Thesis

Enhancing Vision-Language Models

Validation-free Adapters and LLM-based Extensions

Marthinus Christoffel Vorster (MSc)

Supervised by Prof. Derek Molloy, Dr Noel Murphy,
Dr Deepak Ajwani (ML-Labs, University College Dublin)



A thesis presented for the degree of Doctor of Philosophy

School of Electronic Engineering
Dublin City University

June 2026

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Doctor of Philosophy, is entirely my own work, and that I have exercised reasonable care to ensure that the work is original and conforms to the regulations on the use and declaration of Generative AI, and does not to the best of my knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed: Marthinus Christoffel Vorster
ID No.: 21267470
Date: 27 June 2026

Use of Generative Artificial Intelligence

In accordance with Dublin City University's Academic Integrity and Research Integrity policies, I acknowledge the controlled and transparent use of Generative Artificial Intelligence (GenAI) tools during the development of this thesis.

- **Grammar assistance:** Grammarly was used for linguistic correction in all published research and in this thesis.
- **Development assistance:** OpenAI's GPT-3.5, GPT-4.1 and GPT-5.1 models were used via Cursor's Integrated Development Environment for:
  (i) Debugging Python code by creating utility scripts that either sequentially log variable states, show additional graphs or metrics to isolate and fix bugs in experiment scripts.
  (ii) Converting descriptions of mathematical formulas (in English and as images) to LaTeX.
  (iii) Producing text prompts, class descriptions, and counterfactual examples forming part of the experimental methodology described in Chapters 4 and 6.
  (iv) Critiquing my work. I frequently asked GenAI models to critique both code and written content, then critically evaluated whether I agreed, and if so, made changes to the content.
- **Literature exploration:** Heuristica was used for building concept maps grounded in research papers to understand new concepts.

All AI-generated artefacts used within the research are explicitly identified in the thesis. No GenAI system was used to compose, structure, or otherwise create any part of the narrative, argumentation, analysis, or conclusions presented herein. All intellectual contributions, interpretations, and results are entirely my own.

Dedication

In memory of my father

"...never stop learning..."

Acknowledgements

I extend my gratitude to Angela Lally and Vicky Flanagan for their administrative support. To the ML-Labs CRT Management for this opportunity, and to Science Foundation Ireland and Research Ireland for their funding. To my supervisors, Prof. Derek Molloy and Dr Noel Murphy: thank you for the research freedom, for the advice to gain a lay of the land first, for your sensibility and flexible support, and for honing my writing style. I acknowledge those who additionally exposed me to academic rigour: Jaco Greyling (hard work), Prof. Leonard Santana (fluency in symbols), and Dr Kevin McGuinness (intricacies to clarity). I am indebted to Mayug Maniparambil for his endless support and for running with me through the burning forest. To my wife, Emri, for making our house a home (two and a half times) and gracefully enduring my preoccupation, dankie, dankie-dankie-dankie. And again, thanks Mayug.

Publications

- **Hold-One-Shot-Out for Validation-Free Few-Shot CLIP Adapters**
  Chris Vorster, Mayug Maniparambil, Noel E. O'Connor, Noel Murphy, Derek Molloy
  Conference: Conference on Computer Vision and Pattern Recognition (CVPR) Findings 2026 — *Accepted*
  Year: 2026

- **Underrepresented in Foundation Model Pretraining Data? A One-Shot Probe**
  Chris Vorster, Mayug Maniparambil, Noel E. O'Connor, Noel Murphy, Derek Molloy
  Conference: IEEE AFRICON — *Accepted*
  Year: 2025

- **Enhancing CLIP with GPT-4: Harnessing Visual Descriptions as Prompts**
  Mayug Maniparambil, Chris Vorster, Derek Molloy, Noel Murphy, Kevin McGuinness, Noel E. O'Connor
  Conference: International Conference on Computer Vision Workshop: What is Next in Multimodal Foundation Models? — *Published*
  Year: 2023

Contents

Glossary xv

1 Introduction 1
- 1.1 Research Questions . . . 2
- 1.2 Thesis Overview and Contributions . . . 3
  - 1.2.1 RQ1: Validation-Free Adaptation . . . 3
  - 1.2.2 RQ2: LLM-Based Global Enhancement . . . 3
  - 1.2.3 RQ3: Fine-Grained Spatial Alignment . . . 4
  - 1.2.4 RQ4: Performance Prediction . . . 5
  - 1.2.5 Summary . . . 5

2 Literature Review 6
- 2.1 Background: Computer Vision Evolution . . . 6
- 2.2 Vision-Language Foundation Models . . . 9
- 2.3 Pre-training Objectives . . . 11
  - 2.3.1 Text Embeddings and Semantic Spaces . . . 13
- 2.4 CLIP: Contrastive Language-Image Pre-training . . . 14
  - 2.4.1 Dense Features from CLIP . . . 15
- 2.5 Improvements to CLIP Training . . . 17
- 2.6 Parameter-Efficient Transfer Learning . . . 18
  - 2.6.1 Prompt Engineering and Prompt Tuning . . . 18
  - 2.6.2 CLIP Adapters . . . 18
  - 2.6.3 Efficient Tuning and Validation-Free Adaptation . . . 19
- 2.7 Low-Shot Learning and Side Information . . . 21
  - 2.7.1 Standard Zero-Shot Learning Setting . . . 21
  - 2.7.2 Multimodal Zero-Shot Learning Setting . . . 22
- 2.8 Fine-Grained Classification and Spatial Alignment . . . 24
  - 2.8.1 Fine-Grained Visual Classification . . . 24
  - 2.8.2 Visual Prompting and Attention Steering . . . 25
- 2.9 Performance Prediction and Out-of-Distribution Detection . . . 26
- 2.10 Chapter Summary . . . 28

2.10.1 Primary Findings . . . 28
2.10.2 Gaps and Research Questions . . . 29

3 HOSO-Adapter: Validation-Free Few-Shot Adaptation 31
3.1 Introduction . . . 31
3.2 Related Work . . . 34
3.3 Preliminaries . . . 35
3.4 SVL-Adapter Reimplementation . . . 36
3.5 PathCLIP Reimplementation . . . 36
3.6 Experimental Setup . . . 37
3.7 Methodology . . . 38
3.8 Results . . . 40
3.9 Discussion . . . 43
3.10 Detailed Results . . . 44
3.11 Ablation Study . . . 45
3.11.1 Per-Dataset Ablation Study . . . 46
3.11.2 Blending Ratio Over Training . . . 47
3.11.3 Overfitting Analysis . . . 49
3.11.4 Residual Grid Search . . . 49
3.11.5 Ablation Design Choices . . . 51
3.12 Additional Investigations . . . 54
3.12.1 Class-Level Perspective . . . 54
3.12.2 Test Augmentations . . . 58
3.13 Practical Interpretation of Ratio Selection . . . 58
3.14 Chapter Summary . . . 59

4 LLM-Generated Side Information for Low-Shot Learning 60
4.1 Introduction . . . 60
4.2 Background . . . 62
4.2.1 The Low-Shot Learning Problem . . . 62
4.2.2 Side Information for Relating Seen and Unseen Classes . . . 63
4.2.3 Semantic Prototypes for Classification . . . 63
4.2.4 Key Architectures . . . 63
4.2.5 LLMs as Side Information Generators . . . 64
4.3 Methodology . . . 64
4.3.1 Pipeline and Reporting Conventions . . . 64
4.3.2 Embeddings . . . 65
4.3.3 Semantic Contrastive Architecture . . . 68
4.3.4 Detailed Prompt Engineering and Data Generation . . . 70
4.3.5 Attribute Prototype Network Modification . . . 74

4.3.6 Prompting for Attribute Verification . . . 74
4.3.7 CADA-VAE System Prompts . . . 79
4.3.8 Quantitative Comparison of Embedding Families . . . 79
4.3.9 Graph Structure and Neighbourhood Overlap . . . 81
4.4 Truthfulness of Text Embeddings as Semantic Graphs . . . 84
4.4.1 Experimental Protocol . . . 84
4.4.2 Results . . . 85
4.4.3 Implementation Details . . . 86
4.5 Few- and Zero-Shot Recognition on MiniImageNet . . . 86
4.5.1 Prompt Design for CLIP-Based MiniImageNet Prototypes . . . 87
4.5.2 Main MiniImageNet Results . . . 88
4.5.3 GloVe-Based Semantic Priors . . . 89
4.5.4 Implementation Details . . . 89
4.6 CUB: Generative ZSL with CADA-VAE-Style Embedding Transfer . . . 90
4.6.1 Side Information and Baselines . . . 90
4.6.2 Results . . . 91
4.6.3 Implementation Details . . . 92
4.7 CUB: Attribute Prediction Network with LLM-Derived Attributes . . . 92
4.7.1 Side Information Variants . . . 92
4.7.2 Results . . . 93
4.7.3 Implementation Details . . . 94
4.8 Enhancing CLIP with GPT-4 and Visual Descriptions . . . 94
4.8.1 Review of CLIP and CLIP-Adapter . . . 95
4.8.2 Simple Few-Shot Adapters for Visual Sentences . . . 96
4.8.3 Experiments . . . 97
4.8.4 GPT-Generated Visual Sentences Improve Zero-Shot Transfer 98
4.8.5 GPT-Adapters Improve Few-Shot Transfer Performance . . . 99
4.8.6 Ablation Over Different GPT Models . . . 99
4.9 Discussion and Summary . . . 101
4.9.1 What Determines Whether Side Information Transfers . . . 101

5 Spatial-Semantic Alignment for Fine-Grained Classification 103
5.1 Introduction . . . 103
5.2 Methodology . . . 104
5.2.1 Training-Free Interventions . . . 104
5.2.2 Trainable Spatial Architecture . . . 107
5.2.3 Visual Stream with Spatial Tokens . . . 108
5.2.4 Adapter Architectures . . . 110
5.2.5 Spatial Alignment Objectives . . . 110

5.3 Experiments and Results . . . 111
5.3.1 Training-Free Intervention Results . . . 112
5.3.2 Visual Prompting and Text Refinement . . . 112
5.3.3 Inference-Time CLIP Surgery . . . 112
5.3.4 Trainable Spatial Alignment Results . . . 113
5.3.5 Ablation Studies . . . 115
5.4 Discussion . . . 116
5.4.1 Limited Separation Between Text Embeddings . . . 116
5.4.2 Diffuse Patch–Token Similarity . . . 116
5.4.3 Overfitting to Seen Part–Text Pairs . . . 118
5.4.4 Oracle Analysis . . . 119
5.5 Interpretation of the Spatial Results . . . 120
5.6 Chapter Summary . . . 121

6 Predicting Vision-Language Model Performance Using LLM-Generated Counterfactuals 122
6.1 Introduction . . . 122
6.2 Methodology . . . 123
6.3 Experiments and Results . . . 126
6.4 Ablation and Discussion . . . 127
6.5 Operational Use of the Estimate . . . 129
6.6 Chapter Summary . . . 130

7 Conclusion 131
7.1 Summary of Contributions and Findings . . . 131
7.1.1 RQ1: Validation-Free Adaptation . . . 131
7.1.2 RQ2: LLM-Based Global Enhancement . . . 132
7.1.3 RQ3: Fine-Grained Spatial Alignment . . . 133
7.1.4 RQ4: Performance Prediction . . . 133
7.2 Discussion and Speculation . . . 134
7.2.1 Role of the Pre-Trained CLIP Prior . . . 134
7.2.2 Limited Fine-Grained Separation in the Text Encoder . . . 134
7.2.3 The Role of Counterfactuals in Robustness . . . 135
7.2.4 Limitations . . . 135
7.2.5 Future Outlook . . . 136
7.3 Concluding Remarks . . . 137

List of Figures

2.1 Overview of CLIP architecture . . . 7
2.2 FGVC-Aircraft example variants . . . 22
2.3 CADA-VAE architecture . . . 23
2.4 Attribute Prototype Network architecture . . . 24
2.5 CUB part location markers . . . 25
2.6 CUB part location and attribute annotation interfaces . . . 26

3.1 Single-example CLIP probe versus full test-set accuracy . . . 32
3.2 HOSO-Adapter method overview . . . 33
3.3 Effect of the blending ratio on performance . . . 34
3.4 HOSO-Adapter performance across few-shot settings . . . 42
3.5 Overfitting comparison for HOSO-Adapter . . . 43
3.6 Learnt blending ratio traces for selected datasets . . . 44
3.7 Blending ratio trends across datasets . . . 48
3.8 Overfitting comparison across datasets . . . 49
3.9 Hold-out cache-size ablation . . . 53
3.10 EuroSAT per-class accuracy by blending ratio . . . 55
3.11 EuroSAT per-class accuracy across blending ratios . . . 57

4.1 Semantic Contrastive architecture . . . 68
4.2 t-SNE visualisation of image embeddings and prototypes . . . 69
4.3 Cosine-similarity distributions . . . 81
4.4 Thresholded graph degree distributions . . . 83
4.5 Top-5 neighbourhood overlap . . . 83

5.1 Visual prompting examples . . . 106
5.2 Fine-grained CUB classification example . . . 107
5.3 Packing mechanism visualisation . . . 109
5.4 Dense similarity map for standard CLIP . . . 117
5.5 Dense similarity map for CLIP Surgery . . . 118
5.6 Natural-image CLIP and CLIP Surgery similarity maps . . . 119

6.1 PreLabellingProbe method overview 124
6.2 Ground-truth versus predicted zero-shot accuracy 128

List of Tables

3.1 Dataset prompts used for evaluation . . . 38
3.2 Transfer learning performance . . . 41
3.3 ImageNet RN50 accuracy . . . 44
3.4 RN50 main results . . . 45
3.5 ViT-B/16 main results . . . 45
3.6 ViT-B/16 16-shot performance comparison . . . 46
3.7 RN50 16-shot ablation summary . . . 46
3.8 HOSO-Adapter design-choice ablation . . . 47
3.9 CLIP-Adapter ViT-B/16 blending-ratio search . . . 50
3.10 CLIP-Adapter RN50 blending-ratio search . . . 50
3.11 ViT-B/16 blending-ratio strategy comparison . . . 51
3.12 Feature versus logit blending for ViT-B/16 . . . 51
3.13 Blending-ratio learning-rate ablation . . . 52
3.14 HOSO-Adapter with data augmentation . . . 58

4.1 Chapter 4 evaluation pipelines . . . 65
4.2 Pairwise cosine similarity summary . . . 80
4.3 Thresholded semantic graph statistics . . . 82
4.4 Graph-based truthfulness benchmark . . . 85
4.5 MiniImageNet prompt results . . . 88
4.6 MiniImageNet GloVe-prototype results . . . 89
4.7 CUB CADA-VAE zero-shot results . . . 91
4.8 CUB APN zero-shot results . . . 93
4.9 VDT zero-shot top-1 accuracy . . . 98
4.10 ViT-B/16 zero-shot transfer on 12 datasets . . . 98
4.11 Base-to-new generalisation comparison . . . 99
4.12 GPT model comparison for VDT side information . . . 100
4.13 Visual and non-visual prompt ensembles . . . 101

5.1 Visual prompting and textual intervention results . . . 112
5.2 CLIP Surgery spatial–semantic alignment results . . . 113

5.3 Spatial loss weighting results 114
5.4 Global versus spatial alignment objectives 114
5.5 Pairwise cosine similarity of CLIP text embeddings 116
5.6 Overfitting to seen part–text pairs 119
5.7 Top-5 re-scoring results 120

6.1 Dataset overview 126
6.2 CLIP accuracy versus PreLabellingProbe 127
6.3 Test performance comparison 129

Glossary

Aligned Embedding Space Representation space in which semantically related images and texts lie close to one another.

Blending Ratio Hyperparameter that defines the linear combination weights between the features of a pre-trained model and an adapter.

CLIP Contrastive Language-Image Pre-training

CLIP-Adapter Method that adds small, learnable adapter layers to a pre-trained Contrastive Language-Image Pre-training model to improve adaptation to new tasks without full retraining.

Continuous Representation Encoding of information as real-valued vectors rather than discrete symbols.

Contrastive Loss Loss function that encourages similar pairs of inputs to have nearby representations and dissimilar pairs to be far apart.

Cosine Similarity Similarity measure that computes the normalised dot product between two vectors.

Downstream Task Tasks on which a pre-trained model is adapted or evaluated after the initial *pre-training* phase.

Embedding Space Vector space in which a model embeds data points so that geometry reflects semantic relationships.

Few-Shot Learning setting where a model adapts to new classes using only a small number of labelled examples per class.

Fine-Grain Setting where the task is to distinguish highly similar subcategories within a broader category.

Fine-Tuning Further training of a pre-trained model on a smaller, task-specific dataset.

Masked Language Modelling Training objective where some tokens in text are masked and the model predicts the masked tokens from context.

Object Detection Computer vision task that detects and localises objects in an image.

Pre-training Training a model on a large, generic dataset before adapting it to a specific task.

Pretrained Weight Model parameters obtained from pre-training on a large dataset.

Prompt Engineering Designing and refining input prompts to steer model behaviour towards desired outputs.

Self-Supervised Learning Learning paradigm where the model derives training signals from the data itself rather than manual labels.

Self-Training Semi-supervised procedure that first trains on labelled data and then retrains on unlabelled data using model predictions as labels.

Side Information Auxiliary class-level knowledge beyond raw labels, such as attribute vectors, textual descriptions, or taxonomic structure, for relating seen and unseen classes in low-shot learning.

Supervised Contrastive Learning Contrastive learning approach that brings together representations of samples from the same class and pushes apart different classes.

Task-Level zero-shot Inference Ability of a model to perform a task that differs from the one used during training, without task-specific supervision.

Textual Embedding Vector representation of text that captures semantic relationships between words or phrases.

Transformer Neural network architecture based on self-attention mechanisms that weigh the importance of different parts of the input.

Vision-Language Foundation Model Model that learns joint representations over visual and language inputs and supports many tasks.

Zero-Shot Ability of a model to predict labels for categories that do not appear in the training set.

Enhancing Vision-Language Models

Validation-free Adapters and LLM-based Extensions

Marthinus Christoffel Vorster

Abstract: Vision-language models such as Contrastive Language-Image Pre-training (CLIP) learn joint representations of images and text from web-scale data, enabling zero-shot image classification. However, adapting these models to new domains with limited labelled data presents three challenges: existing adaptation methods are rarely strictly few-shot, simple text prompts fail to capture the fine-grained semantics, and current approaches discard spatial information from image encoders.

First, we introduce the Hold-One-Shot-Out Adapter, a validation-free approach that reserves one example per class to tune the blending of prior and adapted features. This method improves on the CLIP-Adapter baseline by four percentage points on average across eleven datasets.

Second, we investigate whether Large Language Models can generate *side information* that improves low-shot transfer. A systematic comparison reveals that visual descriptions reflect visual similarity better than dictionary definitions or taxonomic hierarchies. Our method improves low-shot accuracy by two percentage points on average across twelve datasets.

Third, we examine whether aligning spatial image features with part-level text descriptions improves fine-grained image recognition. Experiments reveal that training-free visual prompting degrades accuracy, and trainable spatial adapters overfit to limited training data. Our analysis shows that distinct part descriptions can have only small angular separation in the CLIP text space, limiting the supervision signal for local alignment.

Fourth, we present PreLabellingProbe, a method that estimates dataset-level zero-shot performance using one labelled image per class. A Large Language Model is used to generate plausible and counterfactual captions, and the margin between similarity scores is used to estimate dataset-level accuracy with a Pearson correlation of 0.96.

The thesis improves CLIP adaptation when labelled data are scarce and shows that current text encoders limit fine-grained transfer.

Chapter 1

Introduction

Recent work in Computer Vision has shifted from training separate models for each task to adapting large pre-trained foundation models. Contrastive Language-Image Pre-training (CLIP) [1] is a central example. CLIP is a vision–language model that learns a shared embedding for images and text from large-scale, weakly aligned, image–text pairs collected from the internet. A practitioner can describe each class with a short text prompt, and then classify an image by finding the text embedding that lies closest to the image embedding. This zero-shot classification setting reduces the need for large labelled datasets. However, it raises new questions about adaptation and evaluation in new domains.

In many applications, a user needs to adapt CLIP to a specific dataset with only a few labelled images per class. Few-shot adaptation freezes the pre-trained backbone (model) and learns a small number of extra parameters, often called an adapter. Standard fine-tuning practice relies on a separate validation set to tune hyperparameters such as the learning rate or the adapter strength. In strict few-shot regimes, such a validation set does not exist. At the same time, common approaches ignore the semantic information that modern Large Language Models (LLMs) can provide about visual concepts, and offer limited support for fine-grained spatial reasoning. Practitioners also lack simple tools to estimate how well a vision-language model will perform in a new domain before investing in full annotation.

Low-data constraints appear in application domains where labels are expensive to acquire. Examples include medical imaging, satellite image analysis, biodiversity monitoring, industrial inspection, and fine-grained product or species recognition. In these settings, a practitioner may have only a few labelled examples for each new class to train or adapt a model.

This thesis, titled **Enhancing Vision-Language Models: Validation-free Adapters and LLM-based Extensions** focuses on adapting and evaluating CLIP when labelled data are scarce. Throughout this thesis, terms such as scarce and few-shot refer to settings with small labelled support sets, typically 1, 2, 4, 8 or

16 labelled images per class. The thesis is structured around four related research questions. The first question asks how an adapter can respect the strict few-shot protocol. The second question asks how LLM-generated descriptions can serve as *side information* at the level of global image embeddings. The third question asks whether similar ideas hold at the local image-patch level. The fourth question asks how CLIP [1] performance on a new dataset can be estimated from one labelled image per class using LLMs.

The four technical contributions address vision-language model adaptation and evaluation. First, HOSO-Adapter enables validation-free few-shot CLIP [1] adaptation by using a one-shot hold-out cache as a dynamic regulariser; the method improves over strict baselines by about four percentage points on average across eleven datasets and surpasses oracle test-set-tuned baselines in some settings. Second, CLIP-GPT and VDT-Adapter use LLM-generated visual descriptions to improve zero-shot transfer and few-shot Base-to-New generalisation; CLIP-GPT improves zero-shot accuracy by about two percentage points on average, and VDT-Adapter improves unseen-class accuracy by about three percentage points relative to CoCoOp [2]. Third, the spatial-alignment experiments measure the separation between fine-grained descriptions in the CLIP text space: distinct part-level descriptions can have cosine similarity above 0.95, limiting fine-grained patch-level transfer. Fourth, PreLabellingProbe estimates full test-set zero-shot accuracy from one labelled image per class by using the margins between LLM-generated plausible and counterfactual captions, achieving a Pearson correlation of 0.96 across the evaluated datasets.

1.1 Research Questions

The research addresses four primary questions, which correspond to the core technical chapters. Chapters 3–6 address these questions in turn:

1. **Validation-free adaptation:** How can CLIP be adapted in a strict few-shot setting without relying on a separate validation set?

2. **LLM-based global enhancement:** How can Large Language Models be used to generate side information for CLIP that enhances zero-shot and few-shot transfer?

3. **Fine-grained spatial alignment:** To what extent can explicit alignment between spatial visual features and LLM-generated, fine-grained, part descriptions improve few-shot generalisation of image classification?

4. **Performance prediction:** Can CLIP's zero-shot accuracy on a full test set

be estimated from one labelled image per class and LLM-generated counterfactuals?

1.2 Thesis Overview and Contributions

The overview links the four research questions to the technical chapters and summarises the main contributions. Chapter 2 contains background on foundation models, CLIP [1], adapter methods and side information in low-shot learning. Chapters 3 to 6 contain the method definitions, experimental protocols and empirical results for CLIP adaptation and evaluation. The relation between the findings, the research questions and wider vision-language model research appears in Chapter 7.

1.2.1 RQ1: Validation-Free Adaptation

The Chapter 3 experiments examine common practice in few-shot adaptation and show that many methods rely on validation sets or implicit tuning on the test set. These practices violate the strict few-shot protocol, which requires only a small support set per class. HOSO-Adapter addresses this issue by reserving a small hold-out cache inside the support set.

The HOSO-Adapter reserves one example per class from the support set, and uses this one-shot cache to tune the *blending ratio* between CLIP and the adapter. The method learns adapter parameters on the remaining support examples while it monitors performance on the held-out cache. When performance on the hold-out examples begins to degrade, the technique increases the contribution of the CLIP prior. This behaviour acts as a dynamic regulariser, limiting the adapter's overfitting.

Experiments in Chapter 3 show that the HOSO-Adapter achieves strong accuracy in the validation-free setting and improves over the baseline by about 4 percentage points on average across 11 datasets (see Chapter 3 for details). In eight-shot and sixteen-shot settings, the HOSO-Adapter also improves on an oracle baseline that tunes the blending ratio on the test set. In these settings, the learned, data-dependent blending ratio outperforms the static ratio chosen by grid search on the test set.

1.2.2 RQ2: LLM-Based Global Enhancement

In Chapter 4 we address the limitation of using class names as text prompts. The experiments test whether longer natural-language descriptions generated by LLMs yield text embeddings that align more closely with visual structure. The analysis

first compares the geometry of text embeddings across different description sources, and then evaluates two methods that use LLM outputs as side information.

The analysis in Chapter 4 shows that prompts based on LLM-generated visual descriptions yield semantic graphs that better match the structure of visual similarity than dictionary definitions or WordNet hierarchies. In the zero-shot setting, the **CLIP-GPT** ensemble aggregates multiple LLM-generated descriptions for each class. This ensemble improves accuracy over default CLIP prompts by about two percentage points on average, with gains of over five percentage points on fine-grained datasets such as CUB, DTD, and EuroSAT.

The few-shot setting uses the **Visually Descriptive Text Adapter (VDT-Adapter)**. The VDT-Adapter uses a self-attention mechanism to select relevant descriptions and combine them into a single text representation. Experiments show that VDT-Adapter improves Base-to-New generalisation and exceeds the CoCoOp [2] baseline by roughly three percentage points on unseen classes (see Chapter 4). The three-point gain is measured on unseen classes after training on the base classes.

1.2.3 RQ3: Fine-Grained Spatial Alignment

The Chapter 5 experiments test whether the embedding space of CLIP supports reliable alignment between local image patches and detailed attribute descriptions, such as different beak shapes or plumage patterns. The goal is to determine whether stronger spatial alignment improves fine-grained few-shot classification.

The analysis first evaluates simple, training-free (zero-shot) spatial interventions such as adding red boxes or blurring parts of the image. These modifications do not improve performance and often degrade accuracy on fine-grained tasks. They alter the input relative to the natural images used in CLIP pre-training, but the experiments do not isolate why accuracy decreases.

A subsequent few-shot experiment trains a spatial adapter to match patches to text descriptions. The adapter achieves about 75% accuracy on the training set but only about 28% on the test set. The train–test gap is consistent with overfitting to the training correspondences. Further analysis finds limited separation between CLIP text embeddings. Many distinct fine-grained descriptions, such as “spotted breast” and “striped breast”, have cosine similarities above 0.95. The small margins between attributes limit the available supervision for discriminative local features. Current CLIP embeddings support global alignment but do not provide sufficiently separated part descriptions for fine-grained spatial transfer in the few-shot setting.

1.2.4 RQ4: Performance Prediction

The Chapter 6 evaluation addresses performance estimation rather than adaptation. The setting assumes that a practitioner needs to select a model for a new dataset, but cannot afford to label a large test set. PreLabellingProbe uses LLM-generated counterfactuals to estimate the zero-shot accuracy of CLIP.

PreLabellingProbe uses an LLM to generate a caption and a set of hard-negative counterfactual captions for a single labelled image per class. The method then measures the margin between the similarity score that CLIP assigns to the true caption and the scores assigned to the counterfactual captions. This margin serves as a predictor of dataset-level performance.

Experiments in Chapter 6 show that a simple linear regressor trained on the aforementioned margins estimates full test-set zero-shot accuracy with a Pearson correlation of 0.96 across a diverse set of datasets. The result indicates that a practitioner can estimate the utility of CLIP in a new domain by using only one labelled example per class, along with a small number of LLM generations.

1.2.5 Summary

This thesis presents a validation-free adapter for the strict few-shot setting, introduces methods that leverage LLM-generated descriptions as side information at the global and local image levels, and proposes a performance prediction tool for data-scarce evaluation of vision-language models. Detailed experimental results and implications for future vision-language model adapters appear in the following chapters.

