}$. Under the assumption that the image embeddings are drawn from a Multivariate Normal distribution $\mathcal{N}(\mu_c, \Sigma)$, this scatter matrix follows a Wishart distribution, $S \sim \mathcal{W}_r(\Sigma, n)$ (Murphy, 2012). The data likelihood for the covariance matrix is therefore proportional to:

$$p(S \mid \Sigma) \propto |\Sigma|^{-n/2} \exp\left(-\frac{1}{2}\operatorname{tr}(\Sigma^{-1}S)\right).$$

To regularize $\hat{\Sigma}$ in the few-shot cases, we place an Inverse-Wishart prior on the true covariance matrix Σ:

$$\Sigma \sim \mathcal{IW}_r(\Psi, \nu_0),$$

where Ψ is the $r \times r$ scale matrix and ν_0 represents the prior degrees of freedom. Following an Empirical Bayes approach to covariance shrinkage (Ledoit & Wolf, 2004), we set the prior scale matrix Ψ to be an isotropic target matrix that preserves the total variance of the observed data

$$\Psi = \nu_0 \hat{\sigma}^2 I_r = \nu_0 \frac{\operatorname{tr}(\hat{\Sigma})}{r} I_r.$$

As shown by Murphy (2012), the posterior distribution $p(\Sigma \mid S) \propto p(S \mid \Sigma)p(\Sigma)$ is also an Inverse-Wishart distribution such that

$$p(\Sigma \mid S) \propto |\Sigma|^{-\frac{n+(\nu_0+r+1)}{2}} \exp\left(-\frac{1}{2}\operatorname{tr}(\Sigma^{-1}(S+\Psi))\right).$$

Following the posterior distribution, the maximum a posteriori (MAP) estimation of Σ is:

$$\Sigma = \frac{S + \Psi}{n + \nu_0 + r + 1}$$

Substituting our definitions of S and Ψ yields:

$$S + \Psi = n\hat{\Sigma} + \nu_0 \frac{\operatorname{tr}(\hat{\Sigma})}{r} I_r.$$

Setting $\nu_0 = r$, we further have

$$S + \Psi = n\hat{\Sigma} + \operatorname{tr}(\hat{\Sigma}) I_r.$$

Dividing this combined scatter matrix by the total effective degrees of freedom yields the standard Bayes-shrunk covariance. However, in few shot learning regimes, especially when the number of shots k is extremely small, e.g. $k = 2$, the covariance $\hat{\Sigma}$ could be highly unstable as shown in Appendix F.4. We therefore introduce a covariance inflation parameter $\beta \in (0, 1]$, explicitly inflating the estimated variance by a factor of $\frac{1}{\beta}$:

$$\Sigma = \frac{1}{\beta} \frac{n\hat{\Sigma} + \operatorname{tr}(\hat{\Sigma}) I_r}{n + 2r + 1}.$$

Substituting $n = K(k-1)$ and taking the matrix inverse yields our final regularized precision matrix:

$$\Sigma^{-1} = \beta(K(k-1) + 2r + 1)\left[K(k-1)\hat{\Sigma} + \operatorname{tr}(\hat{\Sigma}) I_r\right]^{-1}.$$

C DEFTS ALGORITHM

In this section, we provide the algorithm for DEFTS in Algorithm 1, which summarizes our discussions about DEFTS in Section 2.3

Algorithm 1 DEFTS Prototype Generation and Inference

Require: K text embeddings $\{e_c\}_{c=1}^{K}$, k support image embeddings per class $\{x_{c,i}\}_{i=1}^{k}$
Ensure: Estimated image prototypes $\{\hat{\mu}_c\}_{c=1}^{K}$ for classification

1: Compute the centralized scatter matrix S from the text embeddings $\{e_c\}_{c=1}^{K}$.
2: Eigendecompose S to obtain the PCA basis $Q \in \mathbb{R}^{d\times r}$, choosing r to retain 95% of the variance.
3: **for** each class $c \in \{1, \ldots, K\}$ **do**
4: Project text embedding: $e'_c \leftarrow Q^\top e_c$
5: Project support images: $b_{c,i} \leftarrow Q^\top x_{c,i} \quad \forall i \in \{1, \ldots, k\}$
6: **end for**
7: Estimate the prior variance $\hat{\tau}^2$ and the pooled likelihood precision Σ^{-1} directly from the projected k-shot support set $\{b_{c,i}\}$ via Empirical Bayes as shown in (2) and (3).
8: **for** each class $c \in \{1, \ldots, K\}$ **do**
9: Compute the posterior mean $\hat{\theta}_c$ in the subspace following Equation (5).
10: Reconstruct the high-dimensional prototype: $\hat{\mu}_c \leftarrow Q\hat{\theta}_c$.
11: **end for**
12: During inference, for a query image I, compute the logit for class c as:

$$s_{\mathrm{DEFTS}}(F_v(I))_c = \frac{F_v(I)^\top}{\|F_v(I)\|} \frac{\hat{\mu}_c}{\|\hat{\mu}_c\|}.$$

D Structural Comparison Between DEFTS and TAMP

Although both DEFTS and TAMP (Goswami et al., 2026) exploit the text embedding space, they employ fundamentally different estimation mechanisms. TAMP uses the text space primarily as a geometric projection space and combines the projected visual prototype with the text prototype via a scalar interpolation coefficient. In contrast, DEFTS treats the text-derived subspace as a statistical estimation space, performing uncertainty-aware, direction-dependent Bayesian shrinkage. We formalize this distinction below.

Let $\bar{x}_c$ denote the empirical mean of the k-shot support image embeddings for class c in the ambient space, and let e_c denote the corresponding text embedding. TAMP constructs an orthonormal basis $Q' \in \mathbb{R}^{d\times r}$ from the uncentered text embeddings and defines the corresponding projection matrix as

$$P = Q'Q'^{\top}.$$

Since e_c lies in the span of Q', we have $Q'Q'^{\top}e_c = e_c$. TAMP then forms its class prototype by interpolating between the projected image prototype and the text prototype:

$$\begin{aligned} W_{\mathrm{TAMP}} &= \lambda Q'Q'^{\top}\bar{x}_c + (1-\lambda)e_c \\ &= Q'\left[\lambda I_r(Q'^{\top}\bar{x}_c) + (1-\lambda)I_r(Q'^{\top}e_c)\right]. \end{aligned} \tag{6}$$

Thus, within the text subspace, TAMP applies the same scalar weight to every direction. The relative contribution of the visual estimate and the text prototype is therefore controlled by a single global coefficient λ.

DEFTS, instead, constructs a centered PCA basis $Q \in \mathbb{R}^{d\times r}$ from the text embeddings and performs Bayesian estimation in the resulting subspace. Define

$$\bar{b}_c = Q^{\top}\bar{x}_c, \qquad e'_c = Q^{\top}e_c.$$

Following Eq. (5), the posterior mean in the PCA coordinates is

$$\hat{\theta}_c = \left(\Omega_0 + k\Sigma^{-1}\right)^{-1}\left(k\Sigma^{-1}\bar{b}_c + \Omega_0 e'_c\right),$$

and the corresponding prototype in the original embedding space is

$$\begin{aligned} W_{\mathrm{DEFTS}} &= Q\hat{\theta}_c \\ &= Q\left[\left(\Omega_0 + k\Sigma^{-1}\right)^{-1}k\Sigma^{-1}(Q^{\top}\bar{x}_c) + \left(\Omega_0 + k\Sigma^{-1}\right)^{-1}\Omega_0(Q^{\top}e_c)\right]. \end{aligned} \tag{7}$$

For notational convenience, let

$$A = \left(\Omega_0 + k\Sigma^{-1}\right)^{-1}, \qquad B = k\Sigma^{-1}, \qquad C = \Omega_0.$$

Then

$$W_{\mathrm{DEFTS}} = Q\left[AB(Q^{\top}\bar{x}_c) + AC(Q^{\top}e_c)\right]. \tag{8}$$

These expressions reveal three important structural differences.

