An Empirical Approach to Few-Shot Fine-Tuning

GitHub_Repo

D'Angelo Jacopo (3161719) **Thomopoulos Ioannis (3176145)**

Rienth Maximillian (3325056)

Abstract

When applying Prototypical Networks with ImageNet-pretrained backbones to CAPTCHA classification, we observed an unexpected phenomenon: any form of adaptation, whether adding projection layers or fine-tuning the encoder, consistently degraded performance compared to simply freezing the pretrained weights. To investigate whether this was dataset-specific, we validated our findings on CUB-200-2011, a standard dataset in few-shot learning. The pattern persisted: frozen ResNet18 and ViT-B/16 encoders achieved 91-93% accuracy on 5-way 5-shot classification, while fine-tuned variants dropped to 74-75%. After ruling out implementation errors through comparison with the Baseline++ from Chen et al. 2019 [1], we investigated the underlying mechanism. Through margin distribution analysis, we demonstrate that episodic fine-tuning flattens decision boundaries and destabilizes the embedding space, while frozen backbones maintain tight, confident margins. We attribute this degradation to a fundamental mismatch between supervised pretraining and episodic meta-training objectives. Our findings suggest practitioners should either train episodically from scratch or use frozen pretrained backbones, as mixing these approaches degrades performance.

1 Introduction

Few-shot learning addresses the challenge of classifying novel classes using only a handful of labeled examples per class [6]. This capability is particularly valuable in domains where data collection is expensive or where category definitions evolve rapidly. Traditionally, metric-based frameworks like Prototypical Networks train feature extractors from scratch using episodic meta-learning. However, a compelling alternative we wish to explore is to initialise these models with powerful backbones pre-trained on large-scale datasets (e.g., ImageNet), theoretically combining robust feature representations with the flexibility of few-shot classification.

We initially applied this approach to CAPTCHA classification, where visual categories change frequently and labeled data is limited. During our experiments we observed an unexpected pattern: any form of adaptation, whether adding learnable projection layers or fine-tuning the entire backbone, consistently degraded few-shot classification accuracy compared to simply using the frozen pre-trained encoder. This counterintuitive behavior prompted us to investigate whether the issue was dataset-specific or reflected a more general phenomenon in few-shot learning with strong pre-trained models.

Problem Formulation and Relevance

Our empirical investigation addresses a fundamental question: how does adapting pre-trained vision models affect few-shot classification performance? This question has practical implications as foundation models become increasingly common in computer vision. Understanding when to fine-tune versus freeze pre-trained encoders directly impacts deployment decisions and computational costs. From a theoretical perspective, our findings touch on transfer learning, the stability of learned embedding spaces, and the interplay between different training regimes.

Contributions

Through systematic empirical investigation, we make the following contributions:

- We evaluate four adaptation strategies (fully frozen, frozen with projection layers, and fully fine-tuned) across both CNN (ResNet18) and Vision Transformer (ViT-B/16) backbones on CAPTCHA and CUB-200-2011 datasets.
- We demonstrate that frozen pre-trained backbones consistently outperform adapted variants. Through margin distribution analysis, we show that episodic fine-tuning flattens decision boundaries and destabilizes the embed-

ding space, while frozen backbones maintain tight, confident margins.

- We propose that this degradation stems from a mismatch between supervised pre-training and episodic meta-training objectives. Our findings suggest practitioners should either train episodically from scratch or use frozen pre-trained backbones, as mixing these approaches degrades performance.

Background and Related Work

Prototypical Networks, introduced by Snell et al. [6], provide a framework for few-shot learning by computing class prototypes as the mean of support set embeddings and classifying queries based on distance to these prototypes. Early implementations used CNN backbones trained episodically from scratch. More recently, Vision Transformers have been integrated into this framework [5]. Chen et al. [1] established methodological best practices for episodic training and demonstrated that CNNs trained from scratch achieve strong few-shot performance on standard benchmarks. Building on their protocol, we investigate how different adaptation strategies affect performance when starting from strong pre-trained encoders.

2 Methodology

2.1 Data and Preprocessing

Our experiments use two datasets corresponding to different stages of the project: the Google reCAPTCHA V2 Image Dataset [4] for our initial investigation, and the CUB-200-2011 dataset [7] for our revised addition.