Chapter 2

Literature Review

The literature review follows the developmental chronology of Computer Vision from hand-crafted features and CNNs, through transfer learning and self-supervised representation learning, to CLIP. CLIP demonstrated that large-scale image-text pre-training could support zero-shot recognition, and work from 2021 to 2024 expanded this direction through prompt learning, adapter methods, dense CLIP features, and LLM-generated side information. This chronology situates the thesis contributions: validation-free adaptation addresses the practical few-shot use of CLIP, while the later chapters examine whether the recent availability of LLM-generated text can improve global and spatial recognition.

Figure 2.1 summarises the central CLIP mechanism that recurs throughout this chapter. Contrastive pre-training aligns image and text encoders, after which zero-shot classification compares image embeddings with text embeddings for candidate classes.

2.1 Background: Computer Vision Evolution

This section reviews the main stages in Computer Vision that lead to current vision-language models and introduces the key architectures, training strategies, and representation-learning ideas on which later chapters build.

Early work in Computer Vision relied on hand-crafted features and simple classifiers. Methods used filters or kernel functions to detect edges, corners, or texture patterns, and then fed these features to traditional machine learning algorithms for prediction. These pipelines worked well for simple scenes, but they depended on expert design and did not scale well to complex data.

CNNs replaced hand-designed pipelines by learning features directly from pixels. Increased computing power and larger datasets made it possible to train these networks at scale. A major step was AlexNet [3], a CNN with five convolutional layers and three fully connected layers, trained on the ImageNet benchmark. ImageNet

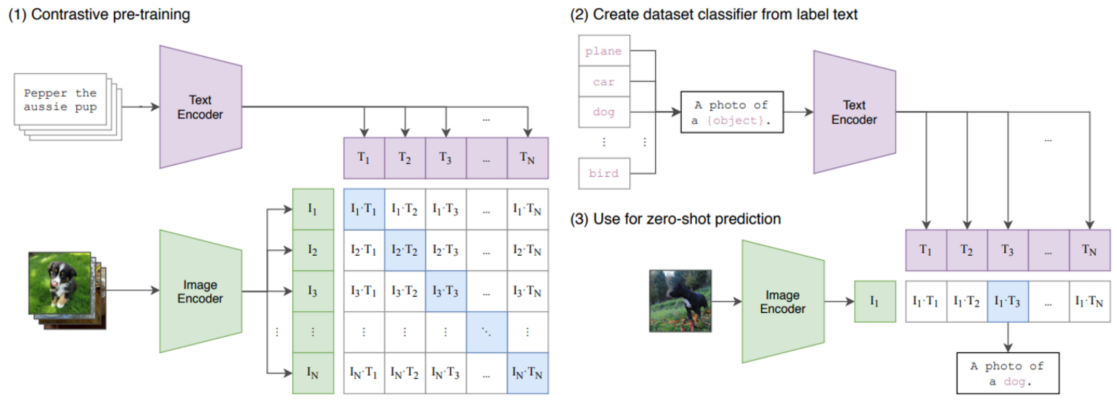


Figure 2.1: Overview of CLIP. The image encoder output corresponds to the normalised image embedding z^I, and the text encoder output corresponds to the normalised text embedding z^T. These are the embeddings used in the symmetric InfoNCE loss in Equation 2.1, with image-to-text and text-to-image components $\mathcal{L}_{I\to T}$ and $\mathcal{L}_{T\to I}$ defined in Equations 2.2 and 2.3. At inference time, each label is converted to a textual description and passed through the text encoder. The zero-shot prediction is the label whose textual embedding has the highest cosine similarity to the image embedding. Adapted from [1].

contains more than 1 million images across 1,000 classes. The deep architecture of AlexNet, together with rectified linear unit (ReLU) activations and overlapping pooling, resulted in a significant drop in error on ImageNet compared to the previous year. This result shows that end-to-end feature learning can outperform hand-crafted pipelines.

After AlexNet, researchers increased network depth and width. The authors of ResNet [4] introduced residual connections (also known as skip connections). These connections allow information and gradients to flow across many layers. Instead of forcing each stack of layers to learn a full mapping, ResNet learns the residual between input and output. This design reduces the difficulty of optimisation and supports deep models, *i.e.* model architectures with many layers.

As models grow, pre-training becomes an effective way to reuse computation and data. In this setting, a large model is trained on a surrogate task and then adapts to a *downstream task* via *fine-tuning*. Pre-training objectives often aim to improve the quality of internal representations rather than directly solve the final task. For example, colourisation tasks ask a model to recover colour from a grey-scale image, and rotation prediction tasks ask a model to infer the rotation applied by a data pipeline. These surrogate tasks encourage the model to encode structural information about objects, lighting, and geometry, which later helps with other tasks.

Although many early benchmarks focused on image classification, the field quickly extended to richer tasks such as *object detection* and image segmentation. Object detection requires a model to locate and classify each object instance in an image.

Image segmentation goes further by assigning a class label to each pixel. Early systems were usually designed with separate architectures and loss functions for each task. More recent work reuses a standard encoder and adds a lightweight trainable classifier. Section 2.2 describes how this trend extends to vision-language systems.

The shift from task-specific architectures to more general models places representation learning at the centre of Computer Vision research. An image representation is useful when a simple classifier can use it to solve many different tasks with minimal fine-tuning. Deeper networks that share parameters across tasks support this goal by learning features that capture object shape, texture, and context in a compact form.

Modern deep networks produce representations, such as those just mentioned, in a vector space often referred to as an *embedding space*. The final layer of a vision encoder maps each input image to a real-valued vector. During training, the objective encourages images from the same class to lie close together, and images from different classes to lie far apart. Distances in this embedding space approximate semantic similarity. This property allows a single encoder to support many tasks through different classifiers or similarity measures.

Two changes drove this transition: larger datasets and GPUs capable of training deeper networks. Representations compress the information in extensive image collections into dense vectors. As a result, models can generalise better, and the need for complex surrogate objectives reduces when enough labelled data are available.

At the same time, researchers began to explore learning without manual labels. *Self-supervised learning* in Computer Vision uses the structure within the data itself to define training signals [5]. These methods aim to learn representations that transfer across tasks. They help when labelled data are scarce or expensive, and they also improve robustness.

A key mechanism in self-supervised learning is *contrastive loss*. Contrastive methods construct positive pairs that should be close in the embedding space and negative pairs that should be further apart. In SimCLR [5], a positive pair consists of two augmentations of the same image, for example, two random crops with different colour jitter. A batch of images provides negative examples, since each augmented image should be closer to its partner than to augmentations of other images. This simple setup yields strong visual representations without labels.

Related supervised methods apply contrastive ideas to labelled data. Triplet loss [6] uses an anchor image, a positive image from the same class, and a negative image from a different class, and it encourages the anchor to move closer to the positive than to the negative. *Supervised Contrastive Learning* (SupCon) [7] extends this idea by treating all images from the same class in a batch as positives for one another.

Contrastive learning also enables vision-language models. By training on exten-

sive collections of weakly aligned image-text pairs, models can learn a joint embedding space. The *Contrastive Language-Image Pre-training (CLIP)* model [1] follows this strategy. It learns a vision encoder and a text encoder that map images and text into a shared embedding space. Training uses a contrastive objective on 400 million image-text pairs, yielding representations that support *zero-shot* recognition.

Zero-shot learning and *few-shot* learning describe how a model uses prior knowledge when labelled data for a new task are limited. In few-shot learning, the model adapts to a new set of classes using a small labelled dataset, often with fewer than 20 examples per class. The meaning of "zero-shot" depends on which training stage supplies the reference point. In conventional class-level ZSL, an unseen class is absent from the labelled classes used to train the downstream recognition model. In CLIP evaluation, zero-shot normally means that no labelled examples from the target dataset are used for task-specific training, adaptation, or prompt optimisation. It does not mean that the underlying concept was absent from CLIP's web-scale image–text pre-training data. A class name, a synonym, or images depicting the concept may have occurred during pre-training even though no target-dataset example was used. Since the web corpus does not provide an exhaustive class inventory, absence from it generally cannot be established.

This thesis therefore uses two explicit conventions. In the CADA-VAE and APN experiments, *unseen* refers to classes withheld from the task-level training split. In CLIP evaluations, *zero-shot* refers to inference without target-dataset training examples or parameter updates. *Task-level zero-shot inference* further describes applying a pre-trained model to a task that differs from its pre-training objective. CLIP exhibits this behaviour because contrastive image–text training can be reused as a classifier by comparing an image with prompted class descriptions, as Section 2.4 explains.

In summary, modern Computer Vision evolved from hand-crafted features and task-specific models to deep encoders, rich representations, self-supervised learning and contrastive training. These developments enabled joint vision-language models and low-shot transfer. The following section provides a more detailed description of vision-language foundation models.

2.2 Vision-Language Foundation Models

As outlined in Section 2.1, Computer Vision research moved from manual feature engineering to representation learning. This evolution included several stages that help to motivate the current *Vision-Language Foundation Models*.

Traditional machine learning and prediction. Early visual recognition pipelines used handcrafted features and classifiers such as support vector machines.

Feature extraction used methods such as the Histogram of Oriented Gradients (HOG) and the Scale-Invariant Feature Transform (SIFT). These systems work well for controlled tasks, but rely on careful feature design, and do not adapt easily to changing data distributions.

Deep learning from scratch and prediction. The introduction of CNNs enabled end-to-end training from pixels to labels. Networks such as AlexNet, VGG, and GoogLeNet learn hierarchical features from large labelled datasets. This removes hand-crafted feature design, although each task still requires many labels.

Supervised pre-training, fine-tuning, and prediction. To reduce the need for large task-specific datasets, the community adopted transfer learning. A model first trains on a broad source dataset such as ImageNet, and then fine-tunes on a smaller target dataset. This procedure reuses the general visual features from pre-training, and often improves performance when labels for the target task are scarce.

Unsupervised pre-training, fine-tuning, and prediction. Researchers then explored pre-training without class labels. Auto-encoders and generative models learn to reconstruct inputs or predict missing parts. These objectives encourage the model to capture structure in the data distribution. Fine-tuning on labelled datasets then adapts the learned features to specific tasks while retaining broad visual knowledge.

Vision-language pre-training and zero-shot prediction. The most recent stage joins images and text. Vision-Language Foundation Models, typified by CLIP [1], pre-train on large sets of image-text pairs. They learn aligned representations across the two modalities, and can perform zero-shot prediction by comparing an image embedding with embeddings of candidate text descriptions. This capability allows models to generalise to new tasks with little or no task-specific training.

The first three paradigms above typically treat labels as discrete indices. Architectures such as ResNet [4] and Vision Transformers (ViT) [8] compute an image feature vector and map it to an integer label using a linear classifier and a softmax loss. Although datasets store human-readable labels such as "bird" or "car", training converts these labels into indices to simplify loss computation. This mapping discards most of the semantic content in label text and restricts the model to a closed set of categories. Recognising new categories, in turn, requires further data collection and training.

Foundation models, in contrast, use *continuous representations* of categories. CLIP and related approaches map both images and text into a shared dense embedding space [1]. Each text description corresponds to a vector that encodes its meaning, and each image corresponds to a vector that encodes its content. Similarity between these vectors provides a basis for comparing visual and textual concepts.

This design breaks the hard link to a fixed set of discrete labels, enabling open-vocabulary recognition.

The term *foundation model* is introduced in [9]. Such models exhibit *emergence*, in which complex behaviours arise from scale and training regimes, and also *homogenisation*, in which many applications build on a shared set of methods and models. These properties bring benefits and risks. A shared model family can reduce duplication and speed adoption, but it can also propagate biases or errors to many downstream systems.

This chapter and the later technical chapters focus on Vision-Language Foundation Models and their adaptations. The following section introduces pre-training objectives for these models, before Section 2.4 gives a more detailed view of CLIP.

2.3 Pre-training Objectives

Within the vision-language pre-training and zero-shot paradigm, training is often referred to as pre-training, because the objective during this phase differs from those of the downstream tasks studied in later chapters. Researchers have proposed several families of objectives that appear in current vision-language models. This section groups them into contrastive, generative, and alignment-based objectives.

Contrastive image-text learning aims to learn representations that align images with their associated texts. The goal is to move matched image-text pairs closer together in the joint embedding space and separate unmatched pairs. A common loss for this setting is a symmetric image-text InfoNCE loss [10]. CLIP [1] popularises this objective in the large-scale image-text setting. The InfoNCE loss is formally defined as

$$\mathcal{L}^{\text{InfoNCE}} = \mathcal{L}_{I\to T} + \mathcal{L}_{T\to I} \tag{2.1}$$

with components

$$\mathcal{L}_{I\to T} = -\frac{1}{B}\sum_{i=1}^{B} \log \frac{\exp\left(z_i^I \cdot z_i^T/\tau\right)}{\sum_{j=1}^{B} \exp\left(z_i^I \cdot z_j^T/\tau\right)} \tag{2.2}$$

and

$$\mathcal{L}_{T\to I} = -\frac{1}{B}\sum_{i=1}^{B} \log \frac{\exp\left(z_i^T \cdot z_i^I/\tau\right)}{\sum_{j=1}^{B} \exp\left(z_i^T \cdot z_j^I/\tau\right)} \tag{2.3}$$

All z^I and z^T vectors are ℓ_2-normalised image and text embeddings, so their dot products are equivalent to cosine similarity. Here,

- B denotes the batch size,
- z_i^I and z_i^T denote the ℓ_2-normalised embeddings of the i-th image and text

sample,

- z_j^I and z_j^T denote the ℓ_2-normalised embeddings of the j-th image and text sample, and
- τ denotes the temperature hyperparameter, which scales similarity scores. A higher temperature produces a softer probability distribution, while a lower temperature makes it sharper.

Unified Contrastive Learning (UniCL) extends this idea to a joint image-text-label space [11]. The method combines supervision from standard image-label pairs with supervision from web data in image-text pairs. UniCL learns representations that are both discriminative for classification and semantically meaningful for retrieval and transfer.

Image-text-label contrastive learning introduces Supervised Contrastive Learning ideas into multimodal setups [7]. It encourages consistency between images, labels, and free-form text that describe the same concept. This approach links traditional supervised learning with multimodal contrastive objectives.

Generative objectives train a model to generate one modality from another. In the vision-language setting, this includes image captioning, text-to-image synthesis, and multimodal translation. These tasks require the model to capture detailed cross-modal structure. They often rely on *Masked Language Modelling*, masked image modelling, or both.

Masked image modelling masks a subset of image patches and asks the model to reconstruct them from the remaining patches. This task encourages the model to capture dependencies between regions. Masked Language Modelling masks a subset of tokens in a sentence and trains the model to predict the missing tokens from context. Masked cross-modal modelling combines these ideas by masking both image patches and text tokens in an image-text pair and predicting the masked elements from the unmasked ones. Text-to-image synthesis and image captioning instead train a decoder that autoregressively generates one modality from a representation of the other.

Alignment objectives refine the match between visual and textual representations, often after an initial pre-training stage. Instead of contrasting pairs across the whole batch, these methods aim to place semantically related image and text features close together in a shared space. Typical architectures use dual encoders and a fusion module that encourages local or global alignment between modalities.

Global alignment methods train a model to predict whether an image and a sentence match. They use a scoring function that takes pooled image and text features, along with a binary classification loss. Local alignment methods focus on region-word matching. They associate specific image regions with words or phrases

and refine features so that regions and phrases that refer to the same object are close in the embedding space. Such objectives tend to improve dense prediction tasks such as object detection and phrase grounding.

Contrastive, generative, and alignment-based objectives each target different aspects of cross-modal learning. In practice, many models combine them. Downstream tasks such as visual question answering, visual commonsense reasoning, and cross-modal retrieval, then serve as benchmarks for zero-shot and few-shot transfer.

2.3.1 Text Embeddings and Semantic Spaces

Vision-language systems rely on text encoders that map words or sentences into continuous vectors. Early approaches such as Word2Vec and GloVe learn static word embeddings from co-occurrence statistics, so each word receives a single vector that reflects its average usage. Word2Vec optimises local prediction objectives: the skip-gram and continuous bag-of-words models predict neighbouring tokens from a centre word or vice versa, learning vectors that place words with similar contexts close together in the embedding space [12]. GloVe instead factorises a global co-occurrence matrix and encourages word-vector differences to encode ratios of co-occurrence probabilities, thereby capturing broader semantic regularities such as analogies [13]. These representations support simple similarity measures and often serve as initialisation for downstream models. Later Transformer-based encoders [14], including BERT-style models and sentence encoders, produce contextual embeddings whose meaning depends on the full sentence.

In CLIP, the text encoder plays a similar role but is trained jointly with the image encoder under a contrastive objective. The resulting *textual embeddings* aligns with visual features and supports open-vocabulary classification. Later chapters of this thesis use both generic language embeddings such as GloVe, which reflect distributional semantics from large text corpora, and task-aligned embeddings from CLIP or other supervised models.

Sentence-level encoders make this idea explicit by optimising directly for sentence similarity. Sentence-BERT [15] contrasts a cross-encoder, which feeds a pair of sentences through a single BERT encoder and predicts a similarity score, with a bi-encoder, which passes each sentence through a Siamese encoder (two encoders that share parameters) and compares the resulting embeddings with cosine similarity. Cross-encoders capture rich interactions between tokens in both sentences, but they require a forward pass for every pair, and so scale poorly to retrieval over many candidates. Bi-encoders instead pre-compute one embedding per sentence and reuse it across pairs, which makes them well-suited to prototype-based methods that average or compare many sentence representations.

Recent work applies contrastive learning directly to text encoders to improve sentence embeddings. Methods such as SimCSE and related approaches treat two dropout-perturbed passes of the same sentence as a positive pair, and use other sentences in the batch as negatives, encouraging the encoder to map semantically equivalent sentences close together while keeping unrelated sentences apart [16]. This training produces more isotropic embedding spaces, where directions carry information more evenly, and cosine similarity better reflects semantic similarity. These text-only contrastive objectives resemble CLIP's image-text training in their use of positives and negatives, but they operate within a single modality rather than across modalities and therefore focus on refining purely linguistic structure.

Other methods inject explicit hierarchical information into textual embeddings. Contextualised Taxonomy Prediction (CTP) fine-tunes pretrained language models to predict parent-child relations between terms. Then it reconciles these predictions into taxonomic trees using a maximum spanning tree algorithm [17]. The model conditions on term glosses and local context so that vectors capture hypernym structure rather than only surface similarity. These results show that Transformer-based sentence encoders can capture taxonomic relations when trained with appropriate objectives, motivating the use of taxonomy-aware sentence representations in later chapters.

2.4 CLIP: Contrastive Language-Image Pre-training

CLIP is a prominent example of a Vision-Language Foundation Model. It shows strong task-level zero-shot inference performance on a wide range of datasets. In many cases, it matches or approaches the accuracy of supervised baselines on benchmarks such as ImageNet, Stanford Cars, Food-101, and Oxford Pets [1].

The original CLIP paper evaluates more than 30 datasets across tasks such as optical character recognition, video action recognition, geo-localisation, and *fine-grain* object classification. CLIP adapts across these tasks without task-specific training, as it relies on the same text and image encoders, and a *cosine similarity* measure in the shared embedding space. On ImageNet, CLIP matches the performance of a supervised ResNet-50 in a pure zero-shot setting, without access to ImageNet training labels.

CLIP trains on image-text pairs collected from public internet sources. Many of these pairs use alternative text (alt text) that web authors provide for accessibility and for search indexers. Web scraping of public sites yields hundreds of millions of noisy, weakly aligned pairs. CLIP uses a contrastive loss that maximises cosine similarity between matching image and text embeddings, and minimises similarity for non-matching pairs. The architecture is summarised in Figure 2.1.

The *aligned embedding space* of CLIP enables a simple zero-shot classification rule. For a downstream task, one writes a set of prompts such as "a photo of a {class}" for each class. The text encoder maps each prompt to a vector. Given an image, the image encoder maps it to a vector in the same space. The classifier then assigns the image to the class whose textual embedding has the highest cosine similarity to the image embedding. In this way, CLIP serves as a general-purpose vision-language model that can tackle new tasks without requiring parameter changes.

Recent work shows that careful visual prompting can change this behaviour. For example, drawing a red circle around an object can increase the contribution of that region to CLIP's image–text similarity while preserving global context [18]. This simple input modification leads to strong performance on zero-shot referring expression comprehension and keypoint localisation tasks, where the goal is to locate specific objects or landmarks from a natural-language description.

2.4.1 Dense Features from CLIP

Since CLIP trains on image-text pairs, one might expect it only to produce global image-level features. However, the attention layers in the image encoder also contain spatial information. MaskCLIP [19] shows how to extract dense patch-level features from the final attention layer of the CLIP image encoder. These features preserve the association between visual regions and textual concepts. The method treats textual embeddings as classifiers and applies them as 1×1 convolutions over spatial features to obtain segmentation masks, without an extra mapping network.

The MaskCLIP authors also introduce two post-processing steps [19]. First, key smoothing computes cosine similarity between key features at different spatial locations and averages predictions over similar keys. Second, prompt denoising removes prompts that are unlikely to correspond to any region in the image. Both steps improve segmentation accuracy without extra pre-training. MaskCLIP further serves as a strong pseudo-label generator in a *self-training* framework. By using MaskCLIP to label unlabelled images and then training a standard segmentation network on these labels, the extended method, MaskCLIP+, achieves state-of-the-art results on several benchmarks.

MaskCLIP can supply segmentation pseudo-labels for methods with limited manual annotation [19]. Its tight link to CLIP enables segmentation of open-vocabulary, fine-grained classes described by phrases such as "white car" or "red bus". Training on diverse, web-curated images also gives robustness to natural distribution shifts and common input corruptions.

The image encoder in CLIP uses a *transformer* with multi-head attention, where a global pooled feature acts as the query, and spatial features act as keys and values

[19]. The final output is a weighted sum of spatial features that then passes through a linear layer $F(\cdot)$. This structure implies that the intermediate features already contain rich local semantics. MaskCLIP modifies the image encoder by removing the query and key embedding layers, reconfiguring the value embedding, and remapping the final linear layer into two 1×1 convolutional layers. The text encoder remains unchanged and produces the textual embedding, which serves as the class prototype for pixel-level classification. This design turns CLIP into a model that outputs pixel-level masks rather than only global predictions.

Subsequent work extends this idea in both trained and training-free directions. DenseCLIP converts the original image-text matching objective into a pixel-text matching signal and uses image-conditioned context prompts to transfer CLIP knowledge to segmentation, object detection, and instance segmentation [20]. More recent training-free methods instead alter the inference dynamics of CLIP itself. CLIP Surgery changes attention and feature composition to reduce noisy activations and recover more useful open-vocabulary localisation [21]. Self-Calibrated CLIP (SC-CLIP) identifies anomalous tokens in ViT-based CLIP, replaces or suppresses them using local context, and fuses multi-level features in a two-pass process to improve open-vocabulary segmentation without additional training [22]. These methods all address the same limitation: image-level contrastive pre-training gives CLIP strong global semantics but does not guarantee reliable local features.

This local-feature limitation is especially important for fine-grain recognition. Part-based fine-grained few-shot work shows that explicitly localising semantic parts and then comparing pose-normalised part appearances can substantially improve recognition when classes differ in small regions such as bird heads, bellies, or wing markings [23]. Dense CLIP features provide a natural bridge between open-vocabulary segmentation and fine-grained recognition, but the bridge is not automatic: patch-level features must preserve the visual distinctions that class-level embeddings tend to pool away.

Recent evaluations show the same limitation. FOCI is a multiple-choice benchmark for fine-grain object classification with distractor labels mined using CLIP, and it shows that CLIP models can outperform larger vision-language models on this type of fine-grained distinction [24]. This suggests that adding a large language decoder does not by itself solve fine-grained visual discrimination; the alignment between local visual cues and text remains a bottleneck.

There is also a text-side constraint. Standard CLIP uses a fixed text context, and Long-CLIP reports that although the nominal context length is 77 tokens, the effective useful text length can be much shorter, often below 20 tokens [25]. Long class descriptions, part lists, or LLM-generated attributes may therefore be truncated or compressed poorly by the original CLIP text encoder. This finding is relevant to

later chapters in this thesis, where fine-grained side information is expressed as natural language. Since Long-CLIP was published after the work described here, our objective at the time was limited to fitting textual descriptions within the encoder's nominal context window, rather than accounting for the encoder's practical effective capacity.

Despite strong performance, CLIP still faces limits that arise from its training data. Alt text often consists of short noun phrases or keyword lists that support search ranking, rather than full captions. As a result, CLIP tends to underuse fine-grain context. Later sections in this work describe methods that inject domain-specific knowledge or additional training signals to mitigate this issue.

2.5 Improvements to CLIP Training

Several works propose modifications that improve the training efficiency and robustness of CLIP-style models. Fast Language-Image pre-training (FLIP) [26] increases efficiency by randomly masking and dropping many image patches during training. This strategy allows training on more image-text pairs in the same wall-clock time and memory budget. Across many downstream tasks, FLIP achieves better accuracy than a standard CLIP model trained on the same data.

The standard CLIP pipeline does not distinguish between visual content and text within images. When the text in an image does not match the visual content, this can reduce robustness. Excluding images with visible text during training improves classification and retrieval on several public benchmarks [27].

Another line of work augments CLIP with external knowledge. K-LITE [28] enriches text entities using structured resources such as WordNet [29] and Wiktionary. During training, it replaces or extends input text with knowledge-enhanced descriptions. This strategy increases sample efficiency and improves transfer learning performance in zero-shot and few-shot settings compared to standard CLIP training.

Language-augmented CLIP (LaCLIP) [30] modifies the text side of training. Standard CLIP keeps each text description fixed. LaCLIP instead uses a large language model to generate multiple paraphrases for each image description. Training with these rewrites exposes the model to a broader range of text that refers to the same visual content. Experiments on datasets such as CC3M, CC12M, RedCaps, and LAION-400M show that LaCLIP improves zero-shot accuracy on ImageNet over the base CLIP model.

2.6 Parameter-Efficient Transfer Learning

In this section, we review methods that adapt Vision-Language Foundation Models to new tasks without full fine-tuning. These parameter-efficient transfer learning methods modify only a small number of additional parameters, thereby reducing compute cost and preserving generalisation.

2.6.1 Prompt Engineering and Prompt Tuning

One way to adapt CLIP is to change the text prompts used at inference time. The original CLIP paper uses the template "a photo of a {class}". CoOp [31] replaces this hand-crafted template with a set of learnable vectors. CuPL [32] instead uses a large language model to generate descriptive prompts for each class.

Context Optimisation (CoOp) [31] keeps the CLIP encoders fixed and learns a small set of context tokens that precede each class name. These tokens are initialised either randomly or with *pretrained weights* such as Word2Vec or GloVe. During training, gradients flow through the text encoder to update only these context tokens. This procedure adapts CLIP to a new dataset without changing the vision or text backbones.

Training in CoOp minimises cross-entropy loss over the learnable context tokens. Because the encoders remain frozen, the optimisation focuses on extracting task-relevant information that already resides in the pretrained weights. CoOp shows that this strategy can match or surpass hand-designed prompts.

CuPL [32] extends prompt design to large language models. Researchers provide a few general templates such as "Describe what a __ looks like", "Describe a __", and "What are the identifying characteristics of a __". They then fill the blanks with combinations of a broad category and a class name (for example, "pet" and the name of a dog breed), or with only the class name for broader datasets such as ImageNet. The large language model generates many candidate prompts for each class. Using these rich, class-specific prompts in CLIP often improves zero-shot classification performance compared to short, handwritten templates.

Despite its simplicity, CuPL outperforms dataset-specific handcrafted prompts on most benchmarks that the authors tested. This result suggests that general prompt-generation strategies can replace manual *prompt engineering* in many cases.

2.6.2 CLIP Adapters

CLIP-Adapter [33] offers an alternative to prompt learning. Instead of optimising prompts, it inserts small feature adapters into the visual or language encoders. These adapters take the form of bottleneck layers that process existing features,

and then blend them with the original features through a residual connection. This design allows the model to add task-specific information while preserving most of the pre-trained representation.

In more detail, CLIP-Adapter adds two linear layers after the last layer of the vision or language backbone. The adapter compresses the CLIP feature into a low-dimensional bottleneck, applies a non-linearity, and then projects back to the original dimension. A residual connection then forms a convex combination of the adapted feature and the original CLIP feature, controlled by a scalar coefficient α. The class logits are computed from this blended feature using the standard CLIP text prototypes, so adaptation happens at the feature level rather than by directly mixing logits. This simple head can be viewed as a two-path architecture in which one path preserves the original CLIP decision rule, while the other path applies task-specific corrections.