Scalar interpolation versus uncertainty-aware anisotropic fusion. TAMP uses a scalar coefficient λ and the identity matrix I_r, effectively treating all directions within the text subspace identically. DEFTS, instead, weights the visual observation and text prior through the dense matrices AB and AC, which depend on the estimated covariance Σ and prior precision Ω_0. Consequently, the relative contribution of the image evidence and the text prior varies across directions based on their estimated uncertainty. Directions with lower visual variance receive relatively greater weight from the empirical image evidence, whereas high-variance directions are shrunk more strongly toward the text prior. DEFTS therefore performs uncertainty-aware, direction-dependent prototype estimation rather than a globally weighted interpolation.

This distinction also provides a useful interpretation of scalar interpolation. If the uncertainty structure were isotropic, i.e.,

$$\Sigma = \sigma^2 I_r \quad \text{and} \quad \Omega_0 = \omega^2 I_r,$$

the corresponding Bayesian fusion would reduce exactly to a scalar weighting of the visual estimate and the prior. Thus, scalar interpolation can be viewed as the isotropic special case of the precision-weighted fusion employed by DEFTS. DEFTS extends this formulation by allowing the estimated uncertainty to dynamically differ across directions.

Scalar hyperparameter selection versus Empirical Bayes estimation. TAMP determines the relative contribution of the projected image prototype and text prototype through the scalar coefficient λ, which must be externally tuned via validation. DEFTS estimates the visual uncertainty structure directly from the available training data using the Empirical Bayes procedure. The resulting precision matrix dictates how strongly the image evidence and text prior are combined along each axis, thus avoiding the need for searching hyperparameter for each dataset.

Uncentered text span versus centered PCA subspace. The two methods also differ in their chosen representation space. TAMP constructs Q' directly from the uncentered text embeddings, operating within their uncentered linear span. DEFTS first centers the text embeddings before constructing a PCA basis that retains a fixed fraction of their variance. Empirically, our ablation in Appendix F.2 suggests that the centered construction yields more effective classification, which is why it forms the foundation of DEFTS.

Overall, while TAMP and DEFTS share the intuition that text embeddings provide a highly structural low-dimensional space for few-shot visual classification, they exploit it differently. TAMP applies a uniform scalar interpolation after projecting the visual prototype into the text space, whereas DEFTS formulates prototype estimation as uncertainty-aware Bayesian shrinkage within a centered PCA subspace. This capability to weight visual evidence and text priors differently across dimensions is critical in few-shot cases, where visual prototype estimates remain highly uncertain. We summarize the differences between TAMP and DEFTS in Table 4.

Table 4: Key structural and methodological differences between the TAMP baseline and the proposed DEFTS framework.

Component	**TAMP**	**DEFTS**
Image representation	Projected image mean	Projected image mean
Text representation	Text prototype	Text prototype as prior mean
Subspace	Uncentered text span	Centered varaince reduced PCA subspace
Fusion	Scalar interpolation	Precision-weighted fusion
Image weighting	Global λ	Direction-dependent
Uncertainty	Not modeled explicitly	Estimated covariance
Estimation	Validation-selected λ	Empirical Bayes
Output	Interpolated prototype	Posterior-mean prototype

E More Experiments details

In this section, we provide more details for the experiments in Section 2.2 and 3.

E.1 Datasets Details

In Table 5, we provide more details about the 12 datasets considered in this paper, including the total number of images, the number of classes, and the minimum and maximum images per class.

Table 5: Summary of the 12 evaluation datasets, detailing the total number of images, the number of classes, and the minimum and maximum images per class.

Dataset	Total Images	# Classes	Min/Class	Max/Class
CIFAR10	10,000	10	1,000	1,000
CIFAR100	10,000	100	100	100
EuroSAT	27,000	10	2,000	3,000
Food101	25,250	101	250	250
OxfordIIITPet	3,669	37	88	100
FGVCAircraft	3,333	100	33	34
Flowers102	8,189	102	40	258
DTD	1,880	47	40	40
ImageNet	50,000	1,000	50	50
Caltech101	8,242	100	31	800
StanfordCars	8,041	196	24	68
SUN397	19,850	397	50	50

E.2 Prompt Templates

In Section 3, we evaluate the performance of DEFTS using the standard prompt templates introduced by CoOp (Zhou et al., 2022). This section details the exact text prompt applied to each corresponding dataset.

Table 6: Text prompt templates utilized for each of the 12 evaluation datasets. The placeholder {c} denotes the specific class name.

Dataset	Prompt Template
CIFAR10	a photo of a {c}.
CIFAR100	a photo of a {c}.
EuroSAT	a centered satellite photo of {c}.
Food101	a photo of {c}, a type of food.
OxfordIIITPet	a photo of a {c}, a type of pet.
FGVCAircraft	a photo of a {c}, a type of aircraft.
Flowers102	a photo of a {c}, a type of flower.
DTD	{c} texture.
ImageNet	a photo of a {c}.
Caltech101	a photo of a {c}.
StanfordCars	a photo of a {c}.
SUN397	a photo of a {c}.

E.3 Experiments Details for Observations

In Section 2.2, we present 3 observations for investigating the text embedding subspace. In this section, we provide details for the experiments supporting our observations.

Experiments for Observation 1. For Observation 1, we train an NCM classifier for each dataset. We first randomly sample 80% of the images from each class as training data. Then, we project all image features into the text subspace, as detailed in Section 2.3 but without centering first. The mean of the image embeddings within this subspace are estimated by taking average and used for performing nearest class mean classification. The classification accuracy is evaluated on the remaining 20% of the images, essentially serving as the validation set. This procedure is repeated over 10 independent runs with varying random seeds, and the final reported accuracy is the average across all runs.

Experiments for Observation 2. The experimental setup for Observation 2 is similar to that for Observation 1. However, in this experiment, we randomly sample k images from each class to serve as the training data, while using all remaining images as the validation set.

Experiments for Observation 3. Similarly, the experiment for Observation 3 builds upon the one for Observation 1. The difference is that we now first reduce the dimensionality of the subspace, as described in Section 2.3.

F Additional Experiments Results

This section includes additional experiments results.

F.1 Text Subspace with Different Template

This additional experiment provides evidence that the observations discussed in Section 2.2 are robust to the choice of prompt template. To this end, we perform the same experiments for the discriminative power of text embedding space in Figure 1 with different prompt templates.

Figure 7 evaluates three different prompt templates: (1) the class name alone; (2) a standard fixed template "a photo of a {class}"; and (3) the hand-crafted templates utilized in CoOp (Zhou et al., 2022), as detailed in Appendix E.2. As expected, improving the template quality steadily increases CLIP's baseline zero-shot accuracy. However, the results provide evidence suggesting that the template quality has a more negligible impact on the discriminative power of the text embedding space itself.

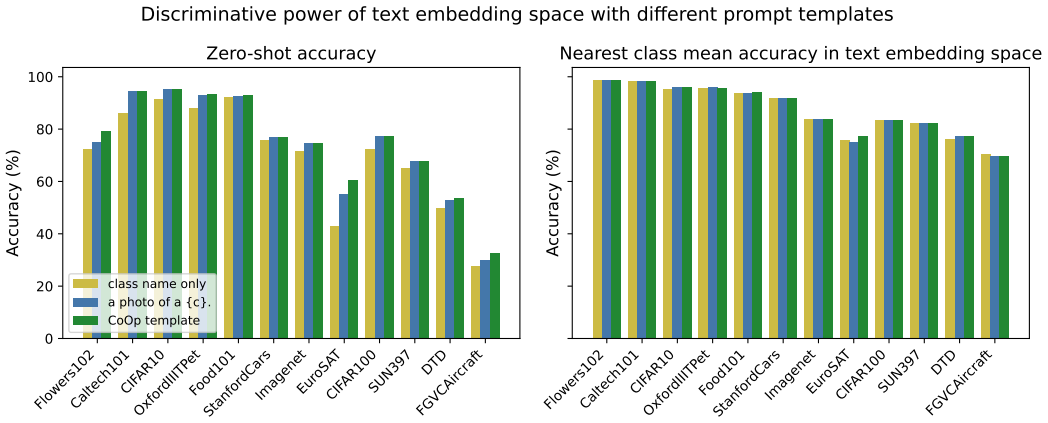

Figure 7: CLIP's zero-shot accuracy and Nearest Class Mean (NCM) accuracy across three different prompt templates. To compute the NCM, all image embeddings are first projected into the text-embedding subspace to estimate the mean embedding for each class. A test image is then classified by assigning it to the class with the closest estimated mean.