CAPTCHA Dataset:

The Google reCAPTCHA V2 Image Dataset contains 10,390 labelled images across 11 classes. Following our few-shot evaluation protocol, the dataset is split into 5 classes for meta-training, 3 classes for meta-validation and 3 hold-out classes for meta-testing to assess generalisation to unseen classes (as per [1]). Each image has an original resolution of 120×120 pixels and is resized to 224×224 pixels before training. Data augmentation includes random horizontal flipping (p = 0.5), rotations up to ±10°, and mild colour jitter in brightness, contrast, and saturation to increase intra-class variability and reduce overfitting [2]. During evaluation, only resizing and ImageNet-style normalisation are applied.

CUB Dataset:

The CUB-200-2011 dataset consists of 11,788 images across 200 bird species, with fine-grained inter-class differences and higher visual quality. Following established protocols such as [1, 2], we adopt their 100/50/50 class splits for meta-training, meta-validation, and meta-testing, respectively. All images are resized to 224×224 pixels, and the same augmentation and normalisation pipeline as the CAPTCHA dataset is applied to maintain consistency across experiments.

2.2 Few-Shot Learning Framework

Our approach follows the *Prototypical Networks* framework introduced by Snell et al.[6], combined with the episodic training and class-splitting protocol of Chen et al. [1].

Each model consists of an embedding network $f_\phi : R^D \to R^M$ and a prototype-based classifier operating on L2-normalised embeddings.

During meta-training, the model is trained on episodes rather than mini-batches. Each episode samples N_C classes (5-way in all experiments), with $N_S = 5$ support images and N_Q query images per class (15 during training, 5 during validation).

Prototype Computation: For each class k, the prototype is computed as the mean of its support embeddings:

$$\mathbf{c}_k = \frac{1}{N_S} \sum_i f_\phi(x_{k,i}).$$

Query Classification: Queries are classified using Euclidean distances to prototypes:

$$p_\phi(y = k \mid x) = \frac{\exp(-\|f_\phi(x) - \mathbf{c}_k\|_2)}{\sum_{k'} \exp(-\|f_\phi(x) - \mathbf{c}_{k'}\|_2)}.$$

The loss for each episode is the cross-entropy over all query examples. Training uses the Adam optimiser.

2.3 Embedding Network Configurations

For our embedding functions f_ϕ both ResNet18 and ViT-B/16 backbones (pre-trained on ImageNet) are evaluated under four training strategies for a total of eight configurations:

1. **Fully Frozen:** Pre-trained parameters fixed, no fine-tuning.
2. **Fully Fine-tuned:** All parameters updated during episodic training.

3. **Frozen + 1 Layer:** Frozen backbone with added learnable linear projection.
4. **Frozen + 2 Layers:** Frozen backbone with added two-layer MLP (linear-ReLU-linear).

See Appendix A for details.

2.4 Training Procedure

Meta-training, meta-validation and meta-testing follow that of [1, 2]:

All trainable models are trained episodically for up to 100 epochs, with: 500 training episodes and 100 validation episodes per epoch, with early stopping based on validation accuracy.

Final evaluation is conducted on the meta-test classes, disjoint from the training and validation sets. All models are evaluated on the *same* 600 test episodes (across various K-shot settings) to ensure comparability.

3 Results

3.1 Classification Performance

CAPTCHA Dataset Results:

We began our investigation by evaluating the 1-shot, 3-shot, 5-shot, and 10-shot classification performance of our eight model configurations on the CAPTCHA dataset. Table 2 summarises the results.

Unsurprisingly, the Vision Transformer outperforms the CNN across the board. However, a clear and unexpected pattern emerges for both encoder categories. Across all k-shot settings, the Baseline configuration with just the fully frozen encoder achieved the highest accuracy, while any attempt to adapt the backbone deteriorated performance. For example, in the 5-shot setting the CNN backbone accuracy dropped from 78% (frozen) to 45% under full fine-tuning, and the transformer backbone dropped from 90% (frozen) to 45% when fully fine-tuned. Similar trends held for the 1-, 3-, and 10-shot settings and for the other fine-tuning configurations.

We suspected that this surprising behaviour stemmed from the CAPTCHA dataset itself for the following reasons: **1.** It contains classes that were likely already seen by the ImageNet-trained backbones, effectively annulling the few-shot element. **2.** The dataset has only 11 distinct classes. **3.** The classes are very different in terms of domain.