The CLIP-Adapter method trains in a standard few-shot regime using cross-entropy loss on top of the classifier applied to the blended features, while keeping the CLIP encoders frozen [33]. The method improves robustness to domain shift, while adding only a small number of parameters. On benchmarks such as ImageNet, Flowers, and DTD, CLIP-Adapter consistently outperforms prompt-learning approaches like CoOp under the same shot counts, especially when the visual appearance differs from the web images used during CLIP pre-training.

Subsequent analysis of the architecture shows how the blending ratio controls the trade-off between generic and task-specific information [33]. Fine-grained datasets often benefit from relying more on the adapter branch, whereas broad datasets favour the original CLIP features. Across several datasets and shot counts, CLIP-Adapter outperforms CoOp and other prompt-based methods. This observation suggests that lightweight feature adaptation can be more effective than tuning prompts alone when adapting vision-language models.

CLIP-Adapter requires gradient-based training of the adapter parameters. Tip-Adapter [34] removes this requirement. It builds a key-value cache from the few-shot training set, where keys are CLIP features and values are label indicators. At inference time, it combines logits from the cache with logits from the CLIP classifier. This non-parametric mechanism produces adapter weights without explicit backpropagation, and achieves performance comparable to or better than CLIP-Adapter.

2.6.3 Efficient Tuning and Validation-Free Adaptation

Few-shot adaptation of CLIP-style models often falls into two categories: prompt learning and adapter-based methods. Prompt learning updates global text or visual prompts and requires access to full encoder weights and gradients [31]. Adapter

methods operate in feature space and can avoid backpropagating through the full backbone. Some adapter variants train a linear classifier or a shallow multilayer perceptron on top of frozen CLIP features. Others, such as Tip-Adapter, use a training-free cache.

Recently, validation-free few-shot learning emerges as a stricter setting [35]. In this setting, methods do not use a validation set to tune hyperparameters. Instead, they must use a single configuration across datasets. This constraint reflects practical scenarios in which labelled validation data are either unavailable or expensive to obtain.

CLIP-Adapter appends learnable bottleneck layers to the image branch of CLIP. During few-shot fine-tuning, it keeps the main backbone frozen. Although extra layers can overfit in few-shot regimes, CLIP-Adapter uses a residual connection that mixes new knowledge with pre-trained features. A scalar blending ratio, α, controls how much the classifier relies on adapted versus original features. By grid-searching α per dataset, prior work finds that fine-grain datasets such as DTD favour larger α, while generic datasets such as ImageNet favour smaller α [33]. Setting $\alpha = 0$ recovers pure zero-shot CLIP, and setting $\alpha = 1$ fully trusts the adapted features.

Tip-Adapter also uses a blending ratio α to combine cache-based logits with CLIP zero-shot logits [34]. The authors argue that α should be larger when the domain gap between pre-training and few-shot data is significant, because the few-shot set then needs to contribute more information.

Proto-Adapter [36] adapts CLIP by learning a single-layer class-prototype adapter and blending adapter logits with zero-shot logits using a ratio α. Reported values of α vary widely across datasets. For example, Caltech-101 uses $\alpha = 0.8$, FGVC-Aircraft uses $\alpha = 0.2$, Oxford 102 Flowers uses $\alpha = 0.6$, and EuroSAT uses $\alpha = 1.0$. Retentive CLIP Adapter Tuning (RCAT) adapts CLIP for few-shot video recognition by combining CLIP with a retentive network that captures temporal structure. RCAT performs a grid search over α for each dataset and finds that fine-grain or high-domain-gap tasks again favour larger blending ratios [37].

Per-dataset grid search over α is incompatible with validation-free few-shot evaluation. This constraint motivates methods that set α automatically without labelled validation data.

Only a small number of works propose such automatic blending strategies for CLIP Adapters. SVL-Adapter [38] combines vision-language pre-training with self-supervised learning and derives α from CLIP zero-shot confidence scores. It sets

$$\alpha = \frac{1}{N} \sum_{i=1}^{N} \max_{k} P(y_i = k \mid x_i),$$

where N denotes the number of test images and $\max_k P(y_i = k \mid x_i)$ denotes the maximum predicted probability for image x_i. When CLIP is confident, the method uses a small α and relies on zero-shot predictions. When CLIP is uncertain, it increases α and gives more weight to adapted features.

PathCLIP [39] targets pathology images. It introduces Residual Feature Refinement (RFR), a shallow residual module that adapts CLIP features to the pathology domain. PathCLIP learns a self-adaptive blending ratio that balances CLIP knowledge with task-specific features. It uses a dual-view vision contrastive (DVC) metric inspired by SimCLR [5]. DVC compares encoder outputs under weak augmentations, such as flips and shifts, and strong augmentations, such as RandAugment [40] and CTAugment [41], using cosine similarity. High DVC scores indicate stable representations under augmentation. PathCLIP combines DVC with support-set accuracy into a bound-constrained rule for selecting α. This design seeks to avoid overfitting when adapting large models with few labels.

Later chapters build on these ideas. In Chapter 3 we describe validation-free adaptation in more depth and propose a new way to learn a blending ratio that does not rely on a held-out validation set.

2.7 Low-Shot Learning and Side Information

This section reviews zero-shot and few-shot learning setups and the role of side information, focusing on settings where models need to recognise unseen classes or adapt to new classes with few labels.

2.7.1 Standard Zero-Shot Learning Setting

Zero-shot Learning (ZSL) allows a model to recognise objects from categories that do not appear in the training labels. The model learns to transfer knowledge from seen categories to unseen categories. To do this, it uses an intermediate semantic space that connects classes.

CUB provides images with human-annotated attributes such as beak colour, eye colour, and wing shape [42]. These attributes supply the class-level side information used by many standard ZSL methods. FGVC-Aircraft is also a fine-grained recognition benchmark, but it does not provide comparable human-annotated attribute vectors [43]. Its official annotations comprise tight object bounding boxes and a hierarchy of manufacturer, family, and variant labels. The hierarchy supports evaluation at different levels of specificity, while methods that require descriptive aircraft attributes must obtain them from another source. Keeping this distinction clear matters because a hierarchical class label provides taxonomic structure, whereas

an attribute vector explicitly describes visual properties that can be shared across classes.



Figure 2.2: Example aircraft variants from FGVC-Aircraft. The dataset contains 100 visually similar aircraft variants organised by manufacturer, family, and variant. Image reproduced from [43].

2.7.2 Multimodal Zero-Shot Learning Setting

Multimodal ZSL extends this idea by using additional sources of information. In addition to visual features and attributes, methods can exploit textual descriptions, class embeddings, or other side information. A cross-modal embedding framework can handle multiple types of side information [44].

Side information refers to any data beyond raw pixels that helps a model recognise unseen classes. Text descriptions, class embeddings, and other metadata can all serve this role. Side information enriches the semantic space and can improve ZSL performance by capturing class relations. For example, class embeddings may encode that two bird species share similar habitats or colours, while textual descriptions provide natural-language explanations of visual traits.

Cross- and Distribution-aligned Variational Autoencoders (CADA-VAE) [44] use variational autoencoders to map visual and semantic features into a shared latent space. The model uses one encoder per modality and a common latent representation. It uses decoder networks to reconstruct the original inputs from this latent code. Figure 2.3 shows the architecture.

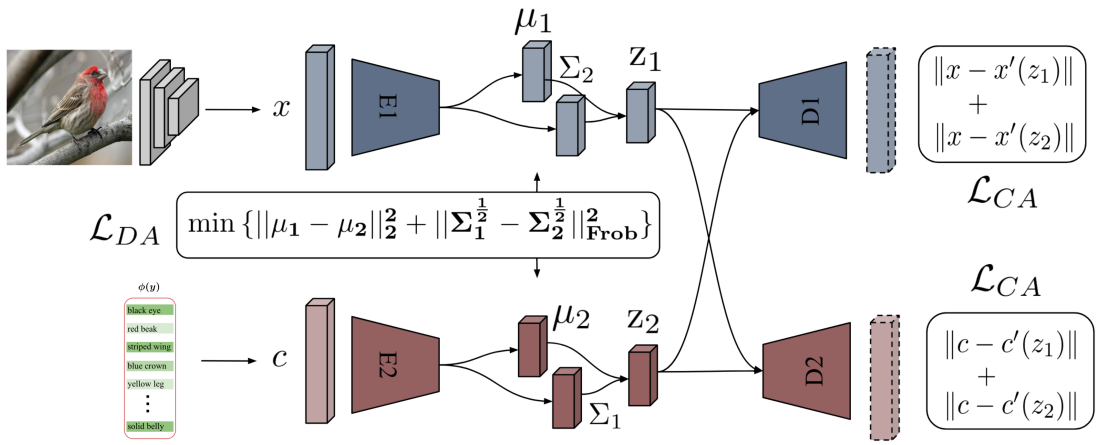


Figure 2.3: Cross- and Distribution-aligned Variational Autoencoder architecture for generalised zero-shot learning. Two variational autoencoders, one for images and one for class attributes, train jointly with reconstruction, cross-alignment, and distribution-alignment losses. Adapted from [44].

Modality-specific autoencoders require additional regularisation to learn similar representations. CADA-VAE uses two alignment losses to achieve this. A cross-alignment loss performs cross-reconstruction, in which an encoder from one modality feeds into a decoder from another modality while preserving class identity. A distribution-alignment loss minimises the Wasserstein distance between latent Gaussian distributions for different modalities. Together, these losses encourage a shared latent structure that supports transfer between modalities.

Unstructured text documents can also serve as side information for ZSL [45]. Such documents describe categories and their relationships in natural language. A later work introduces Large Language Models to generate class descriptions as additional side information [46]. These descriptions can cover fine details that attribute lists may miss.

The Attribute Prototype Network (APN) jointly learns global and local features using class-level attributes [47]. The method employs a visual-semantic embedding layer for global features and an APN module for local features. The APN regresses attributes and decorrelates them from intermediate features, which helps the model encode subtle differences between classes. Figure 2.4 illustrates the architecture.

The weakly supervised framework in [47] injects attribute-related losses into intermediate CNN layers. This encourages local features to focus on visual attributes that define object appearance. The method uses only class-level attribute vectors and their semantic relatedness for supervision, without human annotations linking specific regions to attributes. To reduce the effect of spurious correlations, it incorporates attribute-relatedness into the loss, thereby improving robustness and the discrimination of local features.

In this framework, class-level attribute vectors supervise the learning of attribute

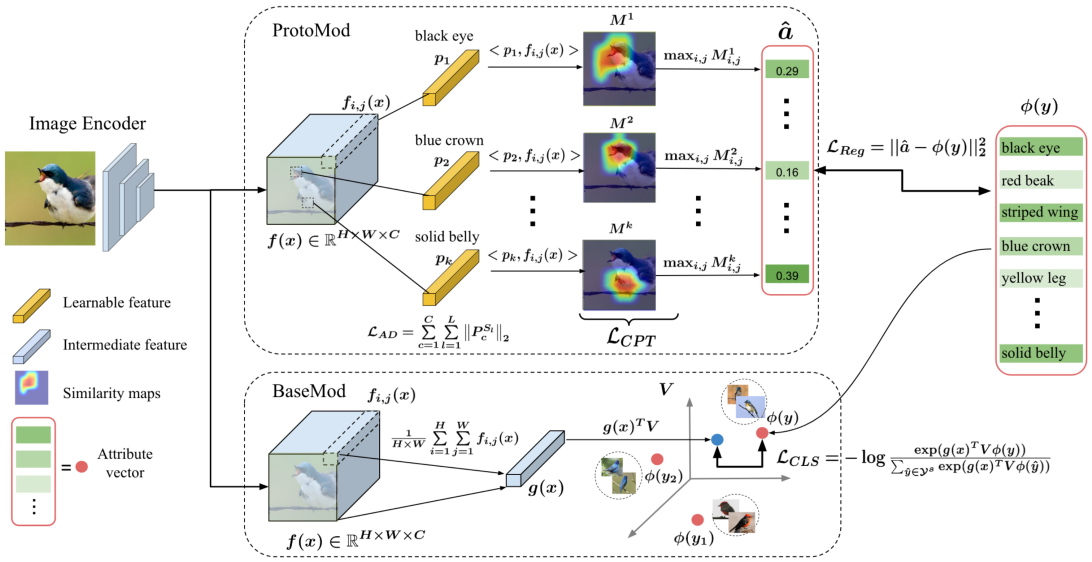


Figure 2.4: Attribute Prototype Network. An image encoder extracts features. A base module performs zero-shot classification via a visual-semantic embedding layer. A prototype module learns attribute prototypes and localises them via similarity maps. Adapted from [47].

prototypes. The network treats attribute prediction as a regression problem, and minimises mean squared error between ground-truth attributes $\phi(y)$ and predicted attributes $\hat{\phi}(y)$.

2.8 Fine-Grained Classification and Spatial Alignment

Fine-grain visual classification and spatial alignment are central to later chapters in this thesis. Section 2.6 covers parameter-efficient transfer learning; this section turns to the local visual details that make fine-grained recognition difficult.

2.8.1 Fine-Grained Visual Classification

Traditional fine-grain visual classification methods rely on explicit part localisation. Many systems use bounding boxes or keypoint annotations for parts such as head, wings, or tail to align images before classification [48]. Methods such as DeepLAC [49] and SPDA-CNN [48] model parts and their alignments to resolve subtle inter-class differences. These annotations are expensive to collect and may not scale to large or diverse datasets.

In parallel, meta-learning methods study how to adapt embedding spaces to each few-shot task. FEAT addresses this by learning a task-specific set-to-set transformation that operates on the whole support set at once [50]. A Transformer encoder

[14] takes support embeddings as input and outputs adapted support embeddings that are contextualised by other classes in the same episode. Class prototypes are then computed from these adapted embeddings, which move class centres away from clutter and towards the query clusters that define the current task. The set-to-set design enforces permutation invariance over the support examples and supports interpolation and extrapolation across tasks. Experiments on standard, cross-domain, transductive, and low-shot benchmarks show consistent gains over fixed-embedding baselines.

With the rise of vision-language models, newer methods seek to use natural-language descriptions to guide fine-grain recognition. Some works [51, 32] use Large Language Models to generate descriptive phrases such as "a bird with a red breast" and then match image features to these phrases. These approaches treat attributes as unordered sets or average text embeddings, which may weaken the link between specific image regions and textual descriptions. In this thesis, we ask *inter alia* whether models can learn explicit spatial-semantic alignment in few-shot regimes or whether the structure of the embedding space limits such alignment.

2.8.2 Visual Prompting and Attention Steering

Prompting in natural language processing modifies the input text to steer the model's behaviour. Visual prompting applies a similar idea to images [52]. Methods either learn small pixel-level perturbations or overlay simple shapes or markers on the image to influence model predictions.

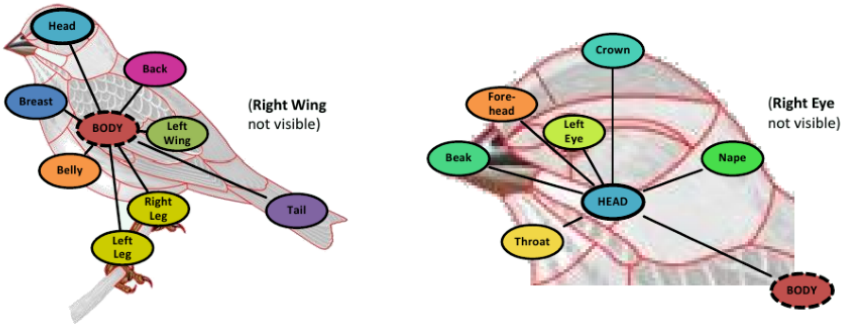


Figure 2.5: Part location markers collected in CUB. The dataset annotates bird anatomy with visible part locations, including beak, head, wing, tail, and leg markers. Image reproduced from [42].

Beyond input prompting, some methods adjust attention patterns during inference. CLIP Surgery [21] points out that standard Vision Transformers use bidirectional attention that encourages the [CLS] token to aggregate information from all patches. This behaviour can lose local detail. The authors of CLIP Surgery propose

a value-value attention variant that preserves more spatial distinctiveness. In this chapter, we evaluate both visual prompting and CLIP Surgery variants on the CUB dataset [42] to study their effect on part-level recognition.

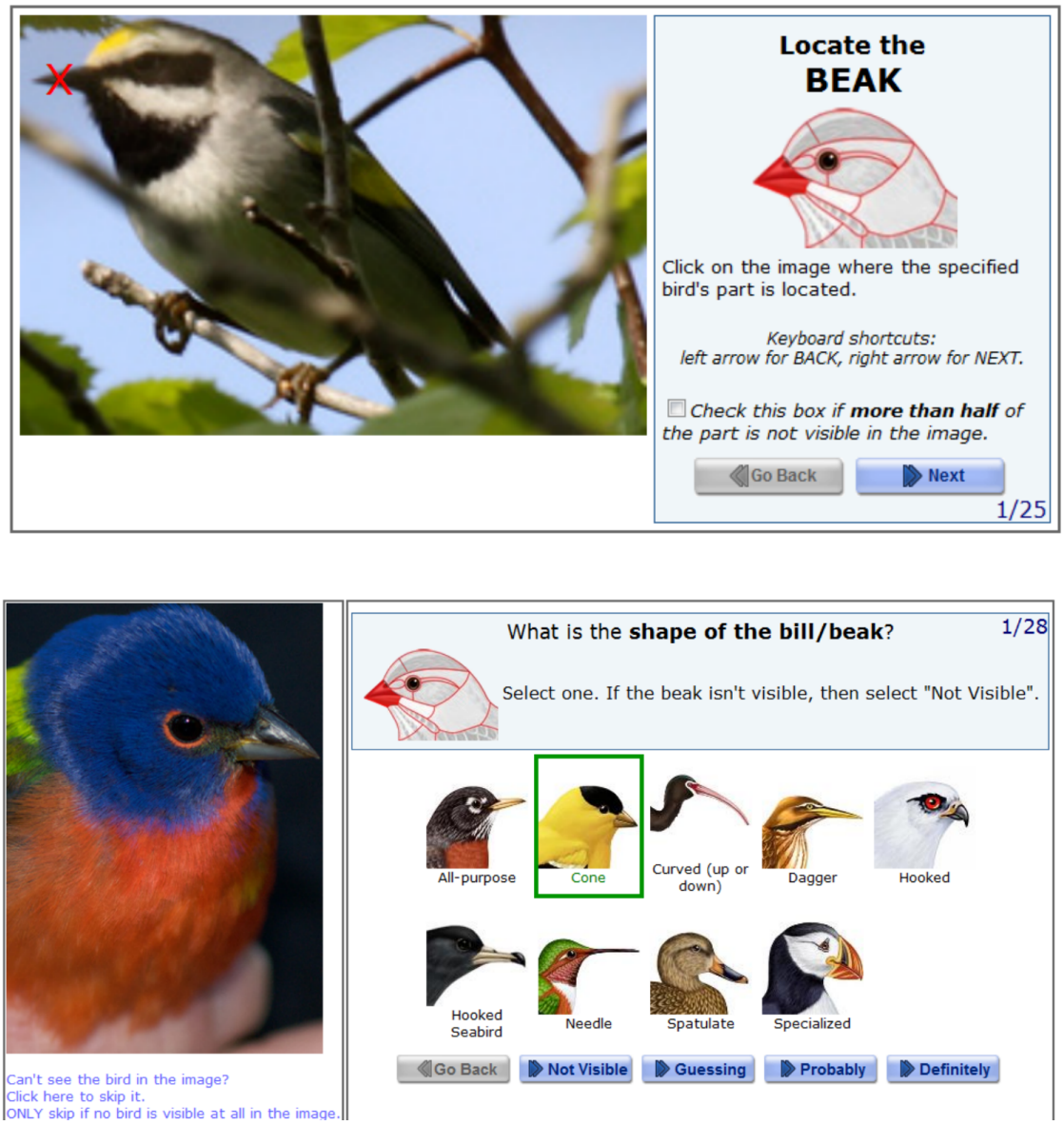


Figure 2.6: Annotation interfaces used in CUB. The upper panel collects point locations for a specified bird part, while the lower panel collects part-specific visual attributes such as bill or beak shape. Images reproduced from [42].

2.9 Performance Prediction and Out-of-Distribution Detection

Performance prediction and out-of-distribution (OoD) detection form a second thread that connects to this thesis. Many applications require a model not only to clas-

sify in-distribution data but also to identify samples that fall outside its training distribution.

Prior work usually focuses on one of two goals: detecting whether a single sample is in distribution, or predicting target-domain performance from correlations between in-distribution and out-of-distribution scores. Chapter 6 addresses a different setting: dataset-level zero-shot accuracy prediction without a full labelled test set. Instead of assuming access to in-distribution data or predefined outlier labels, it probes each target class with Large Language Model-generated counterfactual descriptions and uses the resulting image–text similarity scores to estimate transfer performance.

OoD detection is critical in safety-critical domains such as autonomous driving and medical imaging, where misclassification of unknown samples can have serious consequences. One zero-shot OoD method uses a large set of auxiliary outlier labels to create pseudo-OoD prompts and refine them into prototypes that score OoD likelihood [53]. ZOC [54] extends CLIP with a text generator that produces candidate unseen labels, and performs OoD detection without prior knowledge of unseen classes. A related study shows a strong linear correlation between in-distribution accuracy and OoD accuracy [55]. This relation allows estimation of OoD performance using only unlabelled data by measuring model agreement.

Other work uses counterfactuals or foils to probe model understanding. The FOIL-COCO dataset modifies each caption by changing a single word to introduce a subtle mismatch with the image [56]. Models must identify the incorrect word to succeed on this task.

Recent studies also analyse how data distribution shapes the performance of vision-language models. Multimodal models can be sample-inefficient, requiring exponential increases in concept frequency during pre-training to obtain linear gains in downstream performance [57]. Concept frequency follows a long-tailed distribution similar to Zipf's law [58]. In practice, users of proprietary vision-language models often lack access to the underlying pre-training data, making it hard to verify these distributions directly. Even for open-source datasets, concept counting is difficult because the same concept can appear under many names, and web-scraped alt text contains noise.

2.10 Chapter Summary

This chapter reviews the background, methods, and open problems that motivate the contributions in Chapters 3–6.

2.10.1 Primary Findings

From hand-crafted features to foundation models. Computer vision evolved through several stages. Early systems use hand-crafted features and shallow classifiers. Deep learning replaces these pipelines with end-to-end training from pixels to labels. Transfer learning then allows models to pre-train on large datasets and fine-tune on smaller target datasets. Vision-language foundation models extend this idea by learning a shared embedding space for images and text. CLIP trains on 400 million image-text pairs and achieves strong zero-shot performance on many benchmarks without task-specific fine-tuning.

Contrastive pre-training produces aligned representations. CLIP uses a symmetric contrastive loss that pulls matching image-text pairs together and pushes non-matching pairs apart. The result is a joint embedding space where cosine similarity reflects semantic relatedness. This design supports open-vocabulary classification: a model compares an image embedding with text embeddings for candidate class descriptions and selects the closest match. The approach removes the hard link between training labels and inference labels.

Parameter-efficient transfer learning adapts CLIP with few parameters. CLIP-Adapter inserts small trainable bottleneck layers trained on few-shot cases. CLIP-Adapter uses a blending ratio to control how much the classifier relies on the original CLIP features versus the adapted features. Prior work shows that the optimal blending ratio varies across datasets.

Side information bridges seen and unseen classes. Zero-shot and few-shot learning leverage side information, such as attributes, class embeddings, or textual descriptions, to transfer knowledge from seen to unseen classes. Architectures such as CADA-VAE align visual and semantic features in a shared latent space. Attribute Prototype Networks learn to localise attributes in image regions using only class-level supervision. Recent work uses large language models to generate class descriptions as side information, thereby reducing the need for manual annotation.

Fine-grained classification requires local detail. Fine-grained datasets contain classes that differ only in subtle local features such as beak shape or wing pattern. Traditional methods use part annotations to align images before classification. Vision-language models often discard spatial detail through global pooling. Methods such as MaskCLIP extract dense patch-level features from CLIP, but the alignment between these features and fine-grained text descriptions remains limited.

Performance prediction and robustness. Practitioners often need to estimate model performance on a new dataset before labelling a large test set. Related work on out-of-distribution detection and counterfactual probing shows that the margin between correct and incorrect predictions in the embedding space can indicate model robustness. However, no methods estimate dataset-level zero-shot accuracy with minimal labelled data.

2.10.2 Gaps and Research Questions

We identify four gaps in the literature that the following chapters address.

Gap 1: Validation-free adaptation. Many adapter-based methods tune the blending ratio on a validation set or report the best results from a grid search on the test set to find the optimal blending ratio for each dataset. In strict few-shot settings, no validation set is available. Prior automatic blending strategies either fix the ratio before training based on zero-shot accuracy on the support set or tune it during training by using differences in embedding distributions between strong and weak augmentations. We set out to benchmark these strategies and improve upon them.

Research Question 1 (Chapter 3): How can one adapt CLIP in a strict few-shot setting without a separate validation set?

Gap 2: Generating side information from large language models. Concurrent work shows that richer text prompts can improve zero-shot performance, but no systematic study has compared multiple sources of text (labels, definitions, descriptions) across various text encoders and evaluated whether Large Language Models can generate side information that matches or exceeds human-annotated attributes for fine-grained low-shot image classification.

Research Question 2 (Chapter 4): How can Large Language Models generate side information for CLIP that improves zero-shot and few-shot transfer?

Gap 3: Spatial-semantic alignment for fine-grained recognition. CLIP produces strong global features, but fine-grained classification depends on local detail. No prior work systematically tests whether few-shot generalisation can be improved on fine-grained datasets using the alignment between spatial visual features and part-level text descriptions generated by a Large Language Model.

Research Question 3 (Chapter 5): Can explicit alignment between spatial visual features and Large Language Model-generated part descriptions improve few-shot generalisation on fine-grained image classification tasks?

Gap 4: Estimating zero-shot performance with minimal labels. Practitioners need early estimates of model performance before labelling large test sets. Prior methods for out-of-distribution detection focus on sample-level decisions rather than dataset-level accuracy prediction. No previous work uses Large Language Model-generated counterfactuals to estimate dataset-level zero-shot accuracy from a single labelled example per class.

Research Question 4 (Chapter 6): Can CLIP's zero-shot accuracy on a full test set be estimated from one labelled image per class and LLM-generated counterfactuals?

Chapter 3

HOSO-Adapter: Validation-Free Few-Shot Adaptation

3.1 Introduction

Contrastive Language Image Pre-training (CLIP) [1] learns visual representations from large collections of paired image and text data. A CLIP model transfers this knowledge to new tasks through zero-shot prediction on hand-crafted prompts. As discussed in Chapter 2, this approach already achieves good performance on many benchmarks, but practical systems often need further adaptation to new domains with only a few labelled examples.

Many recent methods keep the original CLIP encoders fixed and add small trainable modules. These modules include prompt tuning layers and adapter networks. Prompt tuning updates text prompts and requires full backpropagation through the model. Adapter methods insert light multi-layer perceptrons into the visual backbone and update only the adapter parameters. This design reduces computation and helps preserve the base model's general knowledge. However, in a strict few-shot setting (*i.e.* few-shot with no validation set), these methods still face a central question: how much weight should be assigned to the adapted features relative to the original CLIP features?

A wide range of CLIP-Adapter-style methods [33, 34, 36, 37] address this question through a *blending ratio*. The blending ratio, often denoted by α, controls a linear mix of base CLIP features and adapter features. A low value of α gives more weight to CLIP features and keeps behaviour close to zero-shot CLIP. A high value of α gives more weight to the adapter, allowing it to fit the target task more closely. A suitable value of this ratio strongly affects performance, especially when training data are scarce.

Many prior works tune the blending ratio on a validation set. In strict few-shot

benchmarks, the protocol does not allow such a validation set [35]. A fixed ratio across datasets also performs poorly because each dataset needs a different trade-off between prior knowledge and adaptation. Figure 3.3 shows that the best ratio varies over datasets and that a single fixed value is rarely ideal.