F.2 Centered PCA Projection

In Section 2.3, DEFTS proposes to project image embeddings into a centered PCA subspace, rather than the original, uncentered subspace. To justify this design choice, we perform an ablation study comparing centered versus uncentered PCA projections while retaining either 95% or 99.9% of the total variance. Table 7 suggests that for 95% variance kept, the centered PCA wins uncentered one by larger margin, around 2-3%, when 99.9% variance are kept, centered one is often slightly better than uncentered one.

F.3 Empirical contribution of total average deviation

In this experiment, we investigate the contribution of the sample variance $\hat{\sigma}^2$ to the averaged deviation $\frac{1}{r}\overline{\|\delta\|^2}$. Table 8 shows the results of the ratio of the sample variance $\hat{\sigma}^2$ to the averaged empirical deviation $\frac{1}{r}\overline{\|\delta\|^2}$ across 12 datasets for different number of shots k. It suggests that this ratio is empirically ranges from 2% to 16%, varying across different datasets, but more stable with different number of shots k.

Table 7: DEFTS's average accuracy (%) across all 12 datasets for centered and uncentered PCA variants across 95% and 99.9% explained variance thresholds for varying support set sizes (k). The best performance for each k is highlighted in **bold red**. Results are averaged over 10 random seeds and reported as percentages.

k	95% Variance		99.9% Variance	
	Centered	**Uncentered**	**Centered**	**Uncentered**
1	**76.97**	73.54	76.63	76.73
2	79.60	76.65	**79.72**	79.56
4	**81.77**	79.25	81.68	81.64
8	**83.59**	81.37	83.48	83.27
16	84.68	82.51	**84.70**	84.41

Table 8: Ratio of the sample variance $\hat{\sigma}^2$ to the averaged empirical deviation $\frac{1}{r}\overline{\|\delta\|^2}$ across 12 datasets for different number of shots k. Results are averaged over 10 random seeds and reported as percentages.

Dataset	**Number of shots (k)**				
	$k=2$	$k=4$	$k=8$	$k=16$	**Average**
CIFAR10	5.09	4.94	3.81	5.21	4.76
CIFAR100	5.97	6.00	6.12	6.14	6.06
Caltech101	6.81	6.61	6.43	6.52	6.59
DTD	11.16	11.47	11.04	10.95	11.16
EuroSAT	2.03	2.82	3.23	3.22	2.83
FGVCAircraft	7.46	7.83	7.69	7.72	7.67
Flowers102	5.31	5.70	5.38	5.37	5.44
Food101	11.42	11.69	12.04	12.15	11.82
ImageNet	16.27	16.00	16.07	16.11	16.11
OxfordPet	3.24	3.20	3.42	3.24	3.27
SUN397	14.52	14.87	14.61	14.71	14.68
StanfordCars	8.91	9.20	8.99	8.98	9.02
Average	**8.18**	**8.36**	**8.24**	**8.36**	**8.28**

F.4 Covariance Estimation for Different Number of Shots

In Section 2.3, we introduce a covariance inflation parameter β to account for the fact that the covariance estimation is often problematic especially when the number of shots k is small. In this section, we presents experiments results in Figure 8 demonstrating that when the number of shots is small, the estimation of the covariance could be highly unstable.

F.5 Impact of the covariance inflation parameter β for DEFTS

In this additional experiments, we investigate the impact of the covariance inflation parameter β on the accuracy of DEFTS. In Figure 9, we present DEFTS' average accuracy with different β across 12 datasets. The results suggest that when the number of shots k is small, it is preferable to choose small β for DEFTS. However, when the number of shots is larger, the choice of β is less critical.

F.6 PCA of the text embedding space

In **Observation 3**, we provide evidence showing that the discriminative power of the text embedding space will decrease if we reduce the dimension of the space using PCA. Note that the discriminative power in **Observation 3** is evaluated using NCM with image embeddings estimated using all available training samples. In this additional experiment, we use only a few shots of training samples and evaluate the nearest class mean (NCM) accuracy after removing different amounts of variance from

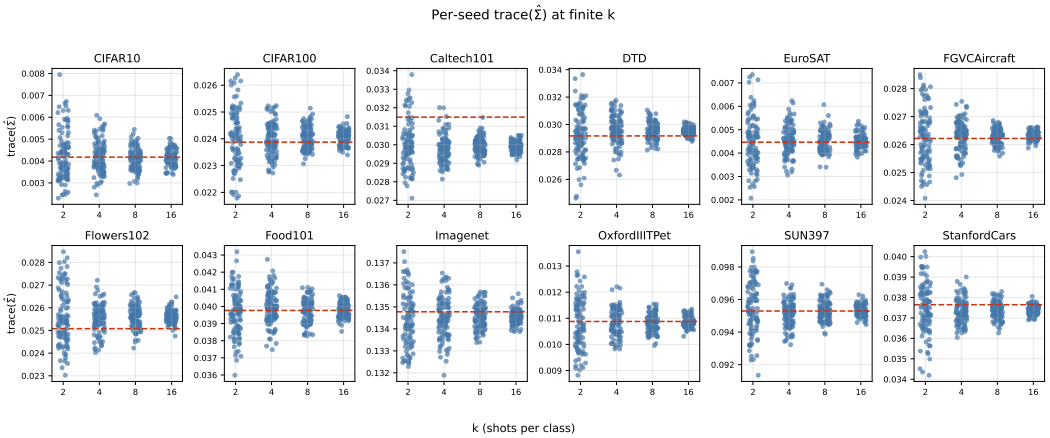

Figure 8: Trace of the empirical sample covariance matrix $\hat{\Sigma}$ across 12 datasets for different number of shots k. Blue dots represent individual estimates across 100 random seeds, while the red dashed line represents the true population trace estimated using all samples.

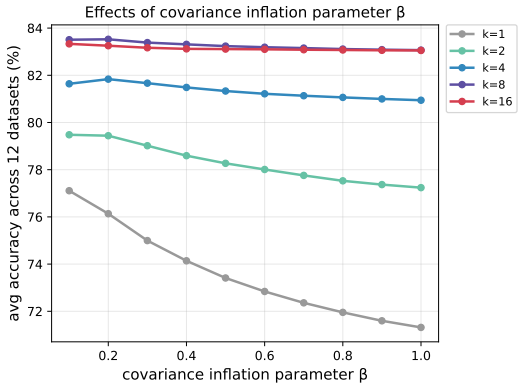

Figure 9: DEFTS' average accuracy with different covariance inflation parameter β across 12 datasets.

the text embedding space. The results in Table 9 suggests that for a given number of shots k, reducing the variance of the text embedding space by 5% yields the highest NCM accuracy within the text subspace.

F.7 Detailed experiment results with different number of shots

In this section, we perform the same simulations for Table 1 with different number of shots $k \in \{2, 4, 8, 16\}$. The results are presented in Table 10 to 13.

F.8 Experiments with different model backbone

In this section, we evaluate the performances of DEFTS and other baselines with different model backbone considered in the main paper. To be specific, we consider ViT-B/32 and ViT-H-14, different than ViT-L/14 used Section 3. The experiments in this section are performed exactly as the one for Table 2, with only model backbone changed. Results shown in Table 14 demonstrate that the DEFTS improvements over other competing methods are consistent across different model backbones.

Table 9: Nearest class mean (NCM) accuracy after reducing different amount of variance of the text embedding space. Bold marks the best representation at each k.