CUB Dataset Results:

To address the issue of the CAPTCHA dataset we repeat our meta-train and meta-test methodologies with the CUB-200-2011 dataset, a standard in the few-shot literature [1, 6, 2]. This dataset has 200 bird classes unseen to ImageNet, thus improving on the CAPTCHA dataset both in terms of class breadth but also domain similarity between classes.

Despite this dataset change we find that fine-tuning still degrades few-shot performance. As summarised in Table 3, the frozen ResNet18 baseline achieves an impressive 5-shot accuracy of 91% on the CUB dataset. However, accuracy drops to 81% with a single linear layer, 77% with two layers, and 74% when fully fine-tuned (accuracy degradation is analogous in the ViT case). This replication confirms that the issue is not CAPTCHA-dataset specific but rather indicates a likely issue with our Baseline ProtoNet implementation. Prompting us to verify it against an established benchmark.

3.2 Baseline Implementation Sanity Check

The benchmark chosen for our implementation validation is the *Baseline++* model introduced by Chen et al. [1]. This model is a standard few-shot classification baseline that trains a ResNet18 backbone from scratch as a feature extractor with a cosine-distance-based classifier on top. To adapt their *Baseline++* model to be comparable to our Baseline, we modified their pipeline by importing the same ResNet18 weights[1] we use. We then run their test implementation with the Cosine Classifier to compare it with our ProtoNet baseline results.

The results revealed a striking similarity: our Prototypical Network implementation achieved a 5-way 5-shot accuracy of 91.13%, while the adapted Baseline++ configuration achieved 91.05%. We attribute this near-identical performance to the mathematical relationship between the two approaches. While Prototypical Networks explicitly calculate the class prototype as the mean of the support set embeddings, the *Baseline++* model learns a weight vector for each class from a few test class examples. Thus, the results being so similar means that the learned weights are empirically similar to the mean embedding. More importantly for our sanity check, this convergence confirms that our Baseline implementation is correct.

[1] ResNet18 Imported Weights: IMAGENET1K_V1

4 Discussion

4.1 Re-evaluating the Baseline Assumption

The sanity check highlighted a critical oversight in our initial experimental premise. It reveals that our starting assumption: *"fine-tuning the baseline is necessary to adapt it to the few-shot task"* was fundamentally flawed.

Our Baseline was not a weak model for few-shot. Its 5-shot accuracy of 91% on CUB substantially exceeds Chen et al.'s Baseline++, which achieves $\sim$62% when trained episodically from scratch on mini-ImageNet and tested on CUB [1, Table 3]. While this difference is partly attributable to our use of full ImageNet pretraining versus episodic training from scratch on a smaller dataset, it demonstrates that the imported backbone already possesses highly robust representations for few-shot classification. The backbone does not require adaptation; it already embeds an effective metric space. Consequently, the "fine-tuning" process was not improving a weak model, but rather distorting a strong one. This leaves the episodic fine-tuning process itself as the sole remaining variable to explain why adapting the model degrades performance compared to the frozen baseline.

4.2 Ruling Out Overfitting

To identify the mechanism behind the performance drop, we first examined whether the models were simply overfitting during episodic training. If this were the case, we would expect to see much higher meta-training accuracy than meta-testing accuracy.

As seen in Table 4, training accuracy is indeed higher than test accuracy across the board. However, the gap between training and testing performance is relatively small, suggesting that overfitting alone cannot explain the steep performance degradation we witness.

4.3 Margin Analysis

To understand why fine-tuning degrades performance, we examine the *classification margins* produced by each model configuration. For each query sample, we define the margin as the difference between the distance to the *nearest incorrect* class prototype and the distance to the *correct* prototype:

$$margin(x) = \min_{k \neq y} \|f_\phi(x) - c_k\| - \|f_\phi(x) - c_y\|$$

where y is the true class label. Positive margins indicate correct classifications, while negative margins indicate errors.

Importantly, the distribution of margins reveals the stability of the embedding space: tight distributions indicate consistent, confident predictions, while flat distributions suggest that the margins are more variable depending on the episode.