We select the blending ratio without separate validation data by using the observed correlation between one-example-per-class and full-test CLIP accuracy. Figure 3.1 compares evaluations of the same frozen zero-shot CLIP classifier on three datasets: accuracy on one labelled image sampled per class, and accuracy on the full test set. The strong correlation indicates that this one-example-per-class estimate provides a useful probe of dataset difficulty for CLIP.

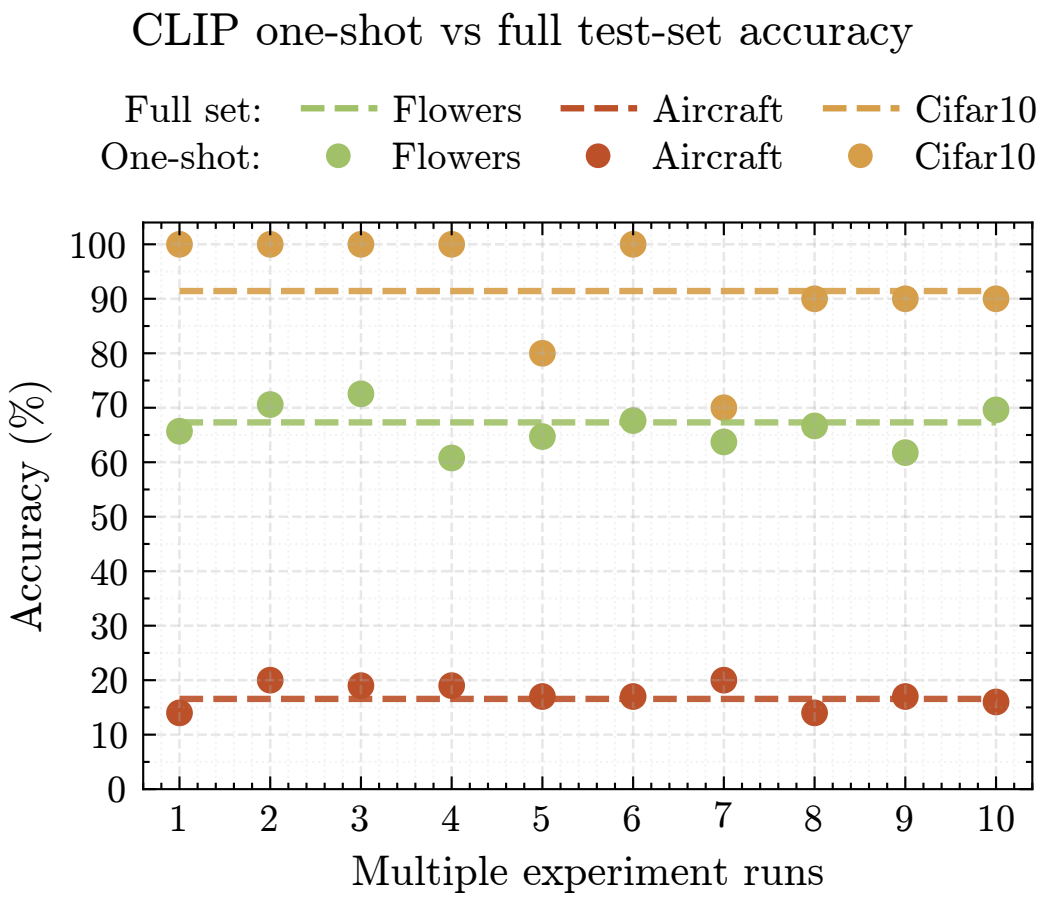


Figure 3.1: Accuracy of the frozen zero-shot CLIP classifier on one labelled image sampled per class, compared with accuracy on the full test set across several datasets and runs. The x-axis reports the one-example-per-class probe accuracy of the zero-shot classifier, not the accuracy of an adapted one-shot model.

We introduce *Hold-One-Shot-Out (HOSO)* to explore this idea. For each class, we hold out one labelled example and place it in a small cache. We then learn the blending ratio using only this cache, while the adapter parameters learn from the remaining examples. This procedure does not use any external validation set and

thus follows the strict few-shot setting of [35]. At the same time, it allows the ratio to change over training rather than remain fixed.

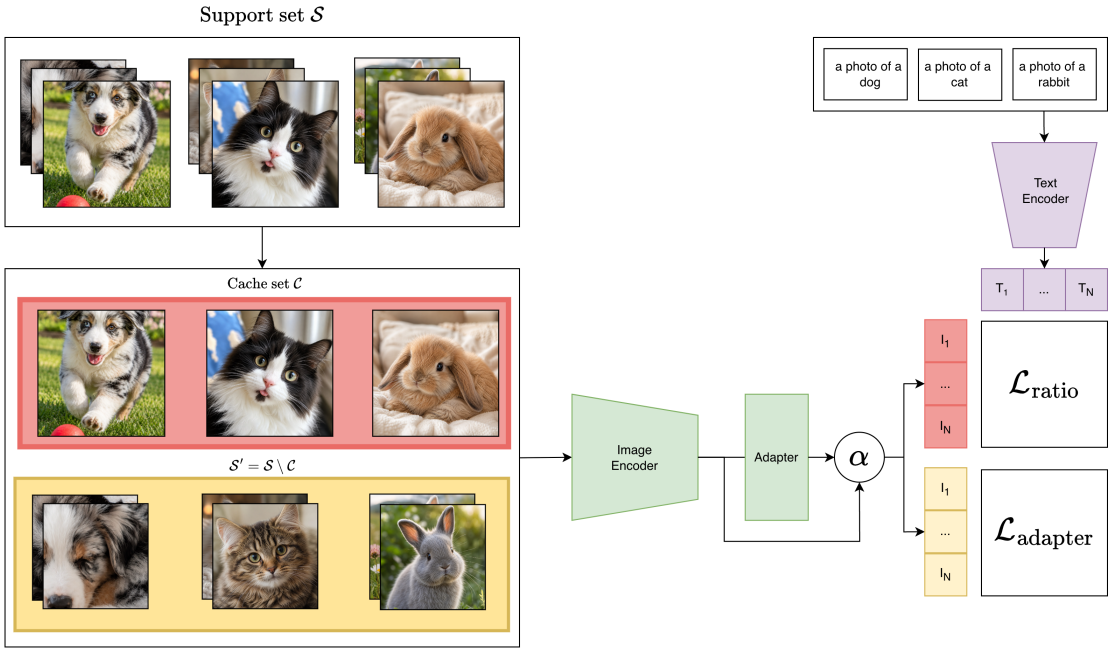


Figure 3.2: Overview of HOSO-Adapter. The frozen CLIP image encoder produces the base visual feature, while a trainable adapter produces an adapted feature. A learnable blending ratio α combines these two features before classification with fixed CLIP text prototypes. Adapter parameters update on the reduced support set $\mathcal{S}'$, while the hold-one-shot-out cache $\mathcal{C}$ updates only the blending-ratio parameter.

This chapter makes three contributions:

1. It introduces Hold-One-Shot-Out as a validation-free method for learning the blending ratio in adapter-based CLIP models.

2. It defines **HOSO-Adapter**, a CLIP-Adapter variant that applies HOSO to the blending ratio and that uses the same architecture and hyperparameters as CLIP-Adapter.

3. It provides empirical and ablation studies that compare HOSO-Adapter with related methods that also use a blending ratio, and that analyse how the ratio behaves during training.

The rest of the chapter proceeds as follows. Section 3.2 summarises related methods that adapt CLIP with a learnable blending ratio, and the section links back to the broader survey in Chapter 2. Section 3.3 presents the formulation of CLIP and adapter-based fine-tuning. Section 3.6 outlines datasets, backbones and training details. Section 3.7 describes the HOSO-Adapter method. Section 3.8 presents the main results and ablations. Section 3.9 discusses overfitting behaviour and the role

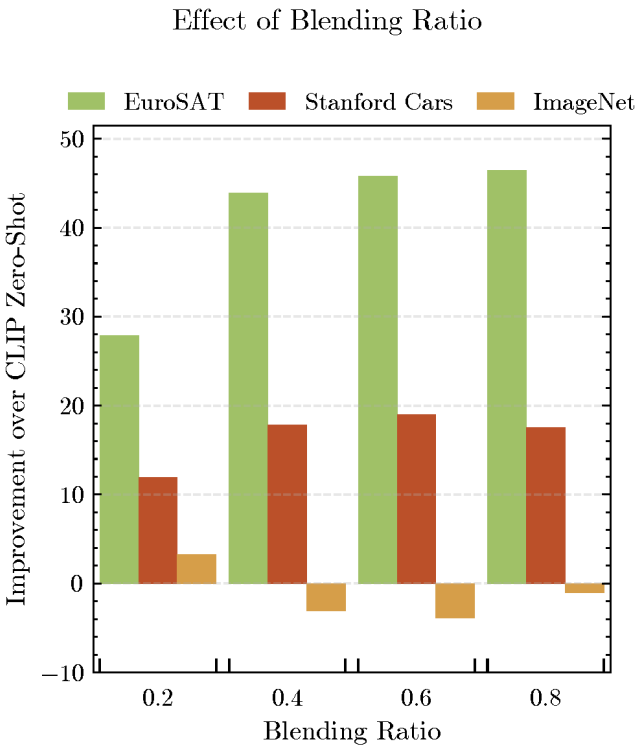


Figure 3.3: Effect of the blending ratio on performance. Each dataset has its own balance between the CLIP prior and the adapted features. Fine-grained datasets such as Stanford Cars benefit from higher values of α, while ImageNet favours lower values that place more weight on the CLIP prior. These results motivate a data-driven and validation-free strategy for the blending ratio.

of the blending ratio. Section 3.14 concludes the chapter and connects to Chapter 4, which explores the use of language models for side information.

3.2 Related Work

Chapter 2 presents a broad review of the few-shot adaptation of vision–language models. In this section, we focus on methods that adapt CLIP by explicitly blending prior and adapted features.

Self-supervised Vision-Language Adapter (SVL-Adapter) [38] estimates α from a measure of confidence of CLIP on the few-shot training set. The method calculates an average zero-shot confidence over all training images of a dataset. This average confidence, denoted by λ, acts as a proxy for zero-shot performance on the target task. SVL-Adapter then sets $\alpha = 1 - \lambda$, so that datasets with low CLIP confidence receive a larger contribution from the adapter.

PathCLIP [39] adapts CLIP to a pathology domain. PathCLIP uses Residual Feature Refinement as a light adapter and learns a blending ratio that changes during training. The update uses two signals. The first is Dual-view Vision Contrastive (DVC), which measures the cosine similarity between features of a weakly augmented

image and those of a strongly augmented image. The second is classification accuracy on mini-batches. These two signals combine to form a target ratio at each training step. A moving average then updates the internal ratio, thereby balancing the CLIP encoder and the adapter.

Both SVL-Adapter and PathCLIP show that a data-dependent blending ratio can improve performance. However, neither work evaluates its blending ratio method in isolation. In addition, PathCLIP focuses on a single pathology domain. In this chapter, we address these gaps by reimplementing both methods within a common CLIP-Adapter framework. The next section defines the notation used by the reimplementations and by HOSO-Adapter.

3.3 Preliminaries

Vision-Language Models, such as CLIP [1], learn a joint space for images and text. CLIP contains a vision encoder $f_\theta(\cdot)$ and a text encoder $f_\phi(\cdot)$. These encoders produce ℓ_2-normalised embeddings v and t for an input image and text, respectively. During training, images and text from the same pairs are learned to be close in the embedding space, and disjoint pairs are far apart. This is the contrastive objective described in more detail in Chapter 2.

For a downstream classification task, CLIP predicts labels without any further training on the images of the target dataset. The method builds a set of natural-language prompts for each class and encodes them with the text encoder. Given C categories and an ensemble of N prompts for each class $\{\{T_{n,c}\}_{n=1}^{N}\}_{c=1}^{C}$, the classifier forms a prototype per class as

$$t_c = \frac{1}{N} \sum_{n=1}^{N} f_\phi(T_{n,c}). \tag{3.1}$$

For an input image x, the visual embedding is $v = f_\theta(x)$. The classifier then computes the probability of each class c through a softmax over cosine similarities,

$$\hat{y}_c = \frac{\exp(v \cdot t_c/\tau)}{\sum_{i=1}^{C} \exp(v \cdot t_i/\tau)}, \tag{3.2}$$

where τ is a temperature parameter learnt during pre-training. The dot product $v \cdot t_c$ equals cosine similarity because the embeddings are normalised.

In a few-shot regime, a method has access to a small labelled support set $\mathcal{S} = \{(x^{(m)}, y^{(m)})\}_{m=1}^{M}$ with K examples per class, that is, $M = K \times C$ with $K \in \{1, 2, 4, 8, 16\}$. Each label $y^{(m)} \in \{0, 1\}^C$ is a one-hot vector that denotes the class of image $x^{(m)}$. The objective is to adapt the pre-trained model to the target

task using $\mathcal{S}$, and to preserve generalisation. Direct fine-tuning of all CLIP parameters is impractical in this regime and often leads to overfitting, so light adaptation mechanisms are more common.

Adapter methods add small trainable modules after the frozen CLIP backbone. Let the original visual feature be $v = f_\theta(x)$. An adapter module with parameters ψ transforms this feature as

$$v' = v + g_\psi(v), \tag{3.3}$$

where $g_\psi(\cdot)$ is a light function, often a bottleneck MLP or a residual linear layer. Training updates only ψ while keeping θ and ϕ fixed, which reduces the number of trainable parameters and helps avoid catastrophic forgetting [59]. The same frozen CLIP backbone can then be paired with a small adapter for each task.

3.4 SVL-Adapter Reimplementation

SVL-Adapter [38] sets the blending ratio α for each dataset before training begins. The method first evaluates a zero-shot CLIP classifier on the few-shot training images and measures CLIP's confidence on the ground-truth labels. It then averages these confidence scores over all images to obtain λ. A high value of λ means that CLIP already aligns well with the target dataset. A low value of λ indicates a domain gap or a more challenging task.

The blending ratio is given by $\alpha = 1 - \lambda$. Datasets with high CLIP confidence use a smaller contribution from the adapter and stay close to the zero-shot classifier. Datasets with low CLIP confidence rely more on adaptation. A key characteristic is that this estimate of α remains fixed during training.

In this chapter, the SVL-Adapter reimplementation follows the same design choices as the PathCLIP reimplementation in the next subsection. We freeze the CLIP encoders and train only a bottleneck multi-layer perceptron (MLP) adapter on visual features. Adapter features and base CLIP features are blended using the fixed value of α predicted from λ.

3.5 PathCLIP Reimplementation

The PathCLIP method [39] learns a blending ratio during training rather than fixing it at the start. The reimplementation in this chapter adapts PathCLIP to the CLIP-Adapter setting and to the standard few-shot benchmarks described in Chapter 2.

The model loads a pre-trained CLIP backbone and freezes both the visual and text encoders. A lightweight MLP adapter with two linear layers and a bottleneck transforms the visual features. The adapter output and the base features then blend

in feature space as

$$\text{image}_{\text{fused}} = \alpha \cdot \text{adapter}_{\text{out}} + (1 - \alpha) \cdot \text{base}_{\text{features}}. \tag{3.4}$$

The pipeline then normalises image and text features and computes logits with the CLIP logit scale. Training uses cross-entropy loss with single dataset-specific prompt templates, in line with CLIP-Adapter.

The core design of PathCLIP is the update rule for α. At each training step, the method forms two augmented views for each image. A weak view uses horizontal flipping and small translations. A strong view uses RandAugment and a light CTAugment module [41]. The method computes a DVC score as the mean cosine similarity between normalised CLIP features of the two views. It also measures the current acc_{batch}. A target value for α is a weighted sum of the DVC score and $(1 - acc_{batch})$. The update then clips this target to a preset interval $[\alpha_{\min}, \alpha_{\max}]$ and applies an exponential moving average,

$$\alpha \leftarrow (1 - \beta) \cdot \alpha + \beta \cdot \text{target}. \tag{3.5}$$

where the coefficient

$$\beta \in (0, 1)$$

is the *smoothing coefficient* that determines how strongly the update reacts to the new α. All augmentation and DVC computations run on the graphics processing unit. This reimplementation provides a direct comparison between our HOSO-Adapter and a prior method that also adapts the blending ratio online.

3.6 Experimental Setup

As in prior work [33, 35], we evaluate methods across eleven datasets: ImageNet, Caltech101, OxfordPets, StanfordCars, Flowers102, Food101, FGVCAircraft, SUN397, Describable Textures (DTD), EuroSAT and UCF101 [60, 61, 62, 63, 64, 65, 43, 66, 67, 68, 69]. Table 3.1 lists these datasets and the corresponding text prompts used. The datasets include general object recognition (for example, ImageNet and Caltech101), fine-grained categories (for example, StanfordCars, Flowers102, and FGVCAircraft), scene recognition (SUN397), material recognition (DTD), satellite images (EuroSAT), and human actions (UCF101). This variety tests both domain shift and fine-grained discrimination.

For each dataset, a few-shot adapter trains on K examples per class, with $K \in \{2, 4, 8, 16\}$. The evaluation protocol follows the strict few-shot setting of [35], as reviewed in Chapter 2. In particular, the method may not use any held-out

validation set from the target dataset to tune hyperparameters such as the blending ratio. Evaluation uses the test sets and data splits that CLIP-Adapter [33] provides.

Table 3.1: Dataset list and corresponding prompt templates used for evaluation.

Dataset	Prompt Template
ImageNet	`"a photo of a {}."`
Caltech101	`"a photo of a {}."`
OxfordPets	`"a photo of a {}, a type of pet."`
StanfordCars	`"a photo of a {}."`
Flowers102	`"a photo of a {}, a type of flower."`
Food101	`"a photo of {}, a type of food."`
FGVCAircraft	`"a photo of a {}, a type of aircraft."`
SUN397	`"a photo of a {}."`
DTD	`"{} texture."`
EuroSAT	`"a centred satellite photo of {}."`
UCF101	`"a photo of a person doing {}."`

We implement all methods in PyTorch and run experiments on an NVIDIA RTX 4090 graphics card. Unless stated otherwise, HOSO-Adapter and all baselines use a pre-trained CLIP model [1] with a ResNet-50 (RN50) vision backbone [8]. Hyperparameters match those of CLIP-Adapter to ensure a fair comparison. The adapter module is a bottleneck MLP with a reduction factor of four and ReLU activations. Training uses stochastic gradient descent with momentum 0.9, weight decay 0.0005 and an initial learning rate of 0.002, with a cosine annealing schedule. The batch size is 32. Standard data augmentations apply, including random resized cropping to 224×224 pixels and random horizontal flipping.

The blending ratio in HOSO-Adapter is a learnable scalar. Training initialises this scalar such that the corresponding α equals 0.5. The ratio then updates via a separate stochastic gradient descent optimiser that uses the hold-one-shot-out cache, with a learning rate of 0.1. All reported results are averages over three runs, each with a different random seed for few-shot sampling.

3.7 Methodology

HOSO-Adapter trains the adapter on $(K - 1)$ examples per class and uses the remaining example to update the blending ratio. The method extends adapter-based fine-tuning with a learnable blending ratio and a decoupled optimisation scheme. The overview in Figure 3.2 shows how the adapter and blending-ratio updates use separate data partitions.

Given a visual feature $v = f_\theta(x)$ from the frozen CLIP encoder and an adapted feature $v_{\text{adapt}} = g_\psi(v)$ from the adapter, the final image embedding $\hat{v}$ is a linear combination,

$$\hat{v} = (1 - \alpha) \cdot v + \alpha \cdot v_{\text{adapt}}, \tag{3.6}$$

where $\alpha \in [0, 1]$ is the blending ratio. A value of α close to zero keeps the behaviour close to that of the original CLIP encoder. A value close to one relies more on the adapter. The classifier then normalises $\hat{v}$ and computes prediction probabilities as in Equation 3.2.

To ensure stable optimisation and to avoid extreme values, HOSO-Adapter parameterises α through a learnable logit α_{logit},

$$\alpha = \text{sigmoid}(\alpha_{\text{logit}}) \cdot 0.8 + 0.1. \tag{3.7}$$

This transformation keeps α in the interval $[0.1, 0.9]$, so neither the CLIP prior nor the adapter can be fully discarded. During training, the CLIP encoders f_θ and f_ϕ remain fixed, and optimisation updates only the adapter parameters ψ and the scalar logit α_{logit}.

A key challenge is learning both the adapter parameters and the blending ratio without overfitting on the small support set. If both sets of parameters update on the same examples, α can increase to fit noise in the adapter features, and the benefit of the CLIP prior can be lost. HOSO-Adapter addresses this risk with a decoupled optimisation scheme based on the hold-one-shot-out cache.

At the start of training, we build a hold-out cache $\mathcal{C}$ by selecting one image-label pair from each class in the full support set $\mathcal{S}$. These C pairs form a small micro-validation set. We then remove them from the training pool to obtain a reduced support set $\mathcal{S}' = \mathcal{S} \setminus \mathcal{C}$. The method pre-computes the text features $\{t_c\}_{c=1}^{C}$ once, using a single dataset-specific prompt template per class, in line with CLIP-Adapter. The text encoder does not update during training, and the stored prototypes are reused.

We use two distinct optimisers on separate data partitions:

- **Adapter training.** The adapter parameters ψ update to minimise cross-entropy loss on the main support set $\mathcal{S}'$. This allows the adapter to learn task-relevant features from $(K - 1)$ examples per class.
- **Ratio training.** The interpolation logit α_{logit} updates through a separate stochastic gradient descent optimiser that minimises cross-entropy only on the cache $\mathcal{C}$.

This decoupled strategy treats $\mathcal{C}$ as a micro-validation set for the blending ratio. The adapter focuses on fitting the main support set, while the ratio tracks how well

the blended features generalise to unseen examples from the same distribution. In this way, HOSO-Adapter learns both task-specific features and appropriate weights for these features, even when the available data is very limited.

3.8 Results

HOSO-Adapter is compared with two existing approaches that learn a blending ratio for CLIP adaptation (SVL-Adapter and PathCLIP) and with a CLIP-Adapter baseline, in which all other components and hyperparameters remain fixed while only the blending ratio differs. The authors of PathCLIP do not release code or report results on standard few-shot benchmarks. We therefore reimplement PathCLIP for this study (see Section 3.5 for implementation details). SVL-Adapter contains multiple components, including a learnable blending ratio based on CLIP confidence scores and a self-supervised encoder. We include both the results from the full SVL-Adapter and a reimplementation that isolates only the learnable blending-ratio component (see Section 3.4).

We evaluate HOSO-Adapter on eleven benchmarks, two CLIP backbone architectures (ResNet-50 and ViT-B/16), and four few-shot settings, $K \in \{2, 4, 8, 16\}$.

Table 3.2 presents a per-dataset breakdown of accuracy (%) for few-shot adaptation of CLIP with an RN50 backbone. We compare HOSO-Adapter with other methods that use blending ratios. We use the reimplementations of CLIP-Adapter [33] and TIP-Adapter [34] under the strict few-shot setting [35] and include SVL-Adapter [38] that complies with the strict setting. The second SVL-Adapter-style entry isolates the unsupervised ratio-selection rule from SVL-Adapter inside the CLIP-Adapter architecture; it is therefore a CLIP-Adapter baseline with SVL's confidence-based ratio, not the full SVL-Adapter system. The original CLIP-Adapter results, in which the best blending ratio is selected for each test dataset, are included for reference. HOSO-Adapter improves over validation-free CLIP-Adapter on average, even though both methods share the same architecture and hyperparameters. The only difference is the way they set the blending ratio.

To assess scalability, we also evaluate HOSO-Adapter using the ViT-B/16 backbone [8]. As shown in Table 3.5 and Table 3.6, HOSO-Adapter achieves an average accuracy of 80.33% in the 16-shot setting. This value is more than four percentage points higher than the validation-free CLIP-Adapter baseline with fixed $\alpha = 0.2$ and lies within one percentage point of the oracle CLIP-Adapter that selects a separate ratio per dataset via a test-set grid search. Gains are largest on fine-grained and specialised datasets such as DTD, FGVCAircraft and EuroSAT. HOSO-Adapter improves the evaluated validation-free baseline with both tested backbones.

A performance comparison across $K = 2, 4, 8$, and 16 shots is shown in Figure 3.4.

Table 3.2: Transfer learning performance (16-shot, RN50).

Method	ImageNet	Caltech101	OxfordPets	StanfordCars	Flowers102	Food101	FGVCAAircraft	SUN397	DTD	EuroSAT	UCF101	Average
CLIP-Adapter IJCV'23[33]†	63.59	92.49	87.84	74.01	93.90	78.25	32.10	69.55	65.96	84.43	76.76	74.44
Zero-Shot ICML'21[1]	60.35	83.81	82.86	55.69	65.94	74.85	17.16	56.80	42.32	37.53	57.47	57.71
CLIP-Adapter IJCV'23[33]	59.02	92.28	84.92	73.49	94.56	73.96	34.19	68.14	65.70	83.24	77.30	73.35
TIP-Adapter ECCV'22[34]	57.81	88.44	81.09	58.83	78.41	72.96	21.96	64.00	54.79	67.90	64.52	64.61
SVL-Adapter BMVC'22[38]*	–	90.00	**89.50**	54.00	87.00	78.00	29.70	65.50	63.00	**97.50**	70.50	–
CLIP-Adapter + SVL ratio BMVC'22[38]§	51.40	82.90	71.10	35.00	81.40	38.80	26.40	59.60	53.80	75.10	63.70	58.11
PathCLIP ECCV'24[39]‡	**63.50**	92.70	84.90	**73.80**	94.10	77.50	34.30	**70.30**	66.40	81.90	**79.00**	73.35
HOSO-Adapter (ours)	62.93	**93.03**	89.47	**73.80**	**95.07**	**80.93**	**34.60**	69.83	**66.77**	83.27	78.03	**75.25**

† Original CLIP-Adapter paper results; blending ratio chosen per dataset. Included for reference, not directly comparable.

* Estimated blending ratio version of SVL-Adapter. Results from their paper, which did not include ImageNet.

§ CLIP-Adapter architecture with SVL-Adapter's confidence-based blending-ratio selection; the self-supervised SVL encoder is not used.

‡ The PathCLIP authors did not evaluate transfer-learning performance; we reimplement their Dual-view Vision Contrastive method to learn the blending ratio.

All three methods in the figure share the same architecture; the only difference lies in the way they set the blending ratio. HOSO-Adapter (green) consistently improves over the fixed-ratio CLIP-Adapter baseline (yellow), with higher average accuracy at $K = 2$, $K = 8$ and $K = 16$. The advantage grows as the number of shots increases. At $K = 16$, for example, HOSO-Adapter outperforms the baseline on fine-grained datasets such as Food101 and OxfordPets, as well as on general-purpose benchmarks such as ImageNet. At both $K = 8$ and $K = 16$, HOSO-Adapter reaches or exceeds the performance of the CLIP-Adapter oracle (grey), which uses a blending ratio tuned on the test set.