	$k=1$	$k=2$	$k=4$	$k=8$	$k=16$
raw (no projection)	50.86	64.99	74.14	79.17	81.82
text span (full)	65.26	74.13	79.36	82.21	83.82
PCA 95% variance	**67.87**	**75.95**	**80.40**	**82.85**	**84.23**
PCA 90% variance	67.71	75.74	80.17	82.60	83.93
PCA 85% variance	67.03	75.20	79.62	82.09	83.48
PCA 80% variance	66.23	74.38	78.91	81.45	82.85
PCA 75% variance	64.86	72.92	77.55	80.10	81.54
PCA 70% variance	63.24	71.45	76.19	78.92	80.38
PCA 65% variance	61.81	70.16	75.18	77.94	79.46
PCA 60% variance	59.68	68.26	73.54	76.41	77.99
PCA 55% variance	56.85	65.34	70.70	73.65	75.24
PCA 50% variance	53.53	61.97	67.66	70.88	72.65

Table 10: Number of shots $k = 2$. Classification accuracy (%) and performance improvements across datasets. The values in parentheses denote the absolute performance change after ensembling with DEFTS and the 90% confidence interval. Positive gains are highlighted in green, and negative changes in red.

Dataset	ZS	SimpleShot	Linear Probe	Tip-Adapter	APE	TIP-X	TAMP+ZS	LDA+ZS	DEFTS
C10	94.89 (+0.38 ±0.18)	79.44 (+15.84 ±3.08)	81.42 (+13.90 ±2.77)	94.85 (+0.24 ±0.24)	94.34 (+0.45 ±0.61)	94.48 (+0.54 ±0.19)	94.80 (+0.39 ±0.16)	94.80 (+0.41 ±0.12)	95.29
C100	77.27 (+0.29 ±0.30)	48.35 (+28.50 ±0.67)	53.39 (+23.38 ±2.10)	77.53 (+0.00 ±0.00)	77.65 (+0.00 ±0.00)	68.43 (+2.32 ±1.02)	77.38 (+0.16 ±0.18)	77.27 (+0.29 ±0.30)	76.85
Cal101	94.62 (+0.53 ±0.31)	91.13 (+4.47 ±0.48)	91.62 (+3.70 ±0.72)	94.64 (+0.00 ±0.00)	94.83 (+0.00 ±0.00)	94.59 (+0.53 ±0.12)	94.95 (+0.41 ±0.28)	94.62 (+0.53 ±0.31)	95.58
DTD	53.03 (+4.98 ±1.87)	48.05 (+14.29 ±1.38)	48.95 (+14.47 ±2.48)	57.09 (+1.70 ±1.64)	56.33 (+0.31 ±0.57)	56.20 (+2.18 ±0.91)	56.82 (+1.28 ±1.23)	53.03 (+4.98 ±1.87)	62.37
Euro	61.17 (+5.66 ±2.26)	67.60 (+5.18 ±2.74)	64.66 (+8.25 ±4.37)	66.87 (+1.39 ±0.94)	68.90 (+2.06 ±2.67)	66.81 (+1.77 ±0.54)	61.07 (+6.26 ±2.24)	67.54 (+4.27 ±3.14)	68.67
Air.	32.18 (+0.25 ±0.47)	34.62 (+8.55 ±1.71)	33.55 (+10.36 ±1.46)	34.76 (+0.00 ±0.00)	37.49 (+0.74 ±1.35)	43.29 (+1.90 ±0.37)	35.56 (+0.15 ±0.32)	32.18 (+0.25 ±0.47)	44.57
Flow.	78.40 (+15.16 ±0.94)	92.75 (+2.85 ±0.66)	91.44 (+3.91 ±1.22)	88.02 (+5.71 ±1.77)	93.00 (+1.19 ±0.59)	92.84 (+1.66 ±0.23)	91.22 (+1.15 ±0.57)	92.81 (+1.64 ±1.12)	95.14
Food	93.11 (+0.23 ±0.11)	75.70 (+17.06 ±0.56)	77.23 (+15.33 ±1.07)	93.19 (+0.00 ±0.00)	93.19 (+0.00 ±0.00)	87.53 (+2.34 ±0.42)	93.17 (+0.21 ±0.10)	93.11 (+0.23 ±0.11)	92.76
INet	74.35 (+0.00 ±0.00)	54.78 (+20.90 ±0.27)	57.89 (+17.77 ±0.23)	74.40 (+0.00 ±0.00)	74.50 (+0.00 ±0.00)	64.45 (+0.00 ±0.00)	75.68 (+0.00 ±0.00)	74.35 (+0.00 ±0.00)	75.68
Pet	92.82 (+0.88 ±0.30)	66.96 (+26.31 ±1.72)	70.14 (+23.02 ±1.69)	92.87 (+0.12 ±0.23)	93.01 (+0.13 ±0.24)	93.42 (+0.35 ±0.45)	92.85 (+1.17 ±0.25)	92.82 (+0.88 ±0.30)	93.27
SUN	67.35 (+0.00 ±0.00)	55.13 (+17.03 ±0.38)	56.14 (+16.02 ±1.04)	67.92 (+0.00 ±0.00)	68.05 (+0.00 ±0.00)	63.45 (+0.00 ±0.00)	69.40 (+0.00 ±0.00)	67.35 (+0.00 ±0.00)	72.16
Cars	76.61 (+0.00 ±0.00)	59.02 (+22.59 ±0.79)	62.43 (+18.77 ±1.27)	76.72 (+0.00 ±0.00)	76.82 (+0.00 ±0.00)	76.83 (+1.62 ±0.65)	76.89 (+0.04 ±0.05)	76.61 (+0.00 ±0.00)	81.64
AVG	**74.65** (+2.36 ±0.69)	**64.46** (+15.30 ±1.30)	**65.74** (+14.07 ±1.07)	**76.57** (+0.76 ±0.31)	**77.34** (+0.41 ±0.25)	**75.19** (+1.27 ±0.18)	**76.65** (+0.94 ±0.32)	**76.37** (+1.12 ±0.37)	**79.50**

G Hyperparameter Selection and Tuning

This section reports our choices for choosing and searching hyperparameters of each method to ensure complete transparency of our experiments.

G.1 Hyperparameters for Each Method

We first detail the hyperparameters used in each evaluated method and their ensemble variants. For notational simplicity, we adopt a slightly different convention here compared to Section 2.1. Specifically, every scoring function defined below operates on a single test image embedding $x \in \mathbb{R}^d$ and yields a score vector $\text{score}(x) \in \mathbb{R}^K$ across all K classes. The c-th entry, $\text{score}(x)_c$, represents the logit for class c, such that the final predicted label is given by $\hat{y} = \arg\max_c \text{score}(x)_c$. Throughout the section, we will use w to denote DEFTS's combining weight whenever DEFTS's score is added to an existing method.

Table 11: Number of shots $k = 4$. Classification accuracy (%) and performance improvements across datasets. The values in parentheses denote the absolute performance change after ensembling with DEFTS and the 90% confidence interval. Positive gains are highlighted in green, and negative changes in red.