Margin Distribution Results:

As shown in Figure 1 and Table 5, the frozen ResNet18 baseline produces a tightly concentrated margin distribution with the majority of the mass falling above zero (mean = 0.097, std = 0.075). This tight distribution indicates that the pre-trained embedding space provides stable class separation: Across episodes the query images consistently receive similar margins regardless of which other classes appear in the episode.

In stark contrast, any form of adaptation dramatically flattens the margin distribution. Adding a single linear projection layer increases the standard deviation to 0.376 (mean = 0.330), while full fine-tuning produces an even flatter distribution (mean = 0.375, std = 0.439). This is visualized in Figure 1, where the fully fine-tuned model's margin histogram is much flatter compared to the frozen baseline's sharp peak.

This flattening indicates instability in the embedding space: Across episodes the query images may be classified with high confidence in one episode but near the decision boundary in another. Suggesting that the finetuning distorts the Backbone's strong geometry. The comparison across all configurations shows that this distortion scales with the number of tunable parameters (see Table 1).

The ViT-B/16 backbone exhibits the same pattern (see Figure 2): the frozen baseline maintains a concentrated distribution (mean = 0.202, std = 0.164), while fine-tuning increases variance substantially (fully tuned: mean = 0.588, std = 0.543). Interestingly, the ViT margin distribution shows a shift toward a bimodal distribution as the number of tuned parameters increases, potentially suggesting that the fine-tuning in the ViT case has a polarizing effect, further indicating a loss of stable embeddings.

4.4 Training Regime Mismatch Hypothesis

Our results reveal an important insight: strong pre-trained encoders do not benefit from episodic fine-tuning, despite Prototypical Networks being effective for few-shot classification when trained episodically *from scratch* [6, 1]. Moreover, Sections 3.2 and 4.2 show that our Fully-Frozen Baseline alone already achieves state-of-the-art few-shot performance.

We hypothesise that performance degradation stems from a fundamental mismatch between training regimes. ResNet18 was pre-trained using standard supervised cross-entropy on ImageNet, while episodic fine-tuning optimises for a different objective: discriminating between the specific classes in each training episode. As shown in Section 4.3, this episode-specific optimisation does not refine the pre-trained geometry toward an ideal ProtoNet solution but rather distorts the already strong baseline. Instead of adapting the backbone to few-shot learning, episodic fine-tuning disrupts the very properties that made it effective in the first place.

5 Conclusions

This work investigated how different adaptation strategies affect few-shot classification performance when using pre-trained backbones within the Prototypical Networks framework. We evaluated ResNet18 and ViT-B/16 encoders under four configurations: 1. Fully frozen, 2. Frozen with one projection layer, 3. Frozen with a Layer-ReLU-Layer MLP and 4. Fully fine-tuned, across both CAPTCHA and CUB datasets.

Our results reveal a consistent and unexpected pattern: strong pre-trained encoders do not benefit from episodic fine-tuning. Any form of adaptation consistently degraded performance.

We hypothesise that this degradation stems from a fundamental mismatch between training regimes, where episodic updates appear to distort the strong pre-existing geometry of the Baseline.

Our findings lead to a clear practical recommendation: for few-shot classification, either train episodically from scratch to learn task-specific representations, or leverage strong pre-trained backbones' robust embedding spaces. Attempting both strategies simultaneously (episodically fine-tuning a non-episodically pre-trained encoder) results in degraded and unpredictable performance.

Limitations

While our Margin Analysis provided a high-level geometric explanation for the performance degradation, our study lacks a deep mechanistic analysis. We did not employ granular, layer-wise techniques, such as feature attribution (e.g. SFAM [3] and attention maps), or embedding topology analysis (t-SNE), to pinpoint exactly where the feature representations collapse during fine-tuning. However, the consistency of degradation across all configurations and both backbones suggests this pattern is robust.

Our investigation is also limited to Prototypical Networks; other meta-learning frameworks (MAML, Matching Networks) may behave differently. Finally, computational constraints prevented us from exploring extended training regimes (500+ epochs). We hypothesise that accuracy may follow a U-shape: starting high (frozen weights), dropping as episodic training distorts the embedding geometry, then potentially recovering as models fully adapt to the few-shot objective.