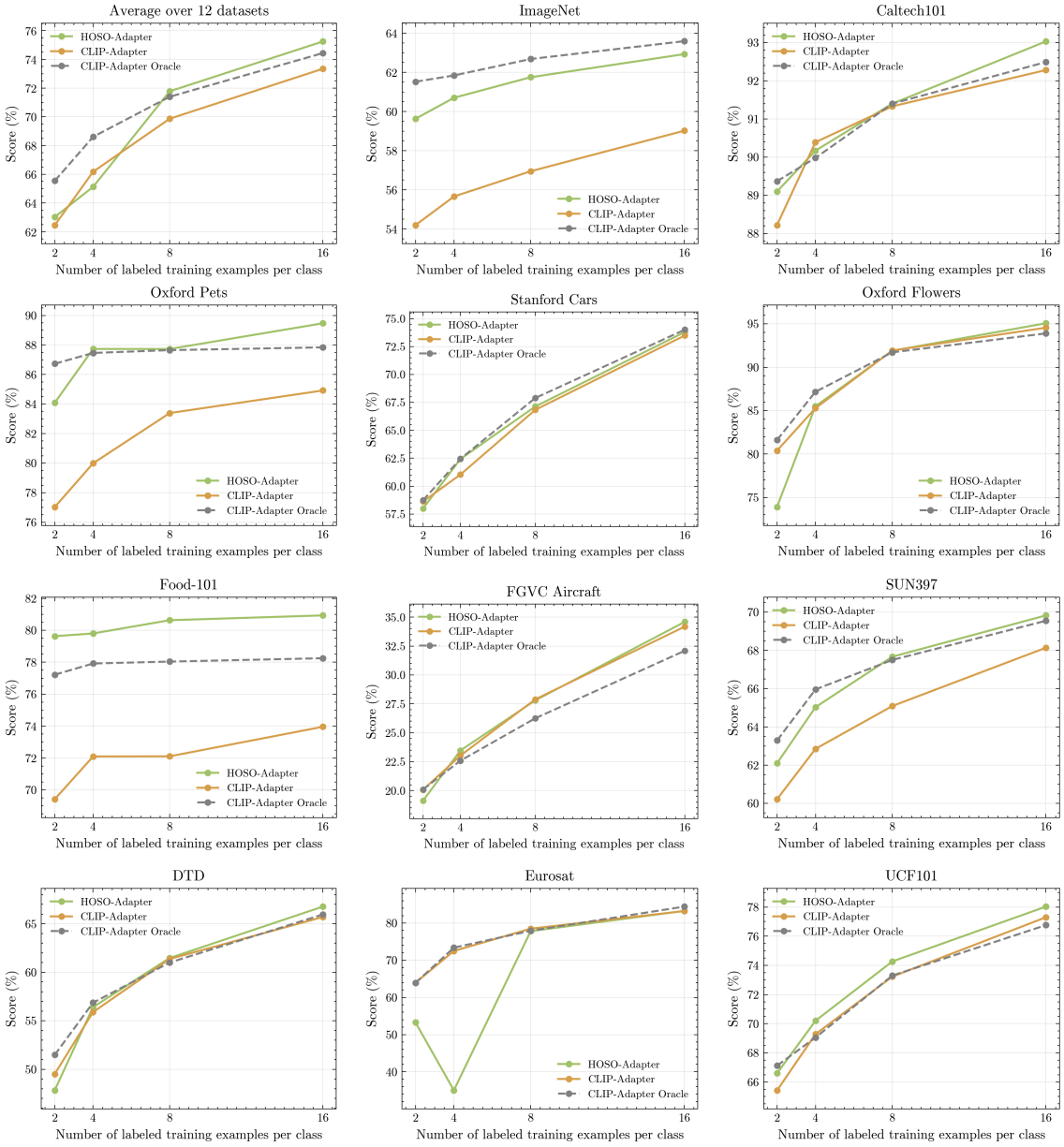


Figure 3.4: Performance of HOSO-Adapter across few-shot settings ($K = 2, 4, 8, 16$) on the RN50 backbone, with averaged results over eleven datasets. The oracle CLIP-Adapter baseline uses a blending ratio selected by grid search on the test set. It serves as a reference and does not follow the strict validation-free protocol.

3.9 Discussion

The design of HOSO-Adapter builds on two empirical observations. The first is the strong correlation between the one-example-per-class zero-shot probe and full test-set accuracy in Figure 3.1. A single example per class provides a compact but informative proxy for dataset difficulty. The second is the clear dependence of performance on the blending ratio, as shown in Figure 3.3. Different datasets reach their best accuracy at different ratios.

These observations frame the learning of the blending ratio as a differentiable search for an effective zero-shot classifier. With the CLIP backbone and text prototypes frozen, the cross-entropy loss on the cache $\mathcal{C}$ directly measures alignment between the blended image embedding $\hat{v}$ and the fixed text prototypes $\{t_c\}$. The gradient with respect to α_{logit} therefore points in the direction that improves classification accuracy on $\mathcal{C}$. Through many iterations, the optimisation process moves α towards a value that performs well on the micro-validation set and, by extension, on the full test set.

The second role of HOSO is to treat the blending ratio as a dynamic regulariser. Figures 3.6 and 3.5 compare HOSO-Adapter with a naive baseline that learns both adapter parameters and the ratio on the full support set. In the naive setting, α tends to increase steadily during training. Predictions then rely increasingly on the adapter, which has many parameters and fits the few training examples too closely. This behaviour leads to large gaps between training and test accuracy.

In contrast, to prevent overfitting, HOSO updates α on the cache $\mathcal{C}$, while the adapter updates on $\mathcal{S}'$. If the adapter begins to overfit, the blended features perform poorly on the cache, and gradients reduce α. The optimisation then shifts weight back towards the CLIP prior. Across datasets, HOSO-Adapter learns more moderate values of α and shows smaller gaps between training and test accuracy (see also Figure 3.7). In this way, HOSO prevents the blending ratio from amplifying overfitting and instead uses it to preserve generalisation.

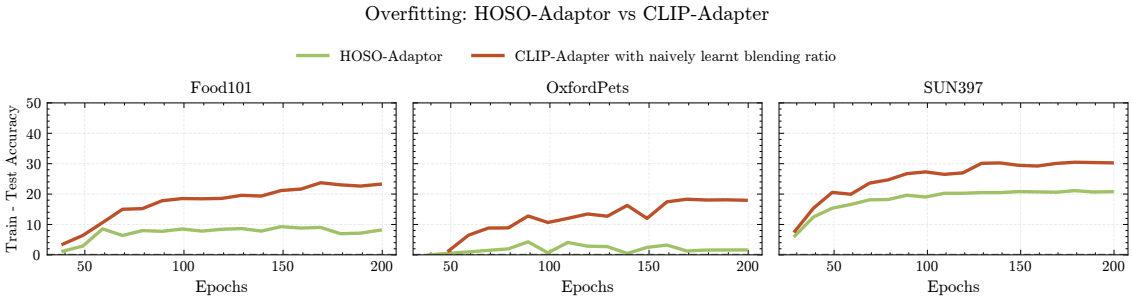


Figure 3.5: Three example datasets that show a consistent trend: *hold-one-shot-out* (green) overfits less compared with a blending ratio that learns on the full support set (red). See Figure 3.8 for results across all datasets.

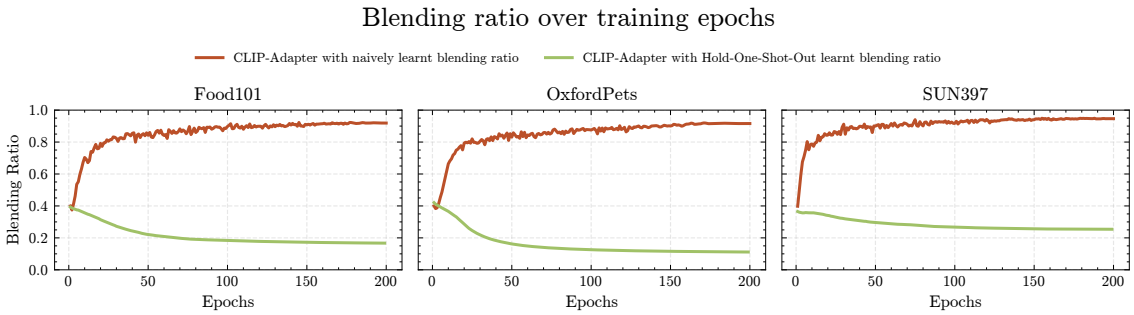


Figure 3.6: Three example datasets that show a consistent trend: *hold-one-shot-out* reduces the blending ratio (green) compared with a naively learnt blending ratio that can overfit on the limited few-shot cases (red). See Figure 3.7 for traces on all datasets.

Figure 3.4 highlights another effect. At $K = 8$ and $K = 16$, HOSO-Adapter reaches or slightly exceeds the oracle CLIP-Adapter baseline that uses a fixed ratio chosen by test-set grid search. The backbone and adapter architecture are identical in these two methods, so the difference arises only from the training of the ratio. A fixed ratio that is optimal at the end of training may still be suboptimal during earlier stages, when the adapter output is noisy. HOSO-Adapter updates the ratio throughout training and relies on feedback from the cache. In early epochs, the ratio can remain low, reducing the influence of unstable adapter features. Later in training, it can increase as the adapter becomes more useful. This schedule appears to benefit both fine-grained and general datasets.

3.10 Detailed Results

Table 3.3: ImageNet accuracy with RN50 backbone. Average of three runs.

Method	2-shot	4-shot	8-shot	16-shot	Average
CLIP-Adapter	59.63	60.67	61.67	63.07	61.26
HOSO-Adapter	59.63	60.70	61.75	62.93	61.25

Table 3.3 shows that HOSO-Adapter and CLIP-Adapter reach very similar average accuracy on ImageNet with RN50 across all shot counts. Both methods improve as the number of shots increases, and the difference between their averages is 0.01 percentage points. One important difference in protocol should be noted; CLIP-Adapter uses the test set to run a grid search over blending ratios for ImageNet, while HOSO-Adapter learns the ratio on the 1-shot cache under the strict few-shot setting.

Table 3.4 gives a broader view of ten datasets and four shot counts. With RN50, HOSO-Adapter improves average accuracy from 63.03% at $K = 2$ to 75.25% at

$K = 16$. EuroSAT, Flowers102, DTD and StanfordCars show the largest gains, with improvements of +30.0, +21.17, +18.94 and +15.80 percentage points, respectively. Food101 and ImageNet change more modestly, which suggests that CLIP already aligns well with these datasets. FGVCAircraft remains the most difficult dataset throughout, but still benefits from additional shots.

Table 3.4: Main results with RN50 backbone, average over three runs. Performance of HOSO-Adapter across different few-shot settings.

Method	Caltech101	DTD	EuroSAT	FGVC	Food101	ImageNet	Flowers	Pets	Cars	SUN397	UCF101	Average
HOSO-Adapter (2-shot)	89.10	47.83	53.27	19.13	79.63	59.63	73.90	84.10	58.00	62.10	66.60	63.03
HOSO-Adapter (4-shot)	90.17	56.40	35.00	23.47	79.80	60.70	85.50	87.73	62.43	65.03	70.20	65.13
HOSO-Adapter (8-shot)	91.40	61.47	77.83	27.80	80.63	61.75	91.90	87.73	67.13	67.67	74.27	71.78
HOSO-Adapter (16-shot)	93.03	66.77	83.27	34.60	80.93	62.93	95.07	89.47	73.80	69.83	78.03	75.25

Table 3.5 shows a similar pattern for ViT-B/16. The mean accuracy of HOSO-Adapter increases from 69.76% at $K = 2$ to 80.33% at $K = 16$. EuroSAT and DTD again show large gains, of +26.07 and +20.27 percentage points. Fine-grained datasets such as FGVCAircraft and StanfordCars also benefit from more shots. Datasets with high zero-shot performance of CLIP, including Food101 and Caltech101, stay strong throughout and change only slowly with K. EuroSAT at $K = 4$ underperforms its $K = 2$ result. This effect appears only for this dataset and remains stable across random seeds. We did not identify an obvious coding or data issue, but the current experiments do not determine the cause of the non-monotonic result.

Table 3.5: Main results with ViT-B/16 backbone, averaged over three runs. Performance of HOSO-Adapter across different few-shot settings.

Method	Caltech101	DTD	EuroSAT	FGVC	Food101	ImageNet	Flowers	Pets	Cars	SUN397	UCF101	Average
HOSO-Adapter (2-shot)	94.20	50.40	59.23	27.73	88.20	67.53	82.07	90.00	67.73	67.03	73.27	69.76
HOSO-Adapter (4-shot)	94.67	59.57	45.10	32.57	87.97	68.67	89.43	91.63	70.90	70.13	77.10	71.61
HOSO-Adapter (8-shot)	94.83	65.40	79.87	37.47	88.90	69.83	95.17	91.57	75.37	72.60	80.73	77.43
HOSO-Adapter (16-shot)	95.40	70.67	85.30	43.23	88.97	70.93	97.23	92.27	81.50	74.67	83.43	80.33

Table 3.6 summarises the 16-shot comparison on ViT-B/16. HOSO-Adapter reaches an average accuracy of 80.33%, compared with 75.82% for CLIP-Adapter with fixed $\alpha = 0.2$ and 81.07% for oracle CLIP-Adapter with a per-dataset ratio. Together with the trend in Table 3.5, this result supports the claim that HOSO-Adapter offers strong 16-shot performance without any test-set hyperparameter search.

3.11 Ablation Study

Table 3.7 reports ablation results averaged across ten datasets for RN50 with $K = 16$. The study covers three design choices: decoupled optimisation of the adapter and ratio, removal of the 1-shot cache from the adapter training set, and cache size.

Table 3.6: ViT-B/16 16-shot performance comparison. CLIP-Adapter results are shown using a fixed blending ratio ($\alpha = 0.2$) and the best-performing ratio selected per dataset. HOSO-Adapter results are averaged over three runs.

Method	Caltech101	DTD	EuroSAT	FGVCAircraft	Food101	ImageNet	Flowers102	OxfordPets	StanfordCars	SUN397	UCF101	Average
CLIP-Adapter ($\alpha = 0.2$)	94.90	59.70	70.50	34.10	89.10	**71.50**	93.10	92.60	73.90	74.20	80.40	75.82
CLIP-Adapter (best α)[†]	**95.90**	**71.70**	**85.80**	**45.80**	**89.30**	**71.50**	**97.40**	**92.70**	**82.10**	**75.60**	**84.00**	**81.07**
HOSO-Adapter (ours)	95.40	70.67	85.30	43.23	88.97	70.93	97.23	92.27	81.50	74.67	83.43	80.33

[†] Best blending ratio from the set {0.2, 0.4, 0.6, 0.7, 0.8} selected independently per dataset using the test set.

The reference method in Table 3.7 is HOSO-Adapter as described in Section 3.7. The first variant, *HOSO-Adapter without decoupled training*, updates both the adapter and the ratio on the same support set. Accuracy drops from 76.43% to 73.02%, which shows that simultaneous optimisation harms generalisation. In this setting, the blending ratio can grow to favour the adapter even when this choice hurts performance on unseen data. The second variant, *HOSO-Adapter without removing the 1-shot cache from the support set*, keeps the cache examples in the adapter training pool. This change reduces average accuracy to 73.35%. The result suggests that the adapter then overfits to cache examples and that the ratio estimate becomes less reliable.

The third and fourth variants change the cache size. *HOSO-Adapter with two shots for the blending ratio* uses two cache examples per class and trains the adapter on fourteen shots per class. This variant achieves 76.04%, close to the baseline. A larger cache of 8 shots per class halves the adapter training data and reduces accuracy to 73.68%. These results point to a trade-off between accurate estimation of the ratio and the availability of a rich training set for the adapter. A single held-out shot per class appears to offer a good balance.

Table 3.7: Ablation study on RN50 with 16-shot. We report average accuracy across ten datasets. Per-dataset values appear in Table 3.8.

Method	Average
HOSO-Adapter	**76.43**
HOSO-Adapter without decoupled training	73.02
HOSO-Adapter without removing 1-shot from few-shot train set	73.35
HOSO-Adapter with 2-shots for blending ratio	76.04
HOSO-Adapter with 8-shots for blending ratio	73.68

3.11.1 Per-Dataset Ablation Study

Table 3.8 gives per-dataset results for the same ablation. The baseline configuration, which uses one cached sample per class and removes these samples from adapter training, achieves the best overall average of 76.43%. Most datasets benefit from this policy. Keeping cached items in the training set reduces the average by 3.08 percentage points and leads to clear drops on Food101 and OxfordPets. EuroSAT shows a small gain in this case, but the improvement does not offset declines

Table 3.8: Ablation on design choices for HOSO-Adapter (RN50, 16-shot). The baseline updates the ratio at each epoch using a cache of size 1 per class, removing cached samples from the training set.

Method	Caltech101	DTD	EuroSAT	FGVC	Food101	Flowers	Pets	Cars	SUN397	UCF101	Average
Baseline (remove cached samples)	93.0	66.6	83.0	34.8	81.1	95.1	89.5	73.7	69.6	77.9	**76.43**
Keep cache in training set	91.2	63.9	84.2	34.4	73.5	94.2	79.5	70.8	65.5	76.3	73.35
Cache size of two per class	92.6	66.5	82.1	33.9	80.8	94.6	89.0	73.6	69.5	77.8	76.04
Cache size of eight per class	91.7	62.6	80.9	29.3	80.7	92.5	88.0	69.1	67.3	74.7	73.68

elsewhere. A cache of two shots per class is near neutral, while a cache of eight per class lowers performance across most datasets. Overall, a minimal cache, combined with removing cache examples from training, yields the most robust results.

3.11.2 Blending Ratio Over Training

HOSO-Adapter uses the blending ratio as a dynamic regulariser that reacts to overfitting. This analysis tracks α under two regimes: the decoupled HOSO setting and the naive joint-training baseline. In the naive case, the optimiser often increases α over time, because the adapter can fit the limited support set very closely. This shift reduces the influence of the CLIP prior, leading to poor test performance. In the HOSO setting, the ratio updates only on the cache. When the adapter overfits, performance on the cache degrades, and gradients push α down. The update then moves the weight back to the prior. This pattern holds for most datasets. EuroSAT is a partial exception; the present analysis does not establish why it differs.

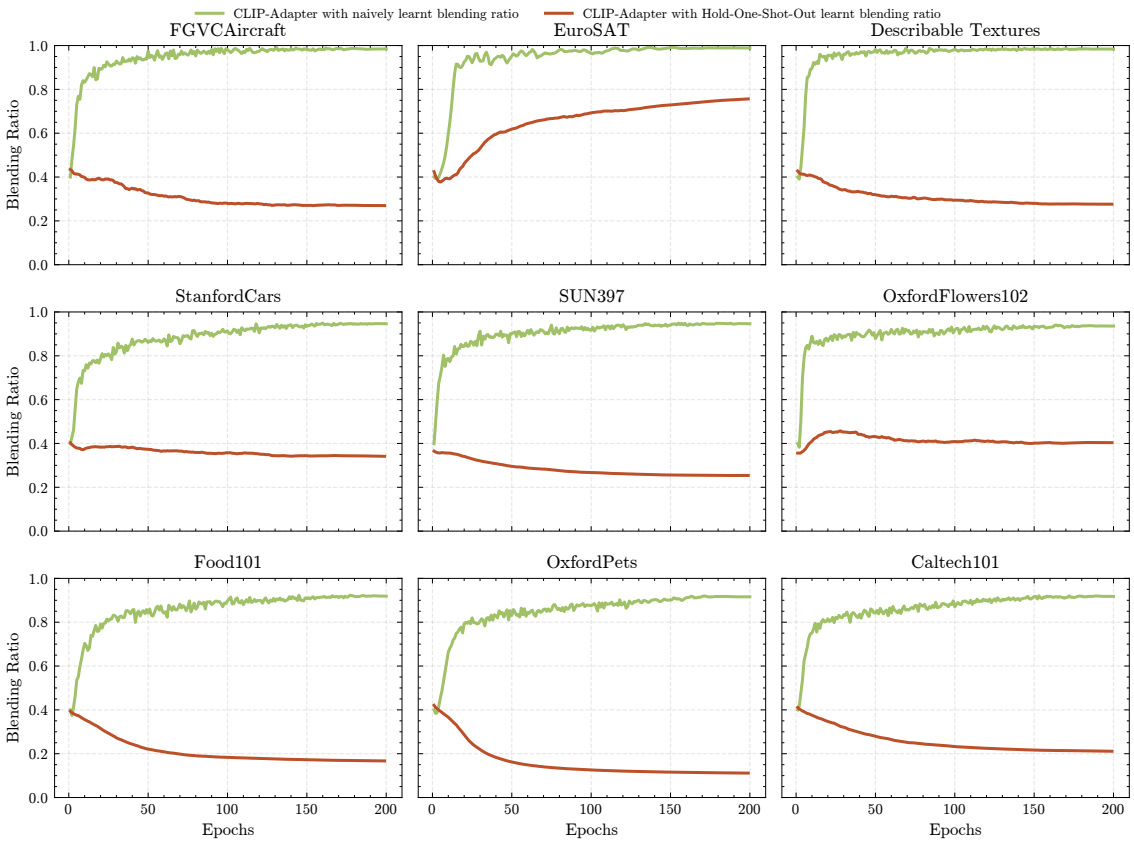


Figure 3.7: Most datasets exhibit a trend: *hold-one-shot-out* reduces the blending ratio (green) compared with a learnt blending ratio that can overfit on the limited few-shot cases (red).

3.11.3 Overfitting Analysis

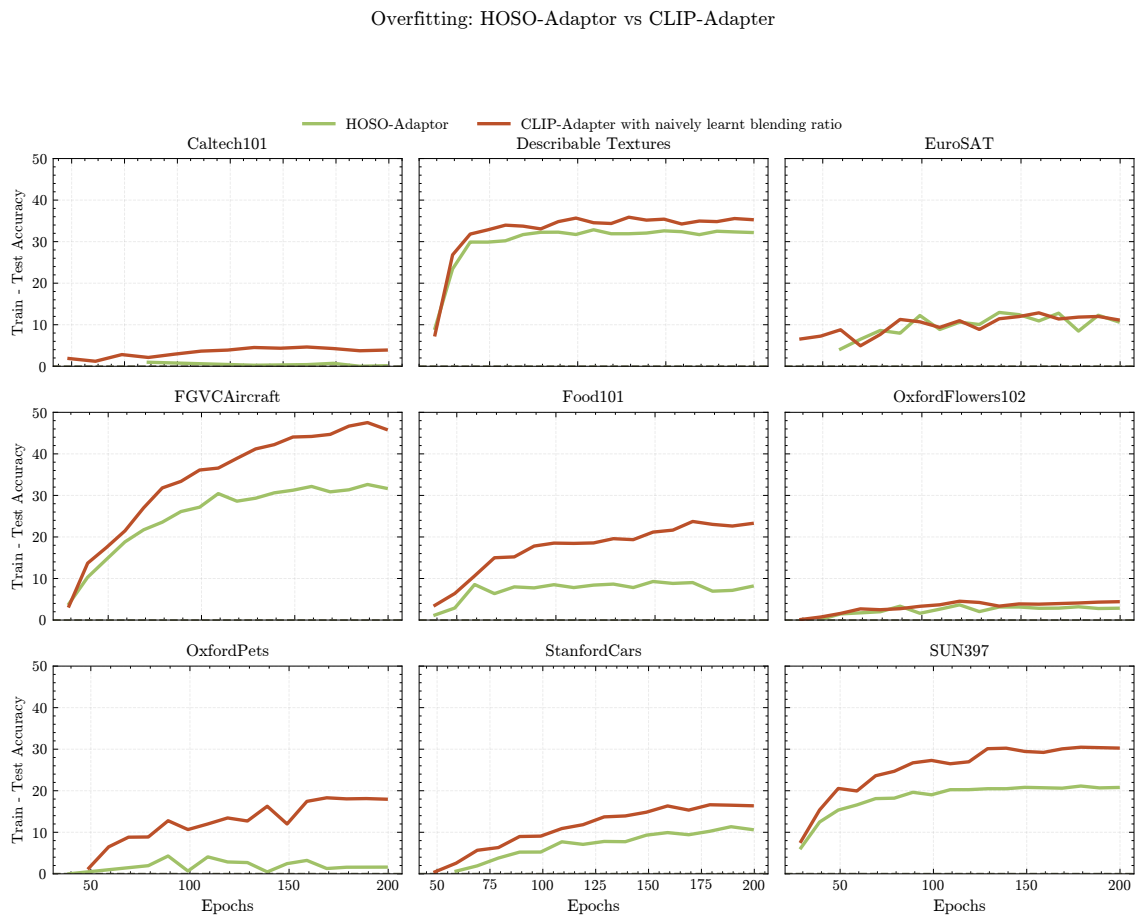


Figure 3.8: All datasets exhibit a consistent trend: *hold-one-shot-out* (green) exhibits less overfitting than a blending ratio learnt directly on the full training set (red).

Figure 3.8 summarises overfitting behaviour for HOSO-Adapter and the naive baseline. The vertical axis shows the gap between training and test accuracy over the course of training. A large gap indicates strong overfitting. Across nine benchmark datasets, HOSO-Adapter maintains smaller gaps than the baseline. The baseline shows gaps of over 40 percentage points on datasets such as DTD, Food101, SUN397, StanfordCars, and FGVCAircraft. HOSO-Adapter keeps these gaps much lower, which supports the view that decoupled optimisation of the ratio reduces overfitting on few-shot data and improves generalisation.

3.11.4 Residual Grid Search

We also study the behaviour of CLIP-Adapter when we vary the fixed blending ratio. With a ViT-B/16 backbone and $K = 16$, average accuracy rises from 74.82% at $\alpha = 0.2$ to a peak of 80.62% at $\alpha = 0.6$, then decreases slightly at $\alpha = 0.8$ (Table 3.9). Fine-grained and distribution-shifted datasets such as DTD, EuroSAT,

FGVCAircraft and StanfordCars benefit most from higher adapter contribution. Tasks with strong CLIP priors or strong scene cues, such as OxfordPets, SUN397 and UCF101, favour moderate values of α around 0.4. Caltech101 and Flowers102 remain stable across a wide range of ratios.

Table 3.9: CLIP-Adapter ViT-B/16 blending search (16-shot). Classification accuracy for different blending ratios (α). A low value of α favours CLIP features, while a high value favours the adapter.

Dataset	$\alpha = 0.2$	$\alpha = 0.4$	$\alpha = 0.6$	$\alpha = 0.8$
Caltech101	94.9	**95.7**	**95.7**	95.3
DTD	59.7	69.7	71.5	**71.6**
EuroSAT	70.5	80.3	84.1	**85.8**
FGVCAircraft	34.1	40.9	44.1	**45.8**
Flowers102	93.1	97.0	**97.3**	96.8
OxfordPets	92.6	**92.7**	92.2	90.7
StanfordCars	73.9	79.7	**81.9**	81.7
SUN397	74.2	**75.6**	75.1	73.7
UCF101	80.4	**84.0**	83.7	83.5
Average	74.82	79.51	**80.62**	80.54

We repeat the grid search for the RN50 backbone, again at $K = 16$ and with $\alpha \in \{0.2, 0.4, 0.6, 0.8\}$ (Table 3.10). Compared with ViT-B/16, RN50's weaker visual prior makes the best ratio more dataset-dependent. Larger blending ratios tend to help again on EuroSAT, FGVCAircraft and StanfordCars, while smaller ratios work well on Food101 and OxfordPets, where CLIP already performs strongly in zero-shot form. The variation across rows shows that a single fixed value does not perform best for every dataset.

Table 3.10: CLIP-Adapter RN50 blending ratio search (16-shot).

Dataset	$\alpha = 0.2$	$\alpha = 0.4$	$\alpha = 0.6$	$\alpha = 0.8$
Caltech101	91.4	**92.7**	92.6	91.8
DTD	59.2	66.1	**67.0**	66.2
EuroSAT	65.4	81.4	83.3	**84.0**
FGVCAircraft	24.5	32.1	35.0	**35.6**
Food101	**81.3**	79.9	77.9	75.7
Flowers102	89.7	94.0	94.7	**94.8**
OxfordPets	**88.6**	87.9	85.1	83.9
StanfordCars	67.6	73.5	**74.7**	73.2
SUN397	69.8	**70.3**	69.4	68.0
UCF101	76.2	78.8	**78.9**	77.9

3.11.5 Ablation Design Choices

Table 3.11 compares three strategies for the blending ratio with ViT-B/16 and $K = 16$. The first uses a fixed value of $\alpha = 0.6$, selected by grid search as the best single ratio across datasets. The second samples α at random from $[0, 1]$ for each dataset. The third is HOSO-Adapter, which learns the ratio from the cache. HOSO-Adapter reaches the highest average accuracy of 80.80%, slightly above the fixed-ratio upper bound and clearly above random sampling. Fixed CLIP-Adapter still leads on some individual datasets, but HOSO-Adapter offers a more robust choice when a practitioner cannot rely on test-set grid search.

Table 3.11: Comparison of different blending ratio strategies for CLIP-Adapter (ViT-B/16, 16-shot). HOSO-Adapter uses a dynamically learned blending ratio.

Dataset	**Fixed** $\alpha = 0.6$	**Random** $\alpha \in [0, 1]$	**HOSO-Adapter**
Caltech101	95.7	95.4	**96.1**
DTD	71.5	67.0	**72.5**
EuroSAT	84.1	79.6	**86.6**
FGVCAircraft	44.1	37.3	**46.4**
Flowers102	**97.3**	94.0	97.1
OxfordPets	**92.2**	88.3	90.4
StanfordCars	**81.9**	69.9	80.4
SUN397	**75.1**	68.2	74.4
UCF101	**83.7**	79.4	83.3
Average	80.62	75.46	**80.80**

Table 3.12: CLIP-Adapter ViT-B/16 feature vs. logit blending (16-shot; top-1 accuracy, %). Both blending methods use a blending ratio of $\alpha = 0.7$.