Dataset	ZS	SimpleShot	Linear Probe	Tip-Adapter	APE	TIP-X	TAMP+ZS	LDA+ZS	DEFTS
C10	94.89 (+0.66 ±0.12)	87.68 (+7.94 ±1.13)	89.68 (+5.85 ±0.63)	95.05 (+0.32 ±0.26)	94.22 (+0.36 ±0.61)	94.34 (+0.45 ±0.15)	94.82 (+0.64 ±0.24)	94.81 (+0.24 ±0.25)	95.56
C100	77.27 (+1.83 ±0.51)	63.50 (+15.21 ±0.49)	69.05 (+9.51 ±0.67)	77.96 (+0.77 ±0.49)	78.27 (+1.11 ±0.55)	69.32 (+0.96 ±0.92)	78.08 (+1.12 ±0.46)	79.66 (+0.55 ±0.76)	78.60
Cal101	94.62 (+1.64 ±0.52)	94.16 (+2.52 ±0.27)	94.78 (+1.67 ±0.25)	94.99 (+1.35 ±0.47)	95.00 (+1.33 ±0.47)	95.12 (+0.15 ±0.19)	95.42 (+0.80 ±0.51)	97.08 (+0.27 ±0.45)	96.55
DTD	53.03 (+12.93 ±1.02)	57.94 (+9.92 ±0.96)	60.61 (+7.38 ±0.96)	60.56 (+5.81 ±2.12)	64.12 (+2.50 ±1.53)	61.36 (+0.00 ±0.00)	60.88 (+4.91 ±1.07)	66.33 (+1.80 ±1.68)	66.86
Euro	61.17 (+9.87 ±0.67)	76.66 (+2.16 ±2.37)	78.53 (-0.57 ±3.00)	75.48 (-0.55 ±1.83)	77.20 (+0.29 ±2.53)	67.21 (+0.25 ±0.28)	60.57 (+10.41 ±1.11)	82.42 (-0.43 ±1.37)	71.49
Air.	32.18 (+14.15 ±1.34)	41.84 (+7.15 ±0.88)	40.61 (+8.51 ±1.78)	39.60 (+6.85 ±2.16)	40.57 (+3.95 ±1.65)	45.59 (+1.86 ±0.66)	43.20 (+1.74 ±1.12)	46.20 (+1.33 ±1.56)	49.00
Flow.	78.40 (+18.80 ±0.25)	96.06 (+1.33 ±0.17)	96.65 (+0.24 ±0.64)	94.96 (+1.90 ±0.57)	96.15 (+0.85 ±0.25)	94.19 (+1.05 ±0.18)	94.11 (+3.00 ±0.39)	97.80 (+0.09 ±0.05)	97.22
Food	93.11 (+0.51 ±0.07)	85.61 (+7.18 ±0.33)	87.33 (+5.53 ±0.55)	93.34 (+0.16 ±0.11)	93.30 (+0.11 ±0.11)	89.59 (+0.42 ±0.33)	93.37 (+0.24 ±0.13)	93.66 (+0.09 ±0.13)	92.96
INet	74.35 (+2.54 ±0.47)	66.04 (+11.34 ±0.34)	69.12 (+8.56 ±0.26)	75.39 (+0.00 ±0.00)	75.84 (+0.00 ±0.00)	67.15 (+0.00 ±0.00)	78.31 (+0.02 ±0.04)	78.46 (+0.00 ±0.00)	77.71
Pet	92.82 (+1.80 ±0.40)	80.84 (+13.60 ±0.79)	82.02 (+12.05 ±1.56)	93.25 (+1.35 ±0.40)	93.20 (+1.50 ±0.34)	93.60 (+0.56 ±0.20)	92.86 (+1.83 ±0.42)	93.53 (+1.16 ±0.48)	94.54
SUN	67.35 (+5.17 ±0.53)	65.96 (+8.65 ±0.27)	67.94 (+6.67 ±0.36)	70.34 (+3.32 ±0.80)	70.88 (+2.61 ±1.12)	66.02 (+0.00 ±0.00)	73.70 (-0.00 ±0.01)	74.68 (+0.22 ±0.21)	74.77
Cars	76.61 (+6.09 ±0.97)	69.67 (+14.78 ±0.62)	73.73 (+10.81 ±0.33)	77.71 (+4.68 ±1.03)	78.63 (+5.01 ±0.72)	79.65 (+0.59 ±0.58)	80.30 (+1.67 ±0.56)	80.81 (+2.80 ±0.96)	84.65
AVG	**74.65** (+6.33 ±0.91)	**73.83** (+8.48 ±0.73)	**75.84** (+6.35 ±0.66)	**79.05** (+2.16 ±0.45)	**79.78** (+1.64 ±0.36)	**76.93** (+0.52 ±0.13)	**78.80** (+2.20 ±0.46)	**82.12** (+0.68 ±0.26)	**81.66**

Table 12: Number of shots $k = 8$. Classification accuracy (%) and performance improvements across datasets. The values in parentheses denote the absolute performance change after ensembling with DEFTS and the 90% confidence interval. Positive gains are highlighted in green, and negative changes in red.

Dataset	ZS	SimpleShot	Linear Probe	Tip-Adapter	APE	TIP-X	TAMP+ZS	LDA+ZS	DEFTS
C10	94.89 (+0.59 ±0.18)	92.41 (+2.75 ±0.68)	93.42 (+2.13 ±1.41)	95.17 (+0.23 ±0.11)	95.39 (+0.20 ±0.11)	94.64 (+0.07 ±0.05)	94.97 (+0.53 ±0.16)	95.81 (+0.12 ±0.09)	95.60
C100	77.27 (+3.21 ±0.19)	70.57 (+9.78 ±0.60)	75.39 (+4.52 ±0.53)	78.19 (+2.51 ±0.22)	79.60 (+1.16 ±0.46)	70.81 (+0.00 ±0.00)	78.54 (+1.91 ±0.41)	81.87 (+0.10 ±0.27)	80.25
Cal101	94.62 (+2.86 ±0.16)	95.53 (+1.56 ±0.29)	96.26 (+0.45 ±0.29)	95.90 (+1.51 ±0.42)	96.20 (+1.23 ±0.41)	95.83 (+0.00 ±0.00)	96.45 (+0.92 ±0.22)	97.61 (+0.03 ±0.09)	97.08
DTD	53.03 (+17.40 ±0.44)	64.58 (+6.73 ±0.87)	67.48 (+2.91 ±1.09)	66.25 (+4.47 ±0.96)	68.69 (+2.96 ±1.06)	63.94 (+0.00 ±0.00)	62.90 (+7.17 ±0.49)	71.49 (+1.39 ±0.65)	70.56
Euro	61.17 (+12.86 ±0.82)	80.45 (+3.30 ±0.34)	85.12 (-1.21 ±1.49)	80.26 (+1.54 ±1.11)	83.52 (+0.56 ±0.50)	69.18 (+0.19 ±0.36)	60.61 (+12.58 ±1.00)	87.30 (+0.43 ±0.56)	74.35
Air.	32.18 (+21.41 ±0.55)	47.96 (+6.46 ±0.47)	49.27 (+4.46 ±0.66)	46.50 (+6.65 ±1.25)	43.00 (+8.34 ±1.33)	47.93 (+0.00 ±0.00)	47.65 (+5.36 ±0.94)	55.29 (+2.25 ±0.64)	53.98
Flow.	78.40 (+19.66 ±0.20)	97.48 (+0.61 ±0.11)	98.19 (-0.46 ±0.22)	96.77 (+1.06 ±0.14)	97.59 (+0.46 ±0.18)	94.62 (+0.14 ±0.18)	95.04 (+2.92 ±0.31)	98.66 (+0.12 ±0.07)	98.06
Food	93.11 (+0.54 ±0.10)	89.05 (+4.10 ±0.16)	90.39 (+2.54 ±0.32)	93.43 (+0.25 ±0.14)	93.51 (+0.27 ±0.18)	90.28 (+0.00 ±0.00)	93.59 (+0.03 ±0.09)	93.93 (-0.10 ±0.12)	93.15
INet	74.35 (+6.31 ±0.09)	72.64 (+7.20 ±0.12)	75.74 (+4.19 ±0.13)	76.94 (+3.71 ±0.21)	77.73 (+2.99 ±0.32)	74.10 (-0.34 ±0.70)	79.54 (+0.61 ±0.20)	81.10 (+0.13 ±0.08)	79.93
Pet	92.82 (+2.26 ±0.24)	86.71 (+8.39 ±0.71)	90.02 (+4.52 ±0.96)	93.61 (+1.44 ±0.38)	93.83 (+1.33 ±0.41)	93.51 (+0.31 ±0.12)	93.01 (+2.11 ±0.27)	95.24 (+0.18 ±0.29)	95.25
SUN	67.35 (+9.75 ±0.14)	72.08 (+5.19 ±0.25)	74.04 (+2.93 ±0.37)	73.32 (+3.83 ±0.24)	74.37 (+2.98 ±0.29)	70.01 (+0.00 ±0.00)	75.71 (+1.11 ±0.29)	78.27 (+0.22 ±0.13)	77.27
Cars	76.61 (+9.98 ±0.20)	76.12 (+10.39 ±0.37)	81.96 (+4.30 ±0.75)	80.04 (+6.24 ±0.38)	81.23 (+4.96 ±0.37)	81.22 (+0.00 ±0.00)	82.03 (+4.38 ±0.42)	86.28 (+1.01 ±0.22)	86.66
AVG	**74.65** (+8.90 ±1.10)	**78.80** (+5.54 ±0.47)	**81.44** (+2.61 ±0.36)	**81.37** (+2.79 ±0.35)	**82.06** (+2.29 ±0.38)	**78.84** (+0.03 ±0.06)	**80.00** (+3.30 ±0.54)	**85.24** (+0.49 ±0.13)	**83.51**