Future Work

Our findings suggest the degradation stems from a training regime mismatch. Future research should address this mismatch through two complementary approaches:

1. Harmonising the Fine-Tuning Regime: A natural follow-up is to investigate whether *non-episodic* fine-tuning (standard supervised training on the support set) preserves few-shot performance better than episodic adaptation. This would test whether the geometric distortion we observed is specific to the case where training and finetuning strategies are mismatched or inherent to adapting these specific backbones.

2. Alternative Pre-Trained Regimes: Conversely, future work should evaluate backbones pretrained via Self-Supervised Learning rather than standard supervised classification. Since SSL objectives are typically better suited to changing domains, these backbones may yield feature spaces that are naturally more robust to the episodic training structure of Prototypical Networks, potentially resolving the fragility observed with ImageNet-supervised weights.

References

[1] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. *arXiv preprint arXiv:1904.04232*, 2019.

[2] Nathan Hilliard, Lawrence Phillips, Scott Howland, Artëm Yankov, Courtney D. Corley, and Nathan O. Hodas. Few-shot learning with metric-agnostic conditional embeddings. *arXiv preprint arXiv:1802.04376*, 2018.

[3] Yi Liao, Ugochukwu Ejike Akpudo, Jue Zhang, Yongsheng Gao, Jun Zhou, Wenyi Zeng, and Weichuan Zhang. Visual explanation via similar feature activation for metric learning. *arXiv preprint arXiv:2506.01636*, 2025.

[4] Mikhail M. Google recaptcha image dataset. https://www.kaggle.com/datasets/mikhailma/test-dataset, 2025. Kaggle dataset; partially hand-marked images for YOLO.

[5] Abdulvahap Mutlu, Şengül Doğan, and Türker Tuncer. Vit-protonet for few-shot image classification: A multi-benchmark evaluation. *arXiv preprint arXiv:2507.09299*, 2025.

[6] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. *arXiv preprint arXiv:1703.05175*, 2017.

[7] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. Cub-200-2011, 2022.

A Model Configurations

A.1 CNN Backbone (ResNet18)

All CNN models use an ImageNet-pretrained ResNet18 producing 512-dimensional embeddings before projection.

1. **Frozen Baseline:** Entire ResNet18 frozen; output dimension 512.
2. **Frozen + 1 Layer:** Linear projection 512 → 256; output dimension 256; learning rate = 0.001.
3. **Frozen + 2 Layers:** Two-layer MLP (512 → 384 → 256) with ReLU activation; output dimension 256; learning rate = 0.001.
4. **Fully Tuned:** All ResNet layers trainable (BN layers frozen); output dimension 512; learning rate = 0.0001.

A.2 Transformer Backbone (ViT-B/16)

ViT models use an ImageNet-pretrained ViT-B/16, which outputs a 768-dimensional `[CLS]` embedding.

1. **Frozen Baseline:** Entire ViT frozen; output dimension 768.
2. **Frozen + 1 Layer:** Linear projection 768 → 512; output dimension 512; learning rate = 0.001.
3. **Frozen + 2 Layers:** Two-layer MLP (768 → 640 → 512) with ReLU activation; output dimension 512; learning rate = 0.001.
4. **Fully Tuned:** All ViT layers trainable; output dimension 768; learning rate = 0.0001.

A.3 Model Architecture Summary

Table 1: Model Architecture and Trainable Parameters Comparison

Model	Trainable Params	Backbone	Output Dim
CNN			
Frozen Baseline	-	Frozen	512
Frozen + 1 Layer	131K	Frozen	256
Frozen + 2 Layers	296K	Frozen	256
Fully Tuned	11.2M	Trainable	512
Transformer			
Frozen Baseline	-	Frozen	768
Frozen + 1 Layer	394K	Frozen	512
Frozen + 2 Layers	820K	Frozen	512
Fully Tuned	86M	Trainable	768

B Results

B.1 CAPTCHA Dataset Results

Table 2: Model Performance on CAPTCHA Dataset Across Different 5-Way, k-Shot Settings

	Shot			
Model	1	3	5	10
CNN				
Frozen Baseline	0.5519	0.7167	0.7789	0.8449
Frozen + 1 Layer	0.4449	0.5048	0.5320	0.5570
Frozen + 2 Layers	0.4272	0.4780	0.4883	0.5189
Fully Tuned	0.4188	0.4344	0.4462	0.4737
Transformer				
Frozen Baseline	0.7016	0.8604	0.8981	0.9290
Frozen + 1 Layer	0.4724	0.5296	0.5426	0.5702
Frozen + 2 Layers	0.4644	0.5154	0.5181	0.5384
Fully Tuned	0.4078	0.4303	0.4521	0.4683