Dataset	Feature Blend	Logit Blend
Caltech101	**95.9**	94.3
DTD	**71.7**	70.7
EuroSAT	85.1	**85.5**
FGVC Aircraft	45.4	**46.4**
Flowers102	**97.4**	96.9
OxfordPets	**91.6**	90.5
Stanford Cars	82.1	**82.4**
SUN397	**74.4**	72.5
UCF101	**83.7**	82.5
Average	**80.8**	80.2

There are two simple strategies for combining information from the pre-trained

CLIP model and an adapter. The first blends features, and the second blends logits. Table 3.12 compares these strategies for ViT-B/16 at $K = 16$ with $\alpha = 0.7$. Feature blending linearly interpolates the feature vector from the CLIP image encoder $\mathbf{f}_{\text{CLIP}}$ and the adapter output $\mathbf{f}_{\text{adapter}}$, then normalises the result,

$$\mathbf{f}_{\text{blend}} = \alpha \cdot \mathbf{f}_{\text{adapter}} + (1 - \alpha) \cdot \mathbf{f}_{\text{CLIP}}. \tag{3.8}$$

The classifier then computes logits from $\mathbf{f}_{\text{blend}}$. Logit blending instead computes logits from CLIP features ($\mathbf{l}_{\text{CLIP}}$) and from adapter features ($\mathbf{l}_{\text{adapter}}$) and combines them,

$$\mathbf{l}_{\text{final}} = \alpha \cdot \mathbf{l}_{\text{adapter}} + (1 - \alpha) \cdot \mathbf{l}_{\text{CLIP}}. \tag{3.9}$$

Neither approach fully dominates, but feature blending shows a small advantage across several datasets and achieves a slightly higher average accuracy. This observation motivates the focus on feature-space blending in the main experiments.

Table 3.13 studies the learning rate of the optimiser for the blending ratio with RN50 and $K = 16$. The average accuracy is stable across a wide range of learning rates from 0.001 to 0.4, peaking at 0.1. This value also matches per-dataset optima on DTD, EuroSAT and UCF101. We therefore use 0.1 as the default choice for the ratio optimiser in the main experiments.

Table 3.13: Ablation on the learning rate for the blending ratio optimiser. Experiments use RN50 with sixteen shots.

Dataset	**0.001**	**0.01**	**0.1**	**0.4**
DTD	65.5	66.5	**66.7**	66.6
EuroSAT	83.9	83.6	**84.0**	83.0
UCF101	76.2	77.4	**78.1**	77.9
Average	75.20	75.83	**76.27**	75.83

Figure 3.9 further explores cache size with ViT-B/16 and $K = 16$. The horizontal axis shows the ratio between cache size and full shot count. HOSO-Adapter reaches peak average accuracy when the cache contains one shot per class, which is about six percent of the data. Larger caches reduce performance across ten datasets. This behaviour again reflects the trade-off between a stable estimate of the blending ratio and enough training data for the adapter. A single hold-out shot offers the best balance in this setting.

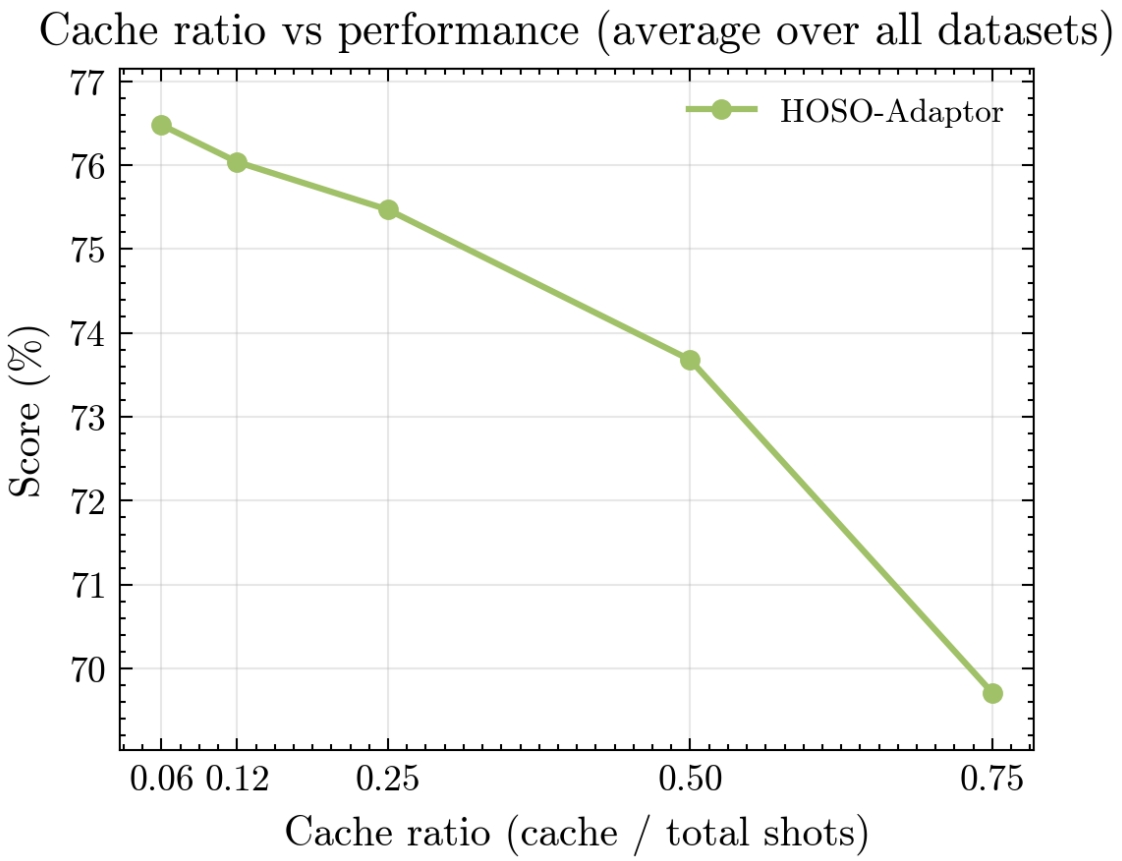


Figure 3.9: A single hold-out shot is optimal. An ablation study of the hold-out cache size in the sixteen-shot setting using ViT-B/16. The average accuracy across 10 datasets is plotted against cache size, expressed as a ratio of total shots. Performance peaks at a cache size of one (ratio ≈ 0.06), which supports the core design choice. Larger cache sizes lead to performance declines, indicating a trade-off between allocating samples to learn the blending ratio and retaining samples for adapter training.

3.12 Additional Investigations

3.12.1 Class-Level Perspective

Figure 3.10 shows per-class accuracy for EuroSAT as a function of the blending ratio. A ratio of zero corresponds to the zero-shot CLIP classifier, while a ratio of one uses only adapter features. The top panel ($K = 1$) exhibits substantial variation across classes and ratios. Classes such as "Residential Buildings" benefit from ratios between 0.7 and 0.9, while "Herbaceous Vegetation Land" peaks near 0.8 despite a low zero-shot baseline. The figure shows that a single ratio is not optimal across all classes in a dataset, particularly in the low-data regime. The bottom panel ($K = 16$) shows higher accuracy overall and reduced sensitivity to the exact ratio. Once the adapter trains on more data, most ratios above 0.4 yield similar, strong results that outperform the zero-shot baseline.

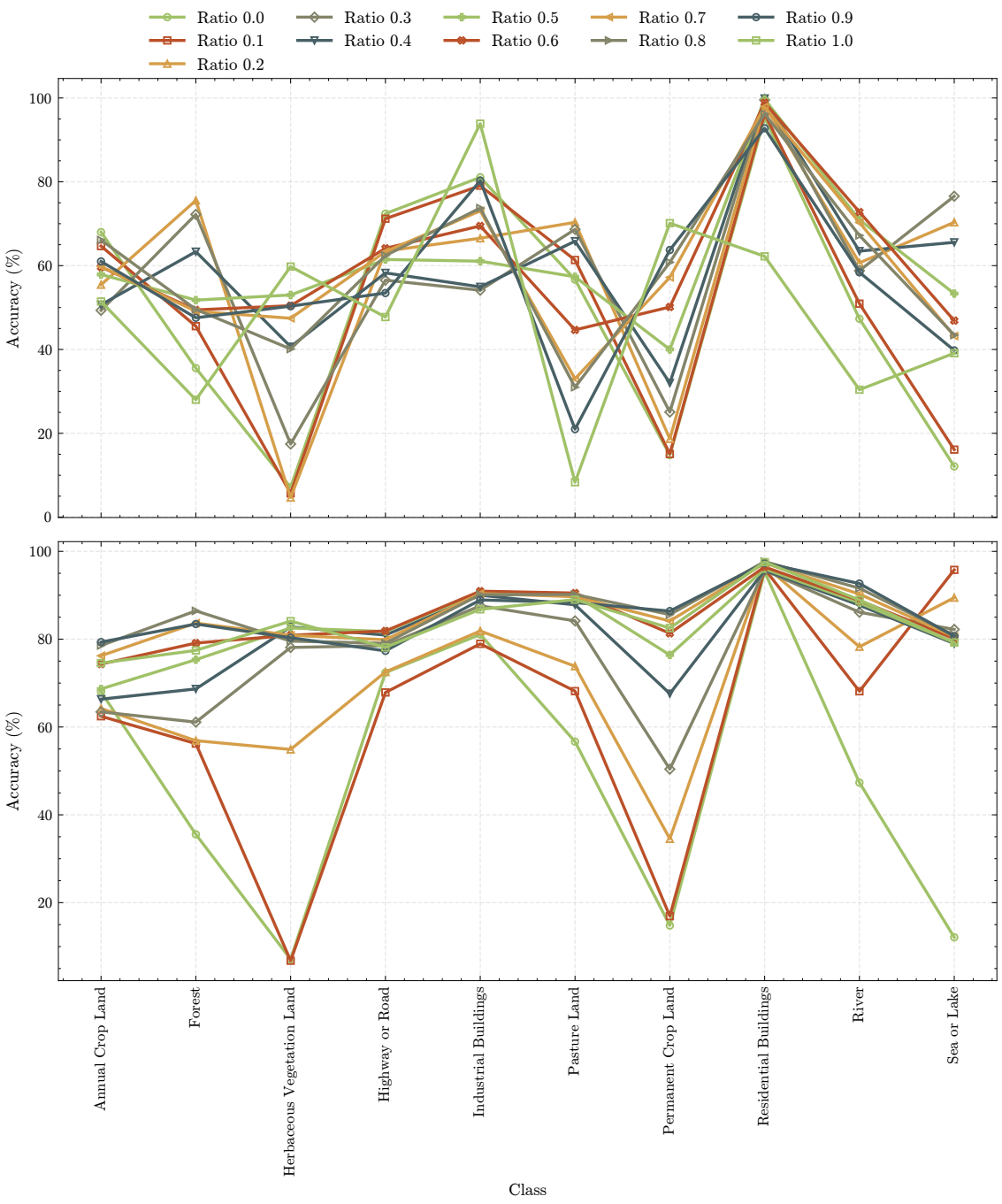


Figure 3.10: Per-class accuracy as a function of class type for different blending ratios (α) on the EuroSAT dataset. The top plot shows the results for the 1-shot setting, while the bottom plot shows the results for the 16-shot setting. Each coloured line represents a fixed blending ratio. The analysis highlights that the optimal ratio varies between classes, particularly in the 1-shot scenario.

Figure 3.11 presents a complementary view. Here, each line traces accuracy for one class as α varies. In the 1-shot case, most classes follow a unimodal curve that starts at the zero-shot baseline $\alpha = 0$, rises to an optimal intermediate ratio and then declines as α approaches one. This pattern suggests that the adapter provides useful task-specific information, but discarding CLIP's general-purpose features is harmful. The optimal ratio depends on the class. For example, "Forest" peaks closer to $\alpha = 0.2$, while "Permanent Crop Land" peaks near $\alpha = 1$. In the sixteen-shot case, curves show a sharper initial rise followed by a broad plateau above ratios of about 0.5. This plateau indicates that once the adapter trains on enough examples, performance becomes robust to a range of higher ratios.

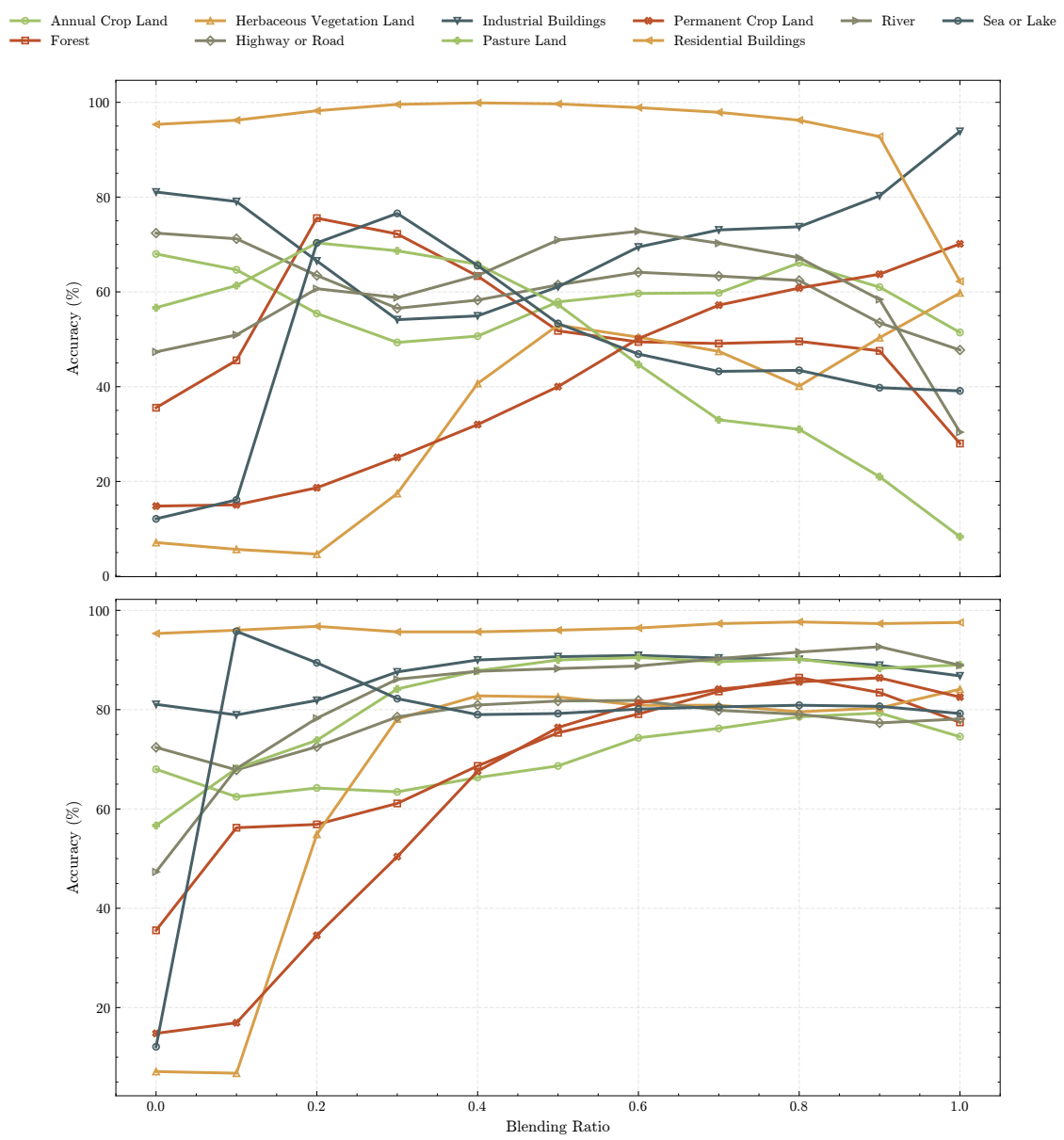


Figure 3.11: Per-class accuracy as a function of the blending ratio (α) on the EuroSAT dataset. The top plot corresponds to the 1-shot setting, and the bottom to the 16-shot setting. Each coloured line represents a specific class. Performance generally improves from the zero-shot baseline ($\alpha = 0.0$) to an optimal intermediate ratio, with the sixteen-shot setting yielding a broader high-performance plateau.

3.12.2 Test Augmentations

Table 3.14 evaluates HOSO-Adapter with on-the-fly data augmentation. Flowers102 and Caltech101 remain the easiest datasets, while FGVCAircraft, DTD and StanfordCars are more challenging. Average accuracy still increases with the number of shots, from 62.83% at $K = 2$ to 76.05% at $K = 16$. EuroSAT again shows a non-monotonic pattern with $K = 4$ underperforming $K = 2$. As in the earlier analysis, this behaviour appears only for EuroSAT and remains consistent across random seeds. The cause is not established by these experiments.

Table 3.14: Performance of HOSO-Adapter with data augmentation. Results use an RN50 backbone.

Shots	Caltech101	DTD	EuroSAT	FGVCAircraft	Food101	Flowers102	OxfordPets	StanfordCars	SUN397	UCF101	Average
2-shots	88.5	47.7	50.7	19.1	78.8	73.6	84.7	57.5	62.0	65.7	62.83
4-shots	89.4	56.4	32.6	22.9	79.4	85.1	87.1	61.7	65.2	69.5	64.93
8-shots	91.2	62.1	77.6	28.2	80.2	91.4	86.9	66.7	67.5	74.0	72.58
16-shots	**92.5**	**66.3**	**81.6**	**34.9**	**80.7**	**94.6**	**89.2**	**73.3**	**69.4**	**78.0**	**76.05**

This experiment tests whether additional augmentation can bring HOSO-Adapter closer to PathCLIP, which also relies on strong augmentations. For each training image, the system forms an additional augmented view. Weak and strong augmentations apply with some probability, using policies such as RandAugment and RandomErasing. The model processes both original and augmented views in a single forward pass, doubling the number of training views.

Comparing HOSO-Adapter with and without these augmentations yields small differences of -0.54, -0.64, -0.20 and -0.43 percentage points for $K = 2, 4, 8, 16$, respectively. Since augmentation slightly lowers performance in this setting and adds complexity, we do not include it in the main method.

3.13 Practical Interpretation of Ratio Selection

The blending ratio is a small but consequential part of few-shot adaptation. It controls one scalar degree of freedom: how much of the final prediction comes from the transferable CLIP representation and how much comes from the task-specific adapter. This is narrower than general hyperparameter selection. The cache is not used to choose an architecture, learning rate, augmentation policy, or stopping rule; it estimates only this balance. Restricting the search in this way is important because one example per class cannot support a reliable high-dimensional model-selection procedure.

The class-level analyses explain why a fixed ratio is inadequate. In the 1-shot EuroSAT results, different classes peak at substantially different values of α, while the 16-shot curves form a broader plateau. The useful contribution of the adapter

therefore depends both on the class and on how much labelled evidence it has received. HOSO does not estimate a separate ratio for every class, which would overfit the cache. Instead, repeated cache updates provide a dataset-level compromise that changes during training as the adapter becomes more reliable. This accounts for the learned trajectories in Figure 3.7: the method is estimating the changing relative value of two representations, rather than recovering a universal constant.

Holding out one example per class also separates two roles that otherwise compete for the same scarce data. The adapter learns task-specific features from the training subset, while the cache evaluates whether those features improve classification when combined with the frozen CLIP prior. The resulting estimate is class-balanced by construction, so a frequent class cannot dominate the ratio solely through sample count. The cost is that HOSO requires at least two labelled examples per class and removes one example from adapter training. Its advantage should therefore be interpreted within the $K \geq 2$ validation-free setting studied here, not as a solution for strict 1-shot adaptation.

The oracle comparison provides a final practical lesson. A fixed ratio selected by test-set grid search is not necessarily stronger than a ratio learned jointly with the state of the adapter. At higher shot counts, HOSO matches or slightly exceeds that oracle baseline because its ratio can evolve during optimisation. The comparison does not make test labels legitimate for tuning; it shows that dynamic estimation from a small, predefined cache can recover the useful balance without them.

3.14 Chapter Summary

In this chapter we study how to choose a blending ratio between CLIP features and adapter features in a strict few-shot, validation-free setting. The proposed HOSO-Adapter method holds out one labelled example per class in a cache, and uses this cache to learn the blending ratio, while the adapter trains on the remaining examples. The design keeps the CLIP encoders fixed, uses a simple adapter architecture and requires no additional hyperparameter search beyond the 1-shot cache.

Experiments across eleven datasets and two backbones show that HOSO-Adapter outperforms validation-free CLIP-Adapter and reimplementations of SVL-Adapter and PathCLIP. At higher shot counts, HOSO-Adapter even matches or slightly exceeds an oracle CLIP-Adapter that uses test-set grid search to choose a fixed ratio. Ablation studies and analyses of training dynamics support the view that the HOSO-trained blending ratio serves as a dynamic regulariser, limiting overfitting and preserving the benefits of the CLIP prior.

Chapter 4 examines a different source of low-shot improvement: LLM-generated side information for class prototypes and adapters.

Chapter 4

LLM-Generated Side Information for Low-Shot Learning

4.1 Introduction

This chapter evaluates class-level semantic prototypes built from labels, taxonomies, natural-language descriptions, and attributes in zero- and few-shot recognition. A semantic prototype represents each class with a vector derived from a natural-language description, a class label, a taxonomy entry, or an attribute representation.

Research Question 2 asks whether low-shot image classification accuracy improves when a Large Language Model generates the side information. Side information denotes any class-level textual signal beyond raw class labels, including attribute lists, short definitions, captions, and visual descriptions. The study compares LLM-generated side information with ground-truth attribute vectors derived from manual annotation. The analysis also evaluates whether an LLM should generate attribute-style facts directly or instead generate descriptions that a text encoder embeds into a semantic space.

We test whether LLM side information works better as visually descriptive text encoded by a vision-language model than as a direct replacement for expert attributes. This hypothesis links the embedding analysis, MiniImageNet experiments, CUB attribute experiments, and VDT-Adapter results: each experiment changes either the source of side information, the encoder that maps it to a prototype, or the classifier that consumes the prototype.

MiniImageNet and CUB provide complementary evaluation settings. MiniImageNet is a 100-class subset of ImageNet with 600 images per class at 84×84 resolution and is widely used for few-shot recognition. Ground-truth supervision contains only image-level class labels, with no human-annotated attributes; class names and the inherited WordNet taxonomy provide weak semantic structure. CUB contains

200 bird species and 11,788 images with fine-grained annotations, including 312 binary attributes per image, part locations, and bounding boxes. Class-level attribute prototypes are obtained by aggregating image-level attributes and serve as strong ground-truth side information. MiniImageNet tests generic object recognition based on label semantics, whereas CUB tests whether LLM-generated side information can substitute for human-defined attributes in a fine-grained domain.

All methods produce a bank of class prototypes in a common format:

$$\{\text{classes} : [c_i]_{i=1}^{100}, \quad \text{semantic_protos} : \mathbf{E} \in \mathbb{R}^{100\times d}\},$$

where the i-th row of $\mathbf{E}$ represents the semantic prototype for class c_i, and d denotes the embedding dimensionality.

The classifier compares image feature vectors to semantic prototypes at test time and assigns each image to the class with the highest similarity under the chosen metric (typically cosine similarity).

GloVe [13] derives static word embeddings from co-occurrence statistics and provides stable lexical similarity without contextual disambiguation. SentenceTransformer [70, 71] encodes sentence-level semantics through supervision from entailment, contradiction, and neutral pairs, which improves alignment across varied phrasings. PromCSE [16] uses prompt-based contrastive objectives to obtain transferable sentence representations from general corpora. CTP [17] targets dictionary-style inputs and taxonomic structure, favouring definitional precision over open-domain discourse. OpenAI embeddings [72, 73] combine instruction-oriented and retrieval-oriented supervision at scale, producing high-dimensional representations that perform strongly across search, clustering, and classification tasks. CLIP text embeddings [74, 1] align text with visual features through contrastive image-text pre-training, enabling zero-shot transfer. WordNet [29] supplies definitions and taxonomic relations, and ChatGPT [75, 73] supplies generated definitions, captions, and visual descriptions. The experiments vary the text source, embedding model, and prompt construction.

The chapter proceeds in five stages. The background first defines low-shot recognition, semantic side information, and the text embedding families used in the study. We then compare the semantic graphs induced by these embeddings and measure their alignment with WordNet and CLIP reference graphs. The MiniImageNet evaluation measures zero-shot and few-shot recognition with text-derived prototypes. The CUB evaluation tests Zero-Shot Learning with CADA-VAE and discriminative attribute prediction with APN. The final experimental section enhances CLIP with GPT-4-generated visually descriptive text and a lightweight few-shot adapter.

The chapter's text refers to embedding variants by short names used in the ta-

bles and figures. The naming convention follows *Encoder-TextSource*, where the prefix denotes the embedding model (CLIP, OpenAI, ST for SentenceTransformer, PromCSE, or CTP) and the suffix denotes the textual source (LabelText for curated label phrases, WordNetDef for WordNet definitions, WikiText for Wikipedia descriptions, HypernymText for hypernym chains, ChatGPTDef for ChatGPT-generated definitions, VisSent for visual sentences, AllCaptions for aggregated captions, and Taxonomy for taxonomic structure). The main variants are:

- CLIP-LabelText; OpenAI-WordNetDef; OpenAI-LabelText; OpenAI-WikiText.
- OpenAI-WikiTextRefined; OpenAI-HypernymText.
- ST-WordNetDef; ST-LabelText; ST-ChatGPTDef.
- PromCSE-VisSent; PromCSE-AllCaptions; CTP-Taxonomy.

Additional variants based on GloVe and Principal Component Analysis use the corresponding names.

Terminology follows a fixed convention. ChatGPT denotes text generated from the ChatGPT web-interface, which used the GPT-3.5 model, and GPT-4 denotes experiments where the specific model was used via API. VDT denotes visually descriptive text, and CLIP-GPT denotes a CLIP prompt ensemble constructed from GPT-generated visual descriptions.

4.2 Background

4.2.1 The Low-Shot Learning Problem

Standard supervised learning assumes abundant labelled training data for every class. Labelled data are costly and time-consuming to collect in many recognition domains. Low-shot learning addresses this constraint. Zero-Shot Learning (ZSL) requires a model to classify images from classes with no labelled examples during training. Few-shot learning relaxes this constraint by allowing a small number of labelled examples per class for training. In this context, "small" typically means between one and sixteen.

Formally, ZSL defines a set of seen classes $\mathcal{Y}^s$ with labelled training images and a disjoint set of unseen classes $\mathcal{Y}^u$ with no training images. The model classifies test images from $\mathcal{Y}^u$. Generalised ZSL evaluates test images from either $\mathcal{Y}^s$ or $\mathcal{Y}^u$, which increases difficulty because the classifier must distinguish seen and unseen classes simultaneously.

4.2.2 Side Information for Relating Seen and Unseen Classes

Side information links seen and unseen classes through shared semantic properties. A bird species with no labelled images may share attributes such as "red beak" or "striped wing" with seen species. A model can use visual-attribute associations learned from seen classes to recognise unseen classes that share the same attributes.

Common forms of side information include:

- **Attributes**: Binary or continuous vectors indicating properties such as colour, shape, or habitat. The CUB dataset provides 312 binary attributes per class, including *e.g.* beak colour, wing pattern, and bill shape.
- **Class embeddings**: Dense vectors from text encoders, such as Word2Vec or GloVe, that capture semantic similarity from word co-occurrence in text corpora.
- **Textual descriptions**: Natural-language sentences or paragraphs that describe each class and can be embedded with sentence encoders.

Human-annotated attributes are effective but expensive. The central question in this chapter is whether Large Language Models can generate side information that matches or exceeds the quality of manual annotation for low-shot image classification.

4.2.3 Semantic Prototypes for Classification

A semantic prototype is a single class vector in an embedding space. During inference, the classifier computes the similarity between a query image's embedding and each class prototype, then assigns the image to the class with the highest similarity. The approach requires a predefined set of classes to form prototypes and can classify query images into these classes without training examples.