Table 13: Number of shots $k = 16$. Classification accuracy (%) and performance improvements across datasets. The values in parentheses denote the absolute performance change after ensembling with DEFTS and the 90% confidence interval. Positive gains are highlighted in green, and negative changes in red.

Dataset	ZS	SimpleShot	Linear Probe	Tip-Adapter	APE	TIP-X	TAMP+ZS	LDA+ZS	DEFTS
C10	94.89 (+0.70 ±0.11)	94.75 (+1.20 ±0.42)	95.35 (+0.66 ±0.47)	95.52 (+0.16 ±0.11)	95.76 (+0.08 ±0.16)	95.26 (+0.00 ±0.00)	95.07 (+0.53 ±0.12)	96.04 (+0.23 ±0.18)	95.67
C100	77.27 (+4.28 ±0.13)	74.89 (+6.73 ±0.43)	79.65 (+1.60 ±0.42)	79.13 (+2.46 ±0.15)	80.40 (+1.45 ±0.28)	73.08 (+0.00 ±0.00)	78.95 (+2.60 ±0.20)	82.93 (+0.09 ±0.19)	81.41
Cal101	94.62 (+2.89 ±0.10)	95.99 (+1.48 ±0.24)	97.09 (+0.27 ±0.08)	96.41 (+1.21 ±0.28)	96.87 (+0.75 ±0.21)	96.08 (+0.20 ±0.19)	96.64 (+0.90 ±0.16)	97.95 (-0.01 ±0.12)	97.36
DTD	53.03 (+19.93 ±0.44)	69.00 (+5.25 ±0.45)	74.12 (-0.75 ±0.84)	69.44 (+5.04 ±0.60)	72.93 (+2.23 ±0.62)	67.14 (-0.05 ±0.28)	65.07 (+7.76 ±0.74)	75.43 (+1.08 ±0.81)	72.95
Euro	61.17 (+13.93 ±0.60)	81.39 (+2.49 ±0.33)	89.47 (-3.98 ±1.44)	82.39 (+0.74 ±0.86)	86.81 (+0.65 ±0.30)	72.78 (+0.58 ±1.66)	60.73 (+14.40 ±0.83)	91.19 (-0.19 ±0.27)	75.44
Air.	32.18 (+24.43 ±0.53)	52.40 (+5.47 ±0.57)	56.20 (+0.97 ±0.34)	51.25 (+6.05 ±0.54)	47.66 (+8.30 ±0.82)	49.58 (+0.00 ±0.00)	51.19 (+5.59 ±0.81)	60.15 (+2.87 ±0.53)	56.62
Flow.	78.40 (+20.11 ±0.08)	98.25 (+0.40 ±0.11)	99.00 (-0.49 ±0.08)	97.38 (+1.07 ±0.12)	98.46 (+0.30 ±0.09)	95.39 (+0.47 ±0.50)	95.65 (+2.89 ±0.26)	99.03 (+0.11 ±0.06)	98.49
Food	93.11 (+0.79 ±0.09)	90.84 (+2.43 ±0.13)	92.23 (+0.88 ±0.19)	93.58 (+0.34 ±0.10)	93.76 (+0.15 ±0.11)	91.02 (+0.00 ±0.00)	93.60 (+0.24 ±0.14)	94.18 (-0.02 ±0.08)	93.33
INet	74.35 (+7.60 ±0.06)	76.08 (+5.31 ±0.13)	80.14 (+1.27 ±0.09)	78.37 (+3.63 ±0.27)	79.70 (+2.41 ±0.18)	76.73 (+3.22 ±0.84)	80.17 (+1.75 ±0.10)	80.38 (+1.96 ±0.12)	81.41
Pet	92.82 (+2.77 ±0.08)	89.98 (+5.46 ±0.42)	93.12 (+2.15 ±0.26)	93.72 (+1.79 ±0.35)	94.14 (+1.42 ±0.31)	93.76 (+0.02 ±0.05)	93.03 (+2.39 ±0.25)	95.86 (+0.04 ±0.16)	95.52
SUN	67.35 (+11.59 ±0.16)	75.37 (+3.28 ±0.16)	78.06 (+0.67 ±0.15)	74.99 (+4.04 ±0.19)	76.27 (+3.06 ±0.23)	73.73 (+0.46 ±0.80)	77.21 (+1.67 ±0.18)	78.85 (+1.49 ±0.15)	78.73
Cars	76.61 (+11.47 ±0.11)	79.73 (+8.53 ±0.34)	87.02 (+1.18 ±0.34)	82.06 (+5.77 ±0.29)	83.34 (+4.47 ±0.34)	82.45 (+0.24 ±0.35)	83.36 (+4.71 ±0.40)	88.35 (+0.73 ±0.12)	88.21
AVG	**74.65** (+10.04 ±1.20)	**81.56** (+4.00 ±0.37)	**85.12** (+0.37 ±0.27)	**82.85** (+2.69 ±0.33)	**83.84** (+2.11 ±0.35)	**80.58** (+0.43 ±0.20)	**80.89** (+3.78 ±0.59)	**86.70** (+0.70 ±0.16)	**84.59**

Table 14: Different model backbones. Average classification accuracy (%) and performance improvements across 12 datasets for different numbers of shots k. Results are reported for two distinct model architectures: **ViT-B/32** and **ViT-H/14**. The values in parentheses denote the absolute performance change after ensembling with DEFTS and the 90% confidence interval. Positive gains are highlighted in green, and negative changes in red. The best baseline accuracy for each k-shot setting (before adding DEFTS) is highlighted in **bold**.