B.2 CUB Dataset Results

Table 3: Model Performance on CUB Dataset Across Different 5-Way, k-Shot Settings

	Shot			
Model	1	3	5	10
CNN				
Frozen Baseline	0.7472	0.8821	0.9113	0.9312
Frozen + 1 Layer	0.7125	0.7877	0.8140	0.8241
Frozen + 2 Layers	0.6787	0.7410	0.7679	0.7788
Fully Tuned	0.6729	0.7296	0.7421	0.7507
Transformer				
Frozen Baseline	0.7939	0.9079	0.9272	0.9428
Frozen + 1 Layer	0.7535	0.8306	0.8483	0.8565
Frozen + 2 Layers	0.7261	0.7972	0.8107	0.8239
Fully Tuned	0.6819	0.7418	0.7549	0.7687

B.3 Training vs. Test Accuracy on CUB Dataset (5-Way, 5-Shot)

Table 4: Training and Test Accuracy Comparison on CUB Dataset for 5-Way, 5-Shot Setting

	CNN			**Transformer**		
Model	Train	Test	Δ	Train	Test	Δ
Frozen + 1 Layer	0.8187	0.8140	0.0047	0.8728	0.8483	0.0245
Frozen + 2 Layers	0.7952	0.7679	0.0273	0.8728	0.8107	0.0621
Fully Tuned	0.8042	0.7421	0.0621	0.8381	0.7549	0.0832

C Margin Analysis

Table 5: Decision Margin Analysis for 7,500 Query Images Sampled from Test Classes (5-Way 5-Shot on CUB)

Model	Accuracy	Mean Margin	Std Margin
CNN			
Frozen Baseline	0.9113	0.0970	0.0752
Frozen + 1 Layer	0.8140	0.3302	0.3755
Frozen + 2 Layers	0.7679	0.4049	0.4581
Fully Tuned	0.7421	0.3748	0.4389
Transformer			
Frozen Baseline	0.9272	0.2021	0.1642
Frozen + 1 Layer	0.8483	0.4779	0.3921
Frozen + 2 Layers	0.8107	0.6278	0.5018
Fully Tuned	0.7549	0.5880	0.5426