Classification quality depends on how well semantic prototypes capture inter-class relationships. A useful prototype geometry places "house finch" near "sparrow" and far from "school bus", so visual features from house finch images align more strongly with bird prototypes than with unrelated object prototypes.

4.2.4 Key Architectures

Two established ZSL architectures serve as the main testbeds for LLM-generated side information.

CADA-VAE (Cross- and Distribution-Aligned Variational Autoencoder) [44] uses paired variational autoencoders to form a shared latent space for visual and semantic features. The two autoencoders map image features and class attributes into

a common latent representation. Cross-alignment and distribution-alignment losses align latent representations from the two modalities for the same class. Section 2.7 provides additional background.

APN (Attribute Prototype Network) [47] localises attributes in image regions using only class-level attribute supervision. APN combines a global visual-semantic embedding layer with a local attribute prototype module. Attribute regression and decorrelation losses encourage discriminative features that correspond to attribute-bearing regions. APN tests whether LLM-generated attributes can substitute for human-annotated attributes in localised recognition.

4.2.5 LLMs as Side Information Generators

I2MVFormer [46] uses LLM-generated descriptions for few-shot learning. Chapter 4 extends this line of work by comparing definitions, descriptions, captions, and attribute lists across OpenAI, SentenceTransformer, CLIP, PromCSE, CTP, and GloVe encoders.

LLMs generate side information from the large text corpora they were trained on. Prompting a model for class descriptions yields natural-language facts about object properties, visual appearance, and context. The experiments test whether these generated facts align with visual features and improve classification.

Sections 2.2 and 2.6 provide background on vision-language foundation models and adapters.

4.3 Methodology

4.3.1 Pipeline and Reporting Conventions

The chapter uses four evaluation pipelines, each isolating a different role for side information. Table 4.1 summarises the pipelines before the details are introduced.

Comparisons are strongest within a pipeline, where the visual features, splits, and evaluation protocol are fixed. Comparisons across pipelines are qualitative because the encoders, feature spaces, and supervision formats differ. The chapter interprets consistent trends and large margins as evidence, while treating small differences between non-identical pipelines as indicative rather than definitive.

Accuracy values are reported as percentages unless a table explicitly reports an intrinsic graph score. Some initial experiment runs do not include confidence intervals or repeated-seed statistics. Point estimates are therefore interpreted with caution, especially when differences are small or when methods use different text encoders.

Table 4.1: Summary of the Chapter 4 evaluation pipelines. Each pipeline changes the source of side information while keeping the relevant visual representation and evaluation protocol fixed for that experiment.

Pipeline	Visual representation	Semantic representation	Purpose
Intrinsic graph analysis	None	Text prototypes from labels, definitions, captions, and taxonomy	Measures semantic geometry before image classification.
MiniImageNet recognition	Fixed image features in episodic evaluation	Text prototypes from CLIP, Sentence-Transformer, GloVe, and generated text	Tests whether text prototypes transfer to zero-shot and one-shot recognition.
CUB with CADA-VAE	ResNet-101 image features	Attribute vectors or text embeddings	Tests generated side information in generative generalised ZSL.
CUB with APN	APN visual features	Ground-truth or LLM-derived attribute vectors	Tests direct replacement of expert attributes.
VDT-Adapter	CLIP ViT-B/16 features	GPT-generated visually descriptive text	Tests CLIP prompt ensembles and few-shot text adaptation.

4.3.2 Embeddings

Text embeddings represent class labels and descriptions in a form that can be compared with visual features in zero-shot learning. The embedding families considered here include OpenAI *text-embedding-ada-002*, SentenceTransformer *all-MiniLM-L6-v2*, PromCSE contrastive encoders, Contextualised Taxonomy Prediction, CLIP text embeddings, and GloVe baselines. Prompt construction varies from short curated labels to WordNet definitions, Wikipedia descriptions, LLM-generated captions, and hypernym chains.

Label Curation and Class-Specific Textual Descriptions

MiniImageNet contains animals (*e.g.* *house finch*, *lion*, *meerkat*), objects (*e.g.* *school bus*, *combination lock*), and fine-grain categories such as dog breeds and insect species. It also contains broader object categories, including musical instruments, that are not fine-grained concepts in the same sense. Raw ImageNet labels can be ambiguous (*file*, *rock beauty*, *stage*) or polysemous (*organ*). A text encoder may encode an unintended sense of such a label, which produces a prototype that no longer corresponds to the intended visual concept. The experiments use a human-curated mapping from raw labels to exact phrases. For example, "file" becomes "filing cabinet with drawers", "rock beauty" becomes "tropical fish in the sea", "house finch"

becomes “house finch small bird”, and “tank” becomes “tank armoured vehicle”.

OpenAI Embeddings

OpenAI variants use the `text-embedding-ada-002` model to embed textual descriptions for each class:

- **OpenAI-WordNetDef** embeds short WordNet definitions. For a class c with canonical name ℓ_c and WordNet definition d_c, prompts take the form “ℓ_c is a d_c”. The prompt captures dictionary-style semantics for each concept.
- **OpenAI-LabelText** embeds the manually refined label phrases from Section 4.3.2. These phrases disambiguate labels such as `file` and `rock_beauty` and produce class-specific prototypes.
- **OpenAI-WikiText** embeds short encyclopaedic descriptions sourced from Wikipedia.
- **OpenAI-WikiTextRefined** adds targeted manual corrections for problematic, outdated, or overly generic descriptions.
- **OpenAI-HypernymText** encodes WordNet hypernym chains. For each class, the method traverses ancestor nodes and builds prompts such as “ℓ_c is a type of h_1, a type of h_2, …”, where h_j denotes increasingly general hypernyms. Generic ancestors such as “object” are truncated so that the text emphasises informative category structure.

All OpenAI variants share an embedding space, allowing direct comparison across lexical, encyclopaedic, and taxonomic prompts. Lexical definitions encode dictionary meanings, refined labels resolve ambiguity, encyclopaedic descriptions add contextual attributes, and hypernym chains encode hierarchy.

SentenceTransformer Embeddings

SentenceTransformer variants use the `all-MiniLM-L6-v2` sentence encoder to embed short class descriptions:

- **ST-WordNetDef** encodes WordNet definitions using prompts such as “ℓ_c is defined as d_c”, where d_c is the synset definition.
- **ST-LabelText** encodes the same manually curated label phrases used by OpenAI-LabelText and CLIP-LabelText.

- **ST-ChatGPTDef** embeds short definitions generated by ChatGPT in response to a global meta-prompt that requests clear and concise definitions of the MiniImageNet classes. For example, the prompt produces "a small, cylindrical sausage typically served in a bun" for *hotdog*.

SentenceTransformer variants operate in a 384-dimensional space, compared with 1536 dimensions for OpenAI embeddings.

PromCSE Embeddings

PromCSE variants use a contrastively trained sentence encoder with prompt-based and prefix-based tuning. The method embeds richer sentences that resemble image captions rather than isolated labels or short definitions:

- **PromCSE-VisSent** builds 20 natural-language descriptions per class that focus on appearance and behaviour in context, such as "The hotdog sizzles on the grill, its smoky aroma and plump shape making it a popular summertime treat." Averaging these descriptions yields one prototype per class.

- **PromCSE-AllCaptions** aggregates multiple prompt groups from ChatGPT, including visual descriptions, item-centric, human-centric, and environment-centric captions. Averaging all captions yields a single embedding per class.

PromCSE is optimised for sentence-level semantic similarity and receives inputs close to the caption-like text used in vision-language pre-training.

Contextualised Taxonomy Prediction

CTP-Taxonomy uses a RoBERTa-based model trained for Contextualised Taxonomy Prediction. For each class, a pair of sentences combines the WordNet definition of the class with the definition of its immediate hypernym. The model encodes the pair, the final-layer token representations are averaged, and ℓ_2-normalisation forms prototypes tuned to hierarchical relations.

CLIP Text Baseline and Augmented Semantic Prototypes

CLIP-LabelText provides the main text-encoder baseline. Each class label maps to a short phrase that reflects the intended visual category, such as "school bus vehicle", and the CLIP text encoder produces a 512-dimensional embedding.

CLIP-LabelText defines the reference semantic graph used in the comparison. CLIP pre-training uses large-scale image-text pairs, so its text space often aligns strongly with visual features.

Augmented semantic prototypes combine text-derived similarity graphs with image-level self-supervised features. SimCLR [5] supplies class-level visual features. For each class, the procedure identifies nearest neighbours in a text embedding space, such as OpenAI `text-embedding-ada-002`, and forms linear combinations of the corresponding SimCLR features. This construction tests whether semantic neighbourhoods from text can improve visual prototypes by blending features from classes with shared properties.

4.3.3 Semantic Contrastive Architecture

The Semantic Contrastive architecture in Figure 4.1 evaluates zero-shot image classification with LLM-generated side information by aligning image and text representations in a shared projection space. The image branch uses a trainable convolutional encoder and projection MLP, while the text branch uses the frozen text encoder of CLIP followed by a projection MLP.

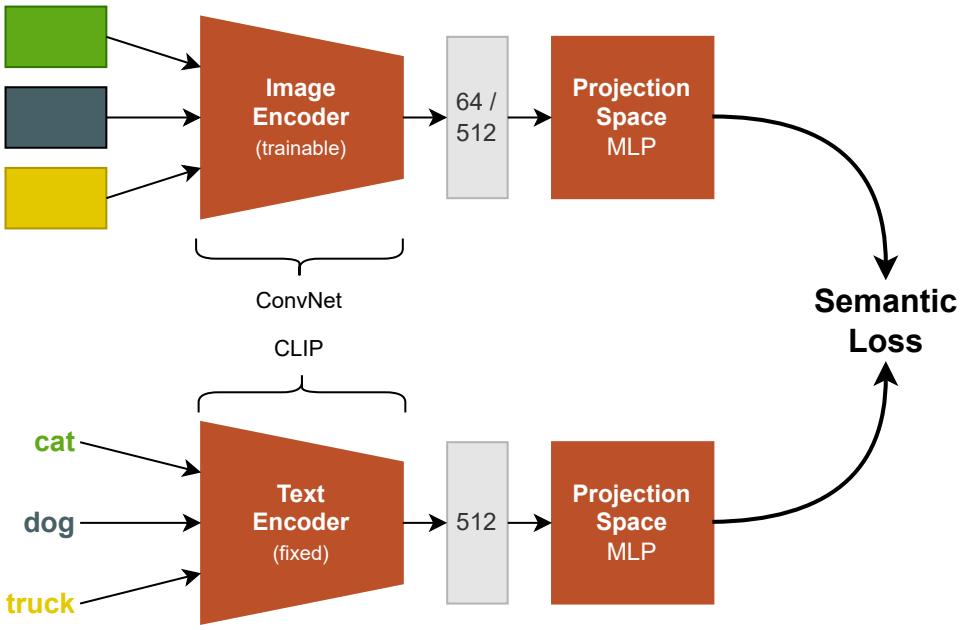


Figure 4.1: Semantic Contrastive architecture. The top branch maps class images through a trainable image encoder and projection-space MLP, while the bottom branch maps class text labels through a frozen CLIP text encoder and projection-space MLP. A semantic loss aligns the projected image and text representations so that images are pulled toward their corresponding class-text prototypes.

Class-level side information produces a single text embedding per class rather than a single embedding per image. The frozen text encoder provides class prototypes in the shared embedding space. Training aligns image embeddings with the corresponding class prototypes.

A temperature-scaled softmax over image-prototype similarities defines the contrastive losses. For each image embedding, the model computes dot-product similarities against all class prototypes, divides the logits by a temperature parameter, applies log-softmax, and optimises cross-entropy with the ground-truth class. The loss increases similarity to the correct prototype and decreases similarity to competing prototypes.

The t-SNE visualisation in Figure 4.2 contains image embeddings as dots and CLIP-derived class prototypes as X markers. Dog breeds such as "Arctic Fox" and "French Bulldog" form distinct clusters. Similar breeds such as "Siberian Husky" and "Alaskan Malamute" lie close together, reflecting visual resemblance. Some central classes overlap because they share visual attributes.



Figure 4.2: t-SNE visualisation of image embeddings (dots) and class prototypes (X markers) derived from the CLIP text encoder.

4.3.4 Detailed Prompt Engineering and Data Generation

The text encoder of CLIP accepts free-form text. The prompt design objective is to generate class-specific descriptions that resemble the text distribution used during CLIP pre-training. Cosine similarity between class embeddings in the embedding space of CLIP provides the class similarity metric. MiniImageNet supports this analysis because its classes span many noun categories and map to the WordNet graph.

WordNet [29] provides a ground-truth semantic graph for comparison. The following prompt generates free-form descriptions with ChatGPT[1]:

Prompt:

```
For each of the following objects, provide a clear and concise
definition:  Dugong, Walker hound, Lipstick, Yawl, list of classes
continues
```

Response:

- "Dugong": a large, herbivorous marine mammal native to the coasts of the Indian and Pacific Oceans, related to manatees and known for its slow-moving, graceful swimming style
- "Walker hound": a breed of hunting dog with a keen sense of smell and a strong, athletic build, used for tracking and trailing game
- "Lipstick": a cosmetic product used for coloring and conditioning the lips, typically in the form of a solid stick or a creamy paste
- "Yawl": a sailing vessel with two masts, the aft one of which is shorter than the mainmast and stepped farther aft, used for fishing or other purposes
- list of definitions continues

Caption-style prompting reduces the influence of a single class word on the CLIP embedding. Each generated description is embedded individually using CLIP, and the average embedding serves as the class prototype.

Prompt:

```
For each of the items in the list, write five plausible image
captions.  The images contain the items interacting with their typical
environment:
Dugong, Walker hound, Lipstick, Yawl, list of classes continues
```

[1]http://chatgpt.com/ Accessed 3 February 2023

Response:

- Dugong:
  - "Graceful dugong glides through the seagrass meadows."
  - "Dugong calf nuzzles closely to its mother in the azure waters."
  - "A curious dugong surfaces, its smooth back glistening in the sunlight."
  - "Underwater ballet: dugongs gracefully dance amidst coral reefs."
  - "Dugong family enjoys a peaceful moment, surrounded by colorful marine life."
- list continues

In-context prompting uses examples to steer response format and content [76]. The number of examples per class and the number of example classes vary across prompt variants, but all variants follow the same example-driven pattern:

Prompt:

Imagine you are an image annotation labeller captioning images containing different classes. You are asked to describe the unique attributes that you see in the image. Here are examples of what other labellers said:

- Rufous Hummingbird:
  - the bird has a thin bill, and black eyerings that are round.
  - this bird has wings that are black and has an orange belly
- Chuck wills Widow:
  - this dark colored bird has a bright red eyering and a long, slender bill.
  - this bird has wings that are black and has red eyes
- European Goldfinch:
  - the bird has a striped crown and a white belly.
  - this bird has a light brown body with black and yellow wings.

You are shown 10 different images of a Dugong. Write the captions

Response:

```
Dugong:

    • This marine mammal has a rounded body with a distinctively
      downturned snout.

    • Its flippers are broad and paddle-like, ideal for swimming
      gracefully through the water.

    • The dugong's skin is a mottled grayish-brown, blending seamlessly
      with its ocean habitat.

    • Its tail fluke is wide and horizontal, aiding in propulsion
      through the water.

    • Dugongs have small, dark eyes and small, rounded ears positioned
      behind their eyes.

    • The dugong's mouth is wide and lip-like, adapted for grazing on
      seagrass.

    • Its body is streamlined, allowing for efficient movement through
      the water.

    • Dugongs often exhibit scars or markings on their skin, evidence
      of encounters with other marine life or human activities.

    • These gentle creatures are herbivores, relying on seagrass
      meadows for their primary food source.

    • Dugongs are closely related to manatees and share similar
      features, such as a lack of dorsal fins and a thick, blubbery
      appearance.
```

Class-attribute prompting evaluates whether the full prompt must be sent separately for each class, or whether a single prompt containing the request, examples, and full class list suffices. The LLM context window defines the maximum number of tokens that can be processed from the conversation history and the current input. Earlier content is truncated once the context limit is reached. Generation may stop earlier because of a maximum-generation cap or a predicted stop token. The continue-reply strategy sends the full prompt once and then sends "continue" messages as the response grows. The strategy reduces repeated prompt tokens, but the earliest examples or class names may be truncated if the conversation reaches the context limit.

Prompt:

I am creating class attributes for a zero-shot image recognition algorithm to classify between different species of birds. List all the visual attributes that would be useful to distinguish between the species. Here are a few examples of the information I am looking for.

- Yellow-throated Vireo:
  - Beak: Medium, slightly hooked, upper mandible dark, lower mandible pale
  - Head: Yellow throat, white eye-ring extending to the sides, gray crown
  - Body: Olive-green upperparts, white underparts with yellow flanks
  - Wings: Olive-green with two white wing bars, rounded shape
  - Tail: Medium, square, olive-green
  - Legs: Blue-gray, medium length
  - Unique features: Yellow spectacles, contrasting head pattern
- Sayornis (Phoebe):
  - Beak: [example sentences continue]

Do the same for the following species 5 at a time: hooded merganser, house wren, list of classes continues, european goldfinch.

Response:

- Hooded Merganser:
  - Beak: Medium, thin, serrated, and pointed, dark color
  - Head: Male - black with a large, white-bordered black crest, Female - reddish-brown crest
  - Body: Male - black upperparts and white underparts, Female - grayish-brown upperparts and grayish-white underparts
  - Wings: Rounded with white patch visible in flight, dark primaries and secondaries
  - Tail: Short and pointed, dark color
  - Legs: Orange, short and webbed
  - Unique features: Crest and distinctive male head coloration, diving bird
- House Wren:
  - Beak: Short, thin, and slightly curved, dark color
  - response shortened
  - Body: Overall brown, barred with darker brown and pale grayish-white
  - Wings: Brown with faint bars
  - Tail: Brown with faint bars, short and often held upright
  - Legs: Pinkish-brown, medium length
  - Unique features: Small size, active and energetic behavior
- response shortened

4.3.5 Attribute Prototype Network Modification

Section 4.2 describes the Attribute Prototype Network as an architecture that incorporates images and class-level attribute vectors as supervision. APN uses attribute annotations provided with CUB in the original experiments. The experiments in this chapter replace those attribute vectors with several LLM-generated alternatives.

4.3.6 Prompting for Attribute Verification

Attribute verification tests the factual correctness of LLM-generated CUB attributes. The prompt asks the LLM to state whether each attribute fact derived from the CUB annotations is true or false. A full list of attributes and option sets is provided, and responses are compared against CUB annotations. The evaluation captures erroneous omissions, where the output misses correct attributes, and erroneous commissions, where the output marks incorrect attributes as true.

Prompt:

For a Forsters Tern bird, answer whether each of the following attributes are True or False for the specific bird:

- has bill shape that is curved (up or down):
- has bill shape that is dagger:
- has bill shape that is hooked:
- has bill shape that is needle:
- has bill shape that is hooked seabird:
- has bill shape that is spatulate:
- has bill shape that is all-purpose:
- has bill shape that is cone:
- has bill shape that is specialized:
- has wing color that is blue:
- has wing color that is brown:
- has wing color that is iridescent:
- has wing color that is purple:
- has wing color that is rufous:
- has wing color that is grey:
- has wing color that is yellow:
- has wing color that is olive:
- has wing color that is green:
- has wing color that is pink:
- has wing color that is orange:
- has wing color that is black:
- has wing color that is white:
- has wing color that is red:
- has wing color that is buff:
- list of attributes-value pairs continues

Response:

- response shortened
- has bill shape that is curved (up or down): *True*
- has bill shape that is dagger: False
- has bill shape that is hooked: False
- has bill shape that is needle: False
- has bill shape that is hooked seabird: False
- has bill shape that is spatulate: False
- has bill shape that is all-purpose: False
- has bill shape that is cone: False
- has bill shape that is specialized: True
- has wing color that is blue: False
- has wing color that is brown: *True*
- has wing color that is iridescent: False
- has wing color that is purple: False
- has wing color that is rufous: False
- has wing color that is grey: False
- has wing color that is yellow: False
- has wing color that is olive: False
- has wing color that is green: False
- has wing color that is pink: False
- has wing color that is orange: False
- has wing color that is black: False
- has wing color that is white: True
- has wing color that is red: False
- has wing color that is buff: False
- list of attributes continues

Slot-filling prompting changes boolean responses to categorical selection. This format enforces a single attribute-value pair per slot, matching the CUB ground-truth format.

Prompt:

For a Black-footed Albatross bird, select the correct option for each attribute:

- ‘has_bill_shape’: [‘curved_(up_or_down)’, ‘dagger’, ‘hooked’, ‘needle’, ‘hooked_seabird’, ‘spatulate’, ‘all-purpose’, ‘cone’, ‘specialized’],
- ‘has_wing_color’: [‘blue’, ‘brown’, ‘iridescent’, ‘purple’, ‘rufous’, ‘grey’, ‘yellow’, ‘olive’, ‘green’, ‘pink’, ‘orange’, ‘black’, ‘white’, ‘red’, ‘buff’],
- list of attributes continues

Response:

Based on the characteristics of the Black-footed Albatross bird, the correct options are:

- ‘has_bill_shape’: ‘hooked’
- ‘has_wing_color’: ‘black’
- list of attributes continues

CSV-formatted prompting scales the previous prompt to more bird species:

Prompt:

Create a table (in CSV format) with attributes as the columns and different birds as the rows. For the attributes, select the correct option for each attribute of the bird:

- Bill shape: [curved, dagger, hooked, needle, hooked (seabird), spatulate, all-purpose, cone, specialized]
- Wing color: [blue, brown, iridescent, purple, rufous, grey, yellow, olive, green, pink, orange, black, white, red, buff]
- list of attributes continues

The rows of the table are the following list of birds: [‘red-winged blackbird’, ‘rusty blackbird’, ‘yellow-headed blackbird’, [list of all birds in CUB dataset]]

Age- and sex-specific CSV prompting probes whether factual errors arise from intra-class variation. The previous prompts assume that a single attribute-value pair holds for all, but some species vary by age, breeding stage, or sex, especially in plumage.

Prompt:

```
create a table (in csv format) with corresponding bird attributes
as the columns and the rows should be:  Adult male, Adult female,
juvenile male, juvenile female, breeding male, breeding female.  For
the attributes, select the correct option for each attribute of the
bird:

   • list of attributes

Create the table for Red-winged Blackbirds
```

Manual verification indicates that LLM outputs often contain unsupported fine-grain facts, and repeated prompts can return inconsistent results. LLM hallucination surveys report similar error modes [77]. LLMs can nevertheless convert text into structured attribute vectors and can extract facts from supplied text. Zero-shot prompting enables large language models to perform fact extraction from provided context [78]. The following experiments test whether unstructured text sources can be converted into CUB-style attribute vectors with an LLM. Wikipedia, All About Birds, and LLM-generated general descriptions provide the text sources. These experiments do not measure truthfulness directly because manual verification of every fact in the supplied text is not performed.

The following prompt requests attribute extraction from a supplied bird description:

Prompt:

```
This is a description of a bird:  Size & Shape

   • A plump seabird with an unusual rounded bill, thick neck, broad
     chest, and very short tail.

RELATIVE SIZE

   • Larger than a Least Auklet, smaller than a Pigeon Guillemot.

[description continues]
Here is a dictionary of attributes as keys and list of values it can
take as values.

   • bill shape:  [curved, dagger, hooked, needle, hooked (seabird),
     spatulate, all-purpose, cone, specialized]

   • wing color:  [blue, brown, iridescent, purple, rufous, grey,
     yellow, olive, green, pink, orange, black, white, red, buff]

   • list of attributes continues

Each attribute can take multiple positive values in the list.  Give the
correct values for each attribute based on the supplied description.
```

Response:

```
{
```

- `"billShape": "spatulate",`
- `"wingColor": ["brown", "white"]`
- `list of attributes continues`

```
}
```

Structured-output evaluation also tests adherence to formats such as JSON.

4.3.7 CADA-VAE System Prompts

The CADA-VAE experiments use the OpenAI public API during development. API requests include a *system prompt*, which specifies response constraints more strongly than a standard prompt on the ChatGPT web interface. The system prompt used here constrains the LLM output to a desired JSON format.

Prompt:

```
For a Black-footed Albatross bird, select the correct option for each
attribute.
System Prompt: You are a truthful assistant that knows a lot about
birds. You only respond in JSON format with no explanations.

"type": "object", "properties": "billShape": "type": "string",
"enum": ["curved(up or down)", "dagger", "hooked", "needle", "hooked
seabird", "spatulate", "all-purpose", "cone", "specialized"] ,
"wingColor": "type": "string", "enum": ["blue", "brown", "iridescent",
"purple", "rufous", "grey", "yellow", "olive", "green", "pink",
"orange", "black", "white", "red", "buff"] , [list of attributes
continues] , "required": ["billShape", "wingColor", [list of attributes
continues]]
```

Response:

```
"billShape": "hooked seabird", "wingColor": "black", [list of
attributes continues]
```

4.3.8 Quantitative Comparison of Embedding Families

Quantitative comparison evaluates how text embedding families encode inter-class semantics relevant to zero-shot and few-shot transfer. The analysis considers class-level prototypes derived from labels, definitions, and curated descriptions for OpenAI, SentenceTransformer, PromCSE, CLIP, and CTP variants. The objective is to assess coherence and separability in embedding space, which indicates the strength of the semantic prior available to downstream classifiers.

Pairwise cosine similarity provides the intrinsic geometry for each embedding family. Semantically related classes should lie close together, and unrelated classes should lie far apart, if prototypes offer a useful prior for classifiers that compare image features with class vectors. The results in Table 4.2 contain the mean, standard deviation, and quantiles of pairwise cosine similarities between class-level prototypes.

Table 4.2: Summary of pairwise cosine similarities between class-level prototypes for each embedding family. Each row corresponds to one embedding variant, identified by its short name.

Short Name	dim	mean	std	p25	p75
CLIP-LabelText	512	0.624	0.110	0.572	0.700
CTP-Taxonomy	1024	0.612	0.524	0.551	0.953
OpenAI-HypernymText	1536	0.777	0.032	0.755	0.793
OpenAI-LabelText	1536	0.766	0.027	0.748	0.779
OpenAI-WikiText	1536	0.728	0.031	0.708	0.743
OpenAI-WikiTextRefined	1536	0.719	0.032	0.697	0.735
OpenAI-WordNetDef	1536	0.746	0.026	0.729	0.760
PromCSE-AllCaptions	768	0.763	0.056	0.725	0.801
PromCSE-VisSent	768	0.730	0.067	0.686	0.772
ST-ChatGPTDef	384	0.183	0.136	0.088	0.250
ST-LabelText	384	0.147	0.110	0.073	0.204
ST-WordNetDef	384	0.120	0.117	0.039	0.176

OpenAI variants and PromCSE variants produce high and relatively tight cosine-similarity distributions, with means around 0.73 to 0.78. OpenAI-HypernymText attains the highest average similarity, consistent with the global coherence imposed by hypernym chains.

SentenceTransformer variants exhibit substantially lower means (≈ 0.12 to 0.18) and larger relative variance, indicating a more dispersed embedding geometry. CLIP-LabelText lies between these extremes, with a mean similarity 0.62 and a broader tail towards lower similarities.

CTP-Taxonomy has a high mean and a very large standard deviation, with similarities ranging from strongly negative to nearly one. This distribution indicates a heterogeneous mixture of tightly and loosely clustered concepts. Kernel-density estimates in Figure 4.3 visualise these patterns.

OpenAI and PromCSE variants form sharp peaks at high similarity values. SentenceTransformer embeddings remain more spread out and closer to zero, while CLIP-LabelText has an intermediate profile.

The distributions show that generated text changes the geometry of the semantic prototypes, but high average similarity alone does not imply better recognition.

Downstream experiments are therefore needed to determine whether these geometries align with image features.