k-shot	ZS+DEFTS	SimpleShot	Linear Probe	Tip-Adapter	APE	TIP-X	TAMP+ZS	LDA+ZS	DEFTS alone
Model Backbone: ViT-B/32 Backbone									
$k = 1$	64.56 (+2.10 ±0.48)	40.45 (+16.26 ±1.23)	42.21 (+15.60 ±1.25)	65.01 (+1.72 ±0.46)	65.78 (+1.07 ±0.42)	65.27 (+1.47 ±0.45)	64.92 (+0.95 ±0.36)	64.66 (+2.01 ±0.47)	**65.08**
$k = 2$	64.56 (+2.12 ±0.58)	52.33 (+16.12 ±1.68)	52.56 (+15.96 ±1.50)	66.74 (+0.44 ±0.24)	67.66 (+0.22 ±0.17)	63.02 (+1.77 ±0.27)	65.73 (+1.09 ±0.43)	67.31 (+0.39 ±0.40)	**67.90**
$k = 4$	64.56 (+5.36 ±0.92)	61.45 (+10.25 ±1.08)	63.77 (+7.37 ±1.10)	68.87 (+1.59 ±0.38)	69.67 (+1.34 ±0.39)	64.72 (+0.54 ±0.15)	68.00 **72.13** (+2.12 ±0.51)	(+0.76 ±0.34)	70.37
$k = 8$	64.56 (+8.47 ±1.09)	67.41 (+6.88 ±0.76)	71.21 (+2.30 ±0.82)	71.87 (+2.27 ±0.32)	73.06 (+1.76 ±0.29)	66.51 (+0.05 ±0.07)	69.32 **76.24** (+3.57 ±0.70)	(+0.47 ±0.13)	72.70
$k = 16$	64.56 (+10.09 ±1.20)	70.89 (+5.02 ±0.59)	76.04 (-0.24 ±0.54)	74.02 (+2.03 ±0.27)	75.34 (+1.35 ±0.23)	68.96 (+0.12 ±0.09)	70.41 **77.27** (+4.22 ±0.73)	(+1.42 ±0.26)	74.38
Model Backbone: ViT-H/14 Backbone									
$k = 1$	81.44 (+1.87 ±0.48)	60.81 (+15.52 ±0.92)	64.27 (+14.11 ±0.89)	81.70 (+1.67 ±0.46)	82.08 (+1.33 ±0.41)	81.83 (+1.55 ±0.44)	82.31 (+0.66 ±0.21)	81.48 (+1.84 ±0.47)	**82.61**
$k = 2$	81.44 (+1.64 ±0.57)	73.52 (+11.48 ±1.13)	73.74 (+11.18 ±1.13)	83.27 (+0.15 ±0.09)	83.82 (+0.11 ±0.15)	80.88 (+0.64 ±0.16)	82.70 (+0.59 ±0.22)	83.12 (+0.47 ±0.21)	**84.65**
$k = 4$	81.44 (+3.78 ±0.83)	81.42 (+5.63 ±0.53)	82.16 (+4.70 ±0.50)	85.01 (+0.81 ±0.28)	85.31 (+0.73 ±0.32)	82.86 (+0.09 ±0.04)	84.42 **86.97** (+1.06 ±0.32)	(+0.12 ±0.16)	85.95
$k = 8$	81.44 (+5.90 ±1.07)	85.39 (+3.00 ±0.31)	86.54 (+1.77 ±0.26)	86.67 (+1.36 ±0.30)	87.30 (+1.02 ±0.26)	84.88 (-0.08 ±0.11)	85.35 **89.23** (+1.94 ±0.57)	(+0.14 ±0.09)	87.39
$k = 16$	81.44 (+6.82 ±1.19)	87.47 (+1.77 ±0.19)	89.17 (+0.12 ±0.16)	87.83 (+1.29 ±0.27)	88.60 (+0.90 ±0.22)	86.89 (+0.46 ±0.16)	85.89 **90.19** (+2.33 ±0.61)	(+0.44 ±0.12)	88.21

First, we define the baseline zero-shot logit vector as $s_{\mathrm{ZS}}(x) \in \mathbb{R}^K$, with its per-class entry computed via cosine similarity:

$$s_{\mathrm{ZS}}(x)_c = \frac{x}{\|x\|} \cdot \frac{t_c}{\|t_c\|},$$

where t_c represents the text embedding for class c.

Tip-Adapter and Variants. Tip-Adapter relies on a key-value cache built from the support set. The hyperparameter β controls the sharpness of the affinity kernel, and α acts as an additive weight on the resulting cache vote. Let $\ell(x_j)$ denote the true label of support image x_j. For a given class c, the total affinity contributed by class-c cache entries is $C(x)_c$:

$$A_j(x) = \exp\big(-\beta(1-\cos(x, x_j))\big), \qquad C(x)_c = \sum_{j:\,\ell(x_j)=c} A_j(x)$$

$$\mathrm{score}(x)_c = s_{\mathrm{ZS}}(x)_c + \alpha\, C(x)_c$$

TIP-X introduces a third parameter γ, weighting a second, independent vote $D(x)_c$. This vote is built from a KL-divergence-based term $\psi_j(x)$ that measures how well cache entry x_j's zero-shot distribution matches x's:

$$D(x)_c = \sum_{j:\,\ell(x_j)=c} \psi_j(x)$$

$$\mathrm{score}(x)_c = s_{\mathrm{ZS}}(x)_c + \alpha\, C(x)_c + \gamma\, D(x)_c$$

APE utilizes the same structural roles for β, α, γ, but computes $A_j(x)$ within a channel-reduced feature space. Here γ weights each cache entry's own reliability $r_j = \exp(\gamma \cdot \mathrm{CE}_j)$, where $\mathrm{CE}_j \geq 0$ is the cross-entropy of entry j's reduced-space zero-shot prediction against its true label.

$$\mathrm{score}(x)_c = s_{\mathrm{ZS}}(x)_c + \alpha \sum_{j:\,\ell(x_j)=c} A_j(x)\, r_j \tag{9}$$

Augmenting with DEFTS. DEFTS's own posterior score $s_{\mathrm{DEFTS}}(x)_c$ as computed in Algorithm C is added as an additional term to each of the three formulas above, with weight w:

Tip-Adapter+DEFTS: $\mathrm{score}(x)_c = s_{\mathrm{ZS}}(x)_c + \alpha\, C(x)_c + w\, s_{\mathrm{DEFTS}}(x)_c$

TIP-X+DEFTS: $\mathrm{score}(x)_c = s_{\mathrm{ZS}}(x)_c + \alpha\, C(x)_c + \gamma\, D(x)_c + w\, s_{\mathrm{DEFTS}}(x)_c$

APE+DEFTS: $\mathrm{score}(x)_c = s_{\mathrm{ZS}}(x)_c + \alpha \sum_{j:\,\ell(x_j)=c} A_j(x)\, r_j + w\, s_{\mathrm{DEFTS}}(x)_c$

LDA Ensembles. The Mahalanobis-based LDA score is defined per class as:

$$s_{\mathrm{LDA}}(x)_c = x^\top \hat{\Sigma}^{-1} \mu_c - \tfrac{1}{2} \mu_c^\top \hat{\Sigma}^{-1} \mu_c, \tag{10}$$

with μ_c representing the class-c support-image mean, and $\hat{\Sigma}^{-1}$ denoting the shrunk pooled precision matrix (Wang et al., 2024). The hyperparameter β controls the weights of the zero-shot score and this LDA score:

LDA+ZS: $\mathrm{score}(x)_c = s_{\mathrm{ZS}}(x)_c + \beta\, s_{\mathrm{LDA}}(x)_c$

When combining with DEFTS, DEFTS's score replaces zero-shot's role in the same additive structure, with w taking over β's share of the zero-shot-vs-DEFTS split:

LDA+ZS+DEFTS: $\mathrm{score}(x)_c = (1-w)\, s_{\mathrm{ZS}}(x)_c + w\, s_{\mathrm{DEFTS}}(x)_c + \alpha\, s_{\mathrm{LDA}}(x)_c$

TAMP Ensembles. For the base TAMP method, λ balances the image-derived mean μ_c and the text prototype t_c (where P is the text-aligned projector). Class c's test-point score is the cosine similarity to this mixed prototype W_c:

$$W_c = \lambda\Big(\frac{\mu_c}{\|\mu_c\|} P\Big) + (1-\lambda)\, t_c$$

$$s_{\mathrm{TAMP}}(x)_c = \Big(\frac{x}{\|x\|} P\Big) \cdot W_c$$

When ensembled with DEFTS, the scores are combined as follows:

$$\textbf{TAMP+ZS:} \quad \text{score}(x)_c = \beta\, s_{\text{ZS}}(x)_c + (1-\beta)\, s_{\text{TAMP}}(x)_c$$

$$\textbf{TAMP+ZS+DEFTS:} \quad \text{score}(x)_c = \beta\, s_{\text{ZS}}(x)_c + (1-\beta)\big[w\, s_{\text{DEFTS}}(x)_c + (1-w)\, s_{\text{TAMP}}(x)_c\big]$$

DEFTS Ensembles (Ours). DEFTS computes its own posterior-based score, $s_{\text{DEFTS}}(x) \in \mathbb{R}^K$, without requiring internal hyperparameter tuning as in Algorithm 1. Ensembled with zero-shot alone:

$$\textbf{DEFTS+ZS:} \quad \text{score}(x)_c = \alpha\, s_{\text{ZS}}(x)_c + (1-\alpha)\, s_{\text{DEFTS}}(x)_c.$$