C.1 Decision Margin Distributions

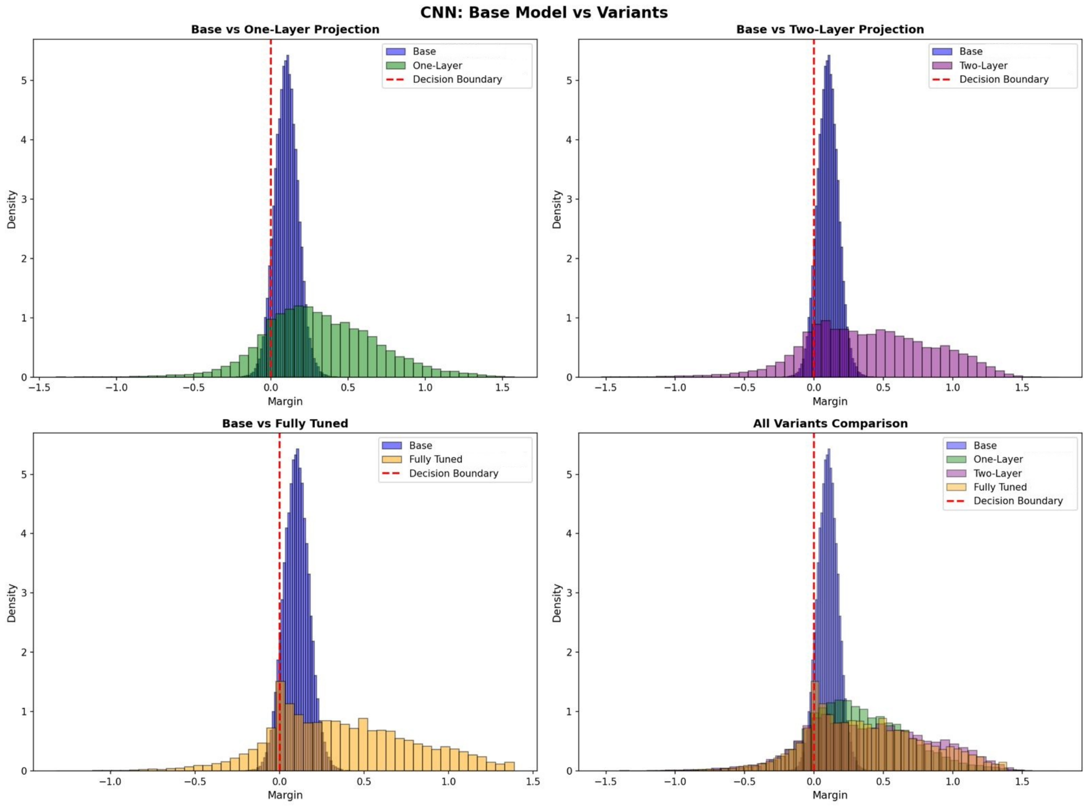

Figure 1: Decision margin distributions for CNN-based models across 7,500 query images from test classes (5-Way 5-Shot on CUB). The frozen baseline shows tight, confident margins, while fine-tuned variants exhibit wider, more variable distributions.

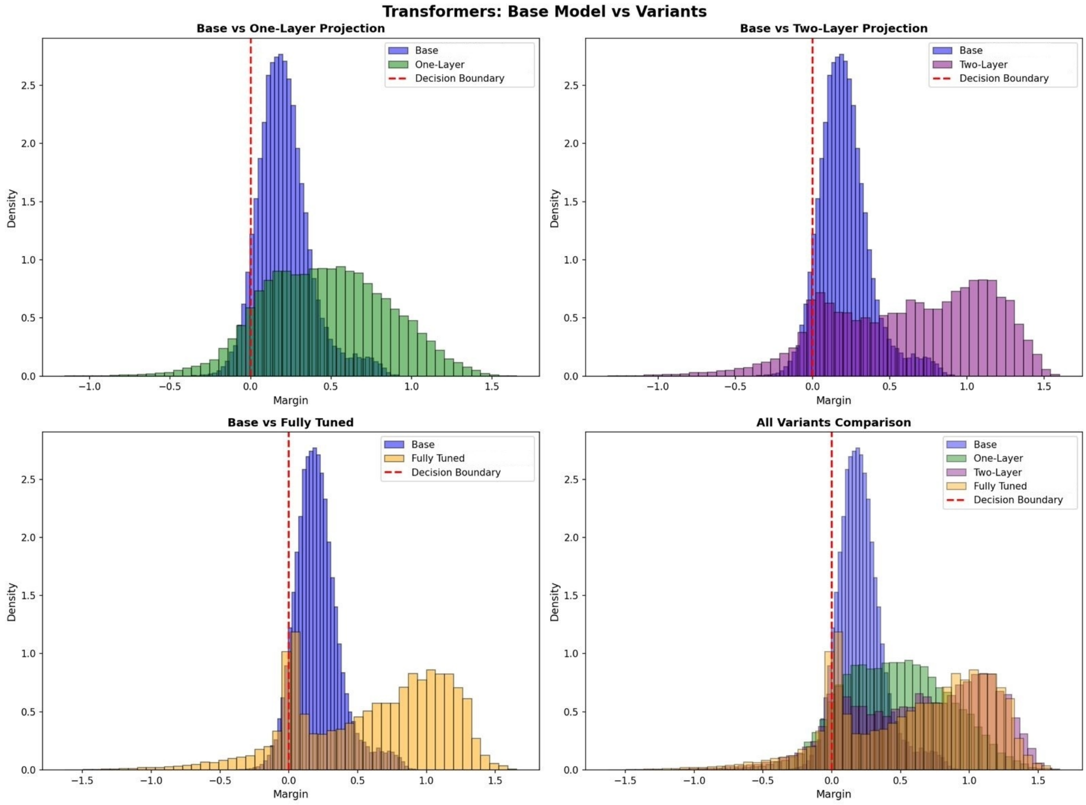

Figure 2: Decision margin distributions for Transformer-based models across 7,500 query images from test classes (5-Way 5-Shot on CUB). Similar to CNN models, the frozen baseline produces more concentrated margins compared to fine-tuned variants.