DEFTS Augmenting Simple Baselines. SimpleShot classifies by nearest class-mean cosine similarity, with no hyperparameters of its own:

$$s_{\text{SS}}(x)_c = \frac{x}{\|x\|} \cdot \frac{\mu_c}{\|\mu_c\|}. \tag{11}$$

Linear Probe fits an ℓ_2-regularized multinomial logistic regression on the k-shot support embeddings and scores x with its decision function, $s_{\text{LP}}(x)_c = \omega_c^\top (x/\|x\|) + b_c$, for fitted weights (ω_c, b_c). Neither method otherwise touches the text embeddings at all – DEFTS is the only source of text information for both once combined:

$$\textbf{SimpleShot+DEFTS:} \quad \text{score}(x)_c = w\, s_{\text{SS}}(x)_c + (1-w)\, s_{\text{DEFTS}}(x)_c.$$

$$\textbf{Linear Probe+DEFTS:} \quad \text{score}(x)_c = w\, s_{\text{LP}}(x)_c + (1-w)\, s_{\text{DEFTS}}(x)_c.$$

G.2 Fixed Weights

When $k = 1$, i.e. each class has only one training sample, we choose fixed hyperparameters for all methods considered.

Natural untuned defaults for ensembling. For ensembling different methods, we default to equal weighting (e.g., $\lambda = 0.5$, $\beta = 0.5$). This equal-weighting strategy serves as the most natural possible baseline without hyperparameter tuning.

Empirical defaults for LDA ensembles. While equal weighting is the natural default for ensembling method, we observed a severe scale mismatch in the LDA formulation of Wang et al. (2024): their raw LDA scores are consistently $\sim 10^3$ times larger in magnitude than the logits in zero-shot CLIP. Consequently, when ensembling LDA with other methods, we down scale the weight of LDA to 0.001.

The exact fixed values used for every hyperparameter across all methods are summarized in Table 15.

Table 15: Fixed hyperparameter values chosen for each method and its ensemble variants.

Method	Parameter(s)	Fixed value
Tip-Adapter	$(\beta,\ \alpha)$	$(1.0,\ 1.0)$
TIP-X	$(\beta,\ \alpha,\ \gamma)$	$(1.0,\ 1.0,\ 1.0)$
APE	$(\beta,\ \alpha,\ \gamma)$	$(1.0,\ 1.0,\ 0.1)$
LDA + ZS	β	0.001
TAMP	λ	0.5
TAMP + ZS	$(\lambda,\ \beta)$	$(0.5,\ 0.5)$
ZS + DEFTS	w	0.5
SimpleShot + DEFTS	w	0.5
Linear Probe + DEFTS	w	0.5
Tip-Adapter + DEFTS	$(\beta,\ \alpha,\ w)$	$(1.0,\ 1.0,\ 1.0)$
TIP-X + DEFTS	$(\beta,\ \alpha,\ \gamma,\ w)$	$(1.0,\ 1.0,\ 1.0,\ 1.0)$
APE + DEFTS	$(\beta,\ \alpha,\ \gamma,\ w)$	$(1.0,\ 1.0,\ 0.1,\ 1.0)$
TAMP + ZS + DEFTS	$(\lambda,\ \beta,\ w)$	$(0.5,\ 2/3,\ 0.5)$
LDA + ZS + DEFTS	$(\alpha,\ w)$	$(0.0005,\ 0.5)$

G.3 Hyperparameter Search Methodology

When $k \geq 2$, we perform a hyperparameter search for each method to identify its optimal configuration. For every method in our comparison, this search is performed via leave-one-out (LOO) cross-validation on the k-shot training set to find the hyperparameter combination that maximizes classification accuracy. Specifically, for each support image, the model is fit without that image and then used to score it. The hyperparameter combination yielding the highest LOO accuracy is selected. In the event of a tie, the configuration that minimizes the cross-entropy loss is chosen.

The search space for each hyperparameter of each method is shown in Table 16.

Table 16: Hyperparameters searched via leave-one-out (LOO) cross-validation for each method and ensemble variant. All grids are defined below the table; "combos" is the total number of joint grid points evaluated for each method.

Method	**Parameter(s)**	**Grid(s)**	**combos**
Tip-Adapter	$(\beta,\ \alpha)$	$\mathcal{G}_\beta^{\mathrm{TIP}} \times \mathcal{G}_\alpha^{\mathrm{TIP}}$	72
TIP-X	$(\beta,\ \alpha,\ \gamma)$	$\mathcal{G}_\beta^{\mathrm{TIP}} \times \mathcal{G}_\alpha^{\mathrm{TIP}} \times \mathcal{G}_\gamma^{\mathrm{TIPX}}$	504
APE	$(\beta,\ \alpha,\ \gamma)$	$\mathcal{G}_\beta^{\mathrm{TIP}} \times \mathcal{G}_\alpha^{\mathrm{TIP}} \times \mathcal{G}_\gamma^{\mathrm{APE}}$	432
LDA + ZS	α	$\mathcal{G}_\alpha$	9
TAMP	λ	$\mathcal{G}_{21}$	21
TAMP+ZS	$(\lambda,\ \beta)$	$\mathcal{G}_{21} \times \mathcal{G}_{21}$	441
ZS+DEFTS	w	$\mathcal{G}_{21}$	21
SimpleShot+DEFTS	w	$\mathcal{G}_{21}$	21
Linear Probe+DEFTS	w	$\mathcal{G}_{21}$	21
Tip-Adapter+DEFTS	$(\beta,\ \alpha,\ w)$	$\mathcal{G}_\beta^{\mathrm{TIP}} \times \mathcal{G}_\alpha^{\mathrm{TIP}} \times \mathcal{G}_w^{\mathrm{DEFTS}}$	576
TIP-X+DEFTS	$(\beta,\ \alpha,\ \gamma,\ w)$	$\mathcal{G}_\beta^{\mathrm{TIP}} \times \mathcal{G}_\alpha^{\mathrm{TIP}} \times \mathcal{G}_\gamma^{\mathrm{TIPX}} \times \mathcal{G}_w^{\mathrm{DEFTS}}$	4032
APE+DEFTS	$(\beta,\ \alpha,\ \gamma,\ w)$	$\mathcal{G}_\beta^{\mathrm{TIP}} \times \mathcal{G}_\alpha^{\mathrm{TIP}} \times \mathcal{G}_\gamma^{\mathrm{APE}} \times \mathcal{G}_w^{\mathrm{DEFTS}}$	3456
TAMP+ZS+DEFTS	$(\lambda,\ \beta,\ w)$	$\mathcal{G}_{21} \times \mathcal{G}_{21} \times \mathcal{G}_{21}$	9261
LDA+ZS+DEFTS	$(\alpha,\ w)$	$\mathcal{G}_\alpha \times \mathcal{G}_{21}$	189

$\mathcal{G}_{21}$: $\{0, 0.05, 0.10, \ldots, 1.0\}$ (21 points).

$\mathcal{G}_\alpha$: $\{0, 0.05, 0.1, 0.35, 0.5, 1.0, 1.5, 2.5, 4.0\} \times 10^{-3}$ (9 points).

$\mathcal{G}_\beta^{\mathrm{TIP}}$: $\{1, 3, 5, 7, 10, 15, 20, 50\}$ (8 points).

$\mathcal{G}_\alpha^{\mathrm{TIP}}$: $\{0.5, 1, 3, 5, 7, 10, 15, 20, 50\}$ (9 points).

$\mathcal{G}_\gamma^{\mathrm{TIPX}}$: $\{0.5, 2, 7, 20, 50, 100, 200\}$ (7 points).

$\mathcal{G}_\gamma^{\mathrm{APE}}$: $\{0, 0.1, 0.35, 0.75, 1.5, 3.0\}$ (6 points).

$\mathcal{G}_w^{\mathrm{DEFTS}}$: $\{0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5\}$ (8 points).