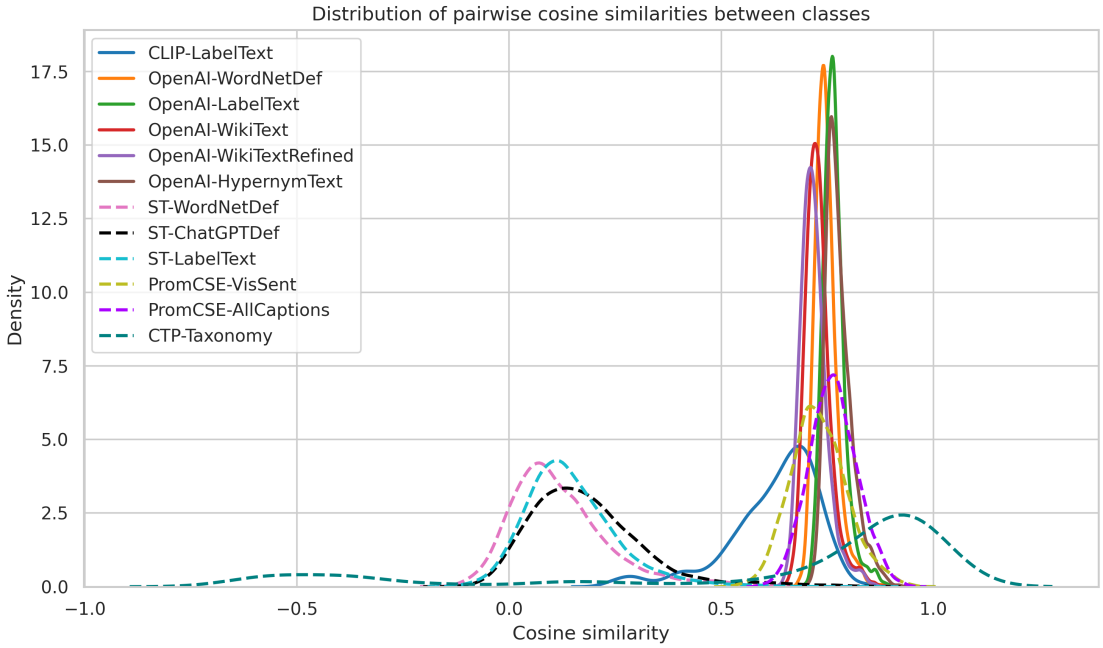


Figure 4.3: Distribution of cosine similarities between class-level prototypes for each embedding family. Short names follow Table 4.2.

4.3.9 Graph Structure and Neighbourhood Overlap

Graph analysis complements pairwise similarity statistics by examining which classes become neighbours under each embedding family. This view is useful because a prototype-based classifier depends on local structure: related classes should form compact neighbourhoods, while unrelated classes should remain separated.

Graph construction starts with the cosine similarity matrix for each embedding family. Diagonal entries are excluded because they compare each class with itself. The 95th percentile of off-diagonal similarities serves as a method-specific threshold, and only edges above it are retained. This produces an undirected graph with classes as nodes and high-similarity class pairs as edges. Since the same percentile threshold is used for every embedding family, all graphs contain the same edge fraction. Differences in graph statistics reflect which neighbours each method selects, not how many edges it keeps.

Average clustering coefficients in Table 4.3 measure how often neighbours of a class also connect to each other. ST-ChatGPTDef has the highest clustering coefficient (0.504), so ChatGPT definitions create the most triangle-rich local neighbourhoods under this threshold. Among OpenAI variants, OpenAI-WikiTextRefined

Table 4.3: Graph statistics for thresholded semantic graphs at the 95th percentile threshold. All graphs have an identical average degree (4.960). Differences in clustering coefficient reflect how consistently each method forms triangles among related classes.

Short Name	Family	Avg Degree	Avg Clustering
CLIP-LabelText	CLIP	4.960	0.358
CTP-Taxonomy	CTP	4.960	0.353
OpenAI-HypernymText	OpenAI	4.960	0.435
OpenAI-LabelText	OpenAI	4.960	0.401
OpenAI-WikiText	OpenAI	4.960	0.386
OpenAI-WikiTextRefined	OpenAI	4.960	0.438
OpenAI-WordNetDef	OpenAI	4.960	0.354
PromCSE-AllCaptions	PromCSE+ChatGPT	4.960	0.332
PromCSE-VisSent	PromCSE+ChatGPT	4.960	0.304
ST-ChatGPTDef	ST+ChatGPT	4.960	0.504
ST-LabelText	ST	4.960	0.393
ST-WordNetDef	ST	4.960	0.311

(0.438) and OpenAI-HypernymText (0.435) have the strongest clustering, consistent with the structured information in encyclopaedic text and hypernym chains. CLIP-LabelText (0.358), CTP-Taxonomy (0.353), and OpenAI-WordNetDef (0.354) have similar clustering. PromCSE variants have lower clustering (0.304 and 0.332), indicating less locally transitive neighbourhoods for these caption-based prototypes.

The degree distributions in Figure 4.4 indicate whether certain classes become hubs. The common threshold yields a mean degree of approximately 5 for all methods. Broader tails indicate that a small number of classes connect to many more neighbours than typical classes. OpenAI and PromCSE variants have broader tails extending to degrees of 20 to 30, whereas SentenceTransformer variants concentrate more tightly around the mean.

Neighbourhood overlap measures whether different embedding families choose the same local neighbours. For each class and embedding family, the five nearest neighbours are selected by cosine similarity. Jaccard overlap then compares neighbour sets between pairs of embedding families. A value of 1 denotes identical neighbour sets, and 0 denotes no shared neighbours. The heatmap in Figure 4.5 reports the mean Jaccard overlap across all classes.

Three patterns emerge from the neighbourhood-overlap results. First, methods within the same encoder family have the strongest agreement. OpenAI-WikiText and OpenAI-WikiTextRefined share 0.732 overlap, the highest off-diagonal value in the matrix. Other OpenAI pairs range from 0.335 to 0.425, SentenceTransformer pairs range from 0.227 to 0.309, and the two PromCSE variants share 0.374 overlap.

Second, CLIP-LabelText overlaps most with OpenAI-LabelText (0.279), followed

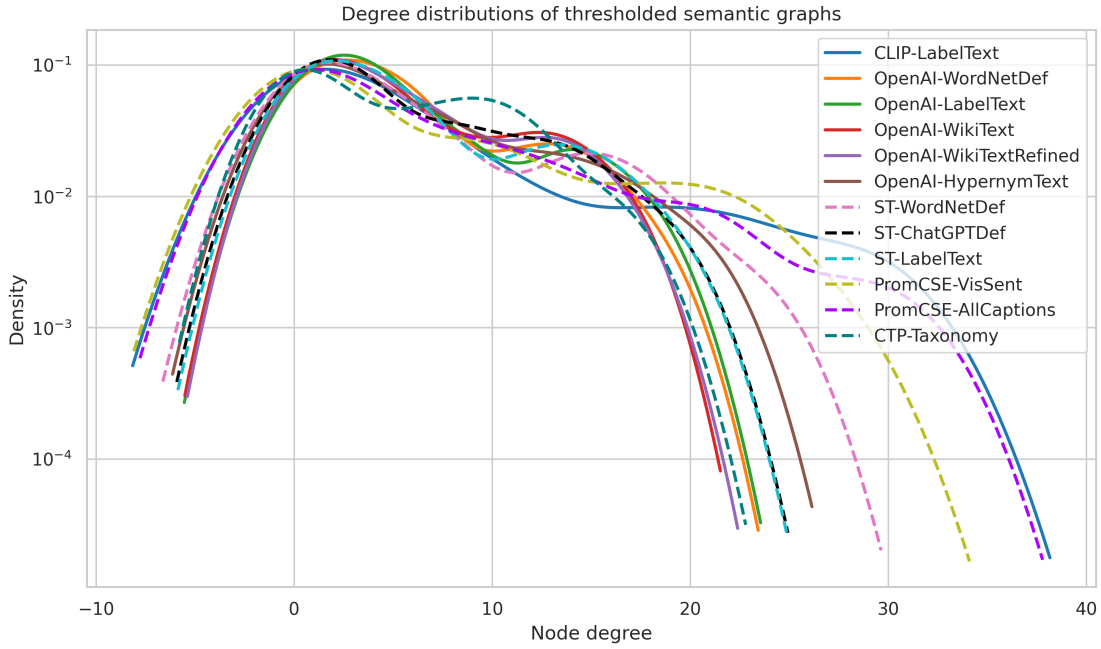


Figure 4.4: Degree distributions of thresholded semantic graphs. Smoothed density estimates centre around degree 5, with broader tails for several OpenAI variants.

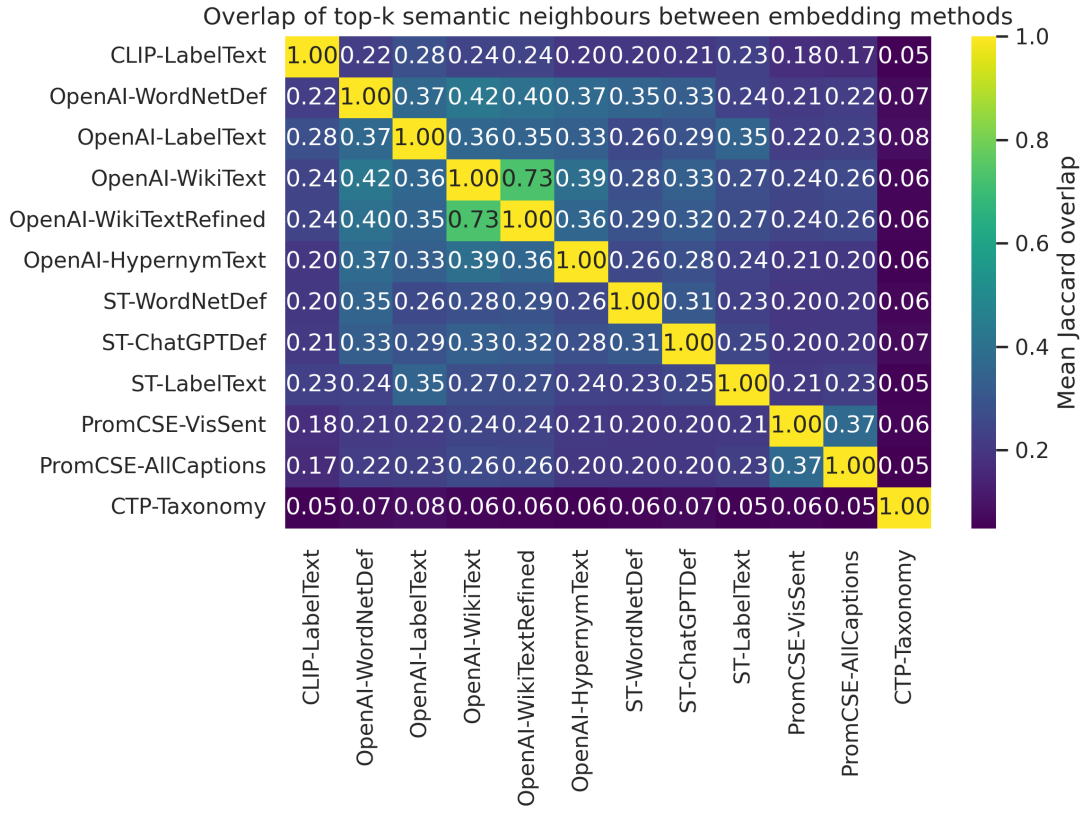


Figure 4.5: Neighbourhood overlap between embedding families, measured as mean Jaccard overlap of top-5 neighbours. Higher values indicate more similar neighbour assignments. Short names follow Table 4.2.

by the Wikipedia-based OpenAI variants (0.240). OpenAI-WordNetDef overlaps with CLIP-LabelText at 0.220, while OpenAI-HypernymText drops to 0.197. SentenceTransformer variants overlap with CLIP-LabelText in a similar but slightly lower range (0.203 to 0.230), and PromCSE variants overlap less strongly (0.165 to 0.177). The result supports OpenAI-LabelText as the closest text-only approximation to the CLIP label neighbourhood, but does not imply that every OpenAI prompt variant is closer to CLIP than every SentenceTransformer variant.

Third, CTP-Taxonomy differs strongly from all other methods. Its overlap with other embedding families ranges from 0.048 to 0.078. Taxonomy-prediction training produces a distinct organisation of the label space that does not closely align with definition-based, caption-based, or CLIP-style label encoders.

Clustering and CLIP overlap measure different properties. High clustering indicates internally coherent neighbourhoods within an embedding family, while high overlap with CLIP indicates closer agreement with a vision-language text space. OpenAI-LabelText best matches the CLIP local neighbourhoods among the text-only variants, ST-ChatGPTDef forms the most clustered graph, and CTP-Taxonomy encodes a markedly different taxonomic structure. The following sections test whether these differences translate into recognition accuracy.

4.4 Truthfulness of Text Embeddings as Semantic Graphs

The analysis above describes similarity distributions and graph topology. The next evaluation measures graph-based "truthfulness" by comparing semantic graphs induced by text embeddings with two external references: the WordNet class hierarchy and the embedding space of CLIP.

Truthfulness denotes agreement between nearest-neighbour relations encoded by an embedding and nearest-neighbour relations implied by an external semantic reference. WordNet provides a lexical and taxonomic reference, whereas CLIP provides a vision-language reference learned from large-scale image-text pairs. The evaluation asks whether each embedding family preserves the relational structure of these references.

4.4.1 Experimental Protocol

Text-prompt construction yields an embedding for each class c under the method of interest. Pairwise cosine similarities between all class embeddings define edge weights in a complete graph.

Two reference graphs provide targets for comparison. The WordNet graph uses path-based similarities on the hypernym tree. The CLIP graph uses the CLIP text encoder with prompts of the form "a photo of `<class_name>`", following standard prompt-based evaluation. Alignment is measured by comparing the neighbour rankings in the embedding graph with those in the WordNet and CLIP reference graphs.

Let $A^{(m)} \in \mathbb{R}^{C \times C}$ denote the sparsified semantic graph induced by method m, and let $A^{(r)}$ denote the corresponding reference graph from WordNet or CLIP. After min-max normalisation, diagonal removal, symmetrisation, and row-wise sparsification, the graph-similarity score is

$$S(m, r) = 100 \times \frac{\langle A^{(m)}, A^{(r)} \rangle_F}{\|A^{(m)}\|_F \|A^{(r)}\|_F},$$

where $\langle \cdot, \cdot \rangle_F$ denotes the Frobenius inner product. Higher values indicate greater overlap between high-similarity edges in the method graph and the reference graph. Scores around 20 indicate partial, rather than complete, preservation of the reference neighbourhood structure.

4.4.2 Results

Table 4.4 contains the main truthfulness scores for representative embedding families and prompt designs.

Table 4.4: Comparison of text embedding families on a graph-based truthfulness benchmark that measures alignment with WordNet and CLIP semantic graphs. The table reports graph similarity scores with respect to WordNet and CLIP. Higher is better.

Embedding	Dim	WordNet graph sim	CLIP graph sim
Baseline CLIP	512	20.157	n/a
BERTSentence	384	11.518	19.041
BERTSentence + ChatGPT def.	384	12.300	20.262
PromCSE + ChatGPT def.	768	18.602	24.040
PCA(CLIP)	64	14.214	20.019

Two trends stand out. PCA compression of CLIP text embeddings to 64 dimensions retains much of their semantic structure. PCA(CLIP) has higher WordNet graph similarity than both BERTSentence variants, but its CLIP graph similarity lies between BERTSentence and BERTSentence + ChatGPT definitions. This more limited result still indicates that a low-dimensional projection preserves a substantial part of the CLIP semantic geometry.

The definitions generated by the LLM raise the graph scores for both sentence encoders. The PromCSE + ChatGPT variant records the highest CLIP-graph similarity in Table 4.4, although this intrinsic score does not establish downstream

recognition performance.

Generated definitions improve text-only graph alignment in this benchmark, but the scores do not establish that generated side information can replace expert attributes in downstream recognition.

4.4.3 Implementation Details

The truthfulness evaluation first maps ImageNet class labels to WordNet noun synsets. A synset groups lemmas that share one sense and connects to other synsets through hypernym and hyponym relations. When several synsets match a class label, the procedure computes Leacock-Chodorow similarity for all candidate synset pairs and retains the maximum score for that class pair. NLTK WordNet provides taxonomy depths and similarity values.

WordNet similarities are min-max normalised over non-diagonal entries, the diagonal is set to zero, and symmetry is enforced with the elementwise maximum of the matrix and its transpose. Row-wise sparsification keeps only similarities above the 95th percentile of non-zero entries. The same sparsification rule is applied to text-derived graphs so that comparisons use the same graph density.

Text prototypes use three sources of descriptions: WordNet glosses, curated label phrases, and concise, single-sentence LLM definitions. These descriptions are encoded with SentenceTransformer, PromCSE, CLIP ViT-B/16 text, and OpenAI embeddings. The SentenceTransformer model is `all-MiniLM-L6-v2`, and the OpenAI model is `text-embedding-ada-002`. CLIP text features are averaged over a small set of prompt templates, SentenceTransformer and PromCSE use pooled sentence embeddings, and OpenAI uses returned unit-normalised vectors. All embeddings are ℓ_2-normalised before cosine-similarity matrices are computed.

The CLIP image reference graph uses a fixed image sample per class. CLIP image features are extracted, ℓ_2-normalised, averaged into class centroids, and converted into a class-to-class cosine-similarity matrix. The matrix is symmetrised, its diagonal is set to zero, and the same 95th percentile row-wise sparsification is applied. The normalised Frobenius inner product in $S(m, r)$ provides the reported alignment score. Top-k neighbour lists provide qualitative inspection.

4.5 Few- and Zero-Shot Recognition on MiniImageNet

MiniImageNet evaluation tests how intrinsic semantic structure translates into few-shot and zero-shot recognition performance.

Episodic evaluation frames recognition as a sequence of small classification tasks. The *way* denotes the number of candidate classes in an episode, so 5-way, 16-way, and 20-way tasks require discrimination among 5, 16, and 20 labels, respectively. The *shot* denotes the amount of supervision per class in the support set. A 0-shot episode provides no labelled examples and relies solely on side information such as class names or text-derived prototypes. A 1-shot episode provides a single labelled example per class. Each episode uses a class-balanced support set and a disjoint query set. Accuracy is averaged over sampled episodes from held-out classes.

The MiniImageNet pipeline does not compare raw image and text vectors. The visual and semantic encoders have different output dimensions and coordinate systems, so cosine similarity is meaningful only after alignment. During training on the base classes, the visual branch and an encoder-specific text projection map their outputs into a shared 64-dimensional space. A semantic contrastive objective aligns an image with the projected prototype of its class. Unless otherwise stated, *cosine-similarity accuracy* $\mathrm{acc}_{\mathrm{sim}}$ denotes mean per-episode accuracy in this aligned space.

The 0-shot and 1-shot paths use that space differently. In a 0-shot episode, the projected text vector is the class prototype and each query is assigned to its nearest prototype. In a 1-shot episode, the single labelled support image for each class is encoded in the same space and supplies the visual class prototype; the FEAT-style adaptation module refines the support and query representations before cosine-similarity classification. The semantic prototypes influence the one-shot result through the base-class alignment training, but an unprojected text vector is not directly averaged with the support image. Thus, the 0-shot result tests transfer through text prototypes alone, whereas the 1-shot result tests the learned representation with one task-specific visual example per class.

4.5.1 Prompt Design for CLIP-Based MiniImageNet Prototypes

The study evaluates several prompt families for MiniImageNet text prototypes:

- **BERTSentence** uses a sentence transformer to embed disambiguated label phrases. Its inputs closely mirror OpenAI-LabelText and ST-LabelText, but it uses a different encoder.

- **CLIP baseline** uses the standard CLIP prompt "A photo of `<class_name>`".

- **ChatGPT definitions** use short definitions generated by ChatGPT in response to a "clear and concise definition" meta-prompt and embedded with a sentence-transformer.

- **ChatGPT captions** use five plausible image captions per class, each describing the object in a typical environment, and average them into one prototype per class.
- **Wikipedia context** uses short Wikipedia excerpts as contextual descriptions and embeds them into a single prototype per class.

All variants use the same visual architecture, base-class split, episodic protocol, and projection dimensionality. The semantic prototype family changes between runs, so performance differences primarily reflect the side information used to align the visual representation.

The MiniImageNet comparison is strongest within rows that share the same image architecture, projection dimension, class splits, and evaluation episodes, but not all rows are encoder-matched. The CLIP baseline benefits from a text encoder trained with images, whereas BERTSentence, GloVe, and several generated-text variants use language-only encoders. These results test the practical value of each prototype construction after alignment, not a perfectly controlled comparison of prompt content alone.

4.5.2 Main MiniImageNet Results

Table 4.5 lists 1-shot and 0-shot 20-way test accuracies for the main prompt families.

Table 4.5: MiniImageNet zero-shot and few-shot results measured by cosine-similarity accuracy, $\mathrm{acc}_{\mathrm{sim}}$, for different text prompts in the learned shared projection space. The table reports 1-shot and 0-shot 20-way test accuracies in percent.

Embedding	1-shot 20-way test [%]	0-shot 20-way test [%]
BERTSentence	21.57	17.75
CLIP baseline	22.07	28.71
ChatGPT definitions	22.23	16.95
ChatGPT captions	24.91	16.28
Wikipedia context	20.12	19.94

The canonical CLIP prompt performs strongly in the 0-shot setting, reaching 28.71% accuracy in the 20-way test configuration. This score exceeds bare BERTSentence label embeddings and all LLM-enhanced BERTSentence variants, showing that CLIP's image–text pre-training supplies a useful visual prior even after its text features are projected into the experiment's shared space. ChatGPT captions achieve the highest 1-shot 20-way accuracy (24.91%) among the non-CLIP baselines, but their 0-shot performance trails CLIP and BERTSentence label embeddings. Descriptive, context-rich captions may overfit typical scenes and become less robust when no example with visual support is available. Wikipedia context

provides balanced but modest performance, improving over ChatGPT definitions in 0-shot evaluation but not matching the CLIP baseline.

On MiniImageNet, LLM-generated text does not improve direct replacement of CLIP-compatible label prompts in the 0-shot setting. LLM-generated captions help in the 1-shot setting, but the gain appears when labelled visual support is available, not when text alone must define the classifier.

4.5.3 GloVe-Based Semantic Priors

GloVe baselines disentangle the effects of large transformer encoders from those of simpler embedding methods. All GloVe models operate in a lower-dimensional language-based embedding space and provide a sanity check for prototype quality.

Table 4.6 lists MiniImageNet results obtained by replacing CLIP or BERTSentence text embeddings with GloVe.

Table 4.6: MiniImageNet zero-shot and few-shot results using GloVe embeddings as semantic prototypes. The table reports 1-shot and 0-shot 20-way test cosine-similarity accuracies in percent.

Embedding	1-shot 20-way test [%]	0-shot 20-way test [%]
GloVe (photo prompt)	21.75	21.33
GloVe (Wikipedia)	21.30	14.82
GloVe (ChatGPT text)	21.74	20.54

GloVe with CLIP-style photo prompts achieves competitive 1-shot and 0-shot performance, including 21.33% 0-shot 20-way accuracy. Richer Wikipedia or ChatGPT context changes performance only modestly. The comparison indicates that alignment between the semantic space and visual pre-training, as in CLIP, contributes more than purely textual enrichment alone.

The GloVe results reinforce the same conclusion: generated text does not automatically improve recognition unless the embedding space preserves visually relevant neighbourhoods.

4.5.4 Implementation Details

MiniImageNet experiments use a FEAT-style episodic pipeline with the four-block convolutional image encoder shown in Figure 4.1. The image branch produces 64-dimensional features. The semantic encoders retain their native output sizes: 384 dimensions for SentenceTransformer, 512 for CLIP, and the corresponding size for each other encoder. A learned linear layer projects each semantic representation to 64 dimensions. The projection and image encoder are trained on MiniImageNet base

classes with the semantic contrastive objective described in Section 4.2. Held-out classes are used only for episodic evaluation.

For each prompt family, the text encoder produces one class-level prototype tensor in its native space and the associated learned projection maps it into the common space. The 0-shot routine selects the projected prototypes for the classes sampled in an episode and compares all query embeddings with them. The 1-shot routine instead passes one support image per class and the query images through the trained image branch; the Transformer-based FEAT module adapts these episode features and forms one visual prototype per class. Accuracy is averaged over 20-way episodes with disjoint support and query images. The GloVe variants use the same image architecture and evaluation protocol, with a projection sized for the GloVe input. This alignment stage is essential: equal vector length alone would not make independently trained language and vision spaces comparable.

4.6 CUB: Generative ZSL with CADA-VAE-Style Embedding Transfer

CUB experiments further test language-model-based semantic priors in a generative zero-shot setting based on CADA-VAE. The objective is to form a joint latent space for visual features and side information, where side information may be attributes or text embeddings. Performance is measured by seen-class accuracy, novel-class accuracy, and their harmonic mean (h).

Standard ZSL predicts only unseen classes, whereas generalised ZSL predicts over the union of seen and unseen classes. Given seen-class accuracy A_{seen} and unseen-class accuracy A_{novel}, the harmonic mean

$$h = \frac{2A_{\text{seen}}A_{\text{novel}}}{A_{\text{seen}} + A_{\text{novel}}}$$

penalises methods that perform well on only one split and provides a balanced summary of performance.

4.6.1 Side Information and Baselines

The analysis considers several forms of side information:

- **Ground-truth attributes**, using the original CUB attribute vectors or variants with only the top one or two attributes per class.
- **Word2Vec** embeddings of human-provided sentences, following the original CADA-VAE setting.

- **GloVe embeddings of ChatGPT captions**, where ChatGPT generates 10 descriptive sentences per class, and GloVe embeds them.
- **ChatGPT slot-filled attributes**, where ChatGPT fills a pre-specified attribute template with natural-language responses.
- **GloVe embeddings of part-based prompts**, where ChatGPT generates 7 visual sentences that focus on key parts and colours of each bird.

4.6.2 Results

Table 4.7 contains the main CADA-VAE results on CUB.

Table 4.7: CUB zero-shot learning results with CADA-VAE-style generative models and different forms of side information. The table reports novel-class accuracy, seen-class accuracy, and their harmonic mean (h).

Side information	Novel	Seen	Harmonic Mean
CUB ground-truth attributes (full)	51.6	53.5	52.4
CUB ground-truth attributes (top 2 per class)	19.41	51.87	28.25
Word2Vec on 10 human sentences	22.67	50.94	31.38
GloVe on ChatGPT 10-shot prompt captions	10.16	44.82	16.56
ChatGPT slot-filled attributes	14.44	49.95	22.41
GloVe on GPT-4 7-part visual sentences	12.53	30.30	17.72
GloVe on ChatGPT class-level captions	44.72	34.14	38.72

CUB class-level attribute vectors average binary image-level annotations across all images of each class, producing per-attribute frequencies as side information. The top-2 variant selects the two highest-frequency attributes per class and zeroes the rest, producing a sparse prototype that roughly matches the slot count used by LLM-derived attributes. This truncation preserves seen-class alignment but removes complementary fine-grained cues needed for novel classes, causing a large drop in novel accuracy. Word2Vec embeddings of 10 human-written sentences achieve a competitive harmonic mean of 31.38, showing that natural language can partially substitute for structured attributes. ChatGPT 10-shot captions embedded with GloVe underperform the human-sentence baseline, reaching a harmonic mean of 16.56.

ChatGPT class-level captions embedded with GloVe produce the strongest LLM-derived result in this table, with a harmonic mean of 38.72. Careful prompt design yields language-model-derived side information that is competitive with traditional human-authored descriptions, although full CUB attributes remain stronger.

In the CADA-VAE experiments, generated captions outperform several weaker semantic baselines but do not surpass full ground-truth attributes. In this set-

ting, LLM side information supplements rather than replaces high-quality manual attributes.

4.6.3 Implementation Details

CADA-VAE-style experiments use an established two-branch variational architecture. One branch models ResNet-101 visual features, and the other models semantic side information. Cross-reconstruction and distribution-alignment losses align the two latent distributions.

The CUB preparation stage loads precomputed ResNet-101 image features and CUB attribute matrices, constructs the standard seen and unseen splits, and remaps labels to contiguous indices. The same loader also converts the standard test-unseen split into N-shot examples for few-shot ablations. Each semantic modality is aligned to the CUB class order before training.

Each row of Table 4.7 uses the same ResNet-101 visual features and changes only the semantic input. The semantic inputs include full CUB attributes, sparsified CUB attributes, Word2Vec embeddings of human sentences, GloVe embeddings of ChatGPT 10-shot captions, ChatGPT slot-filled attribute vectors, GloVe embeddings of part-based visual sentences, and GloVe embeddings of class-level ChatGPT captions.

4.7 CUB: Attribute Prediction Network with LLM-Derived Attributes

The APN evaluation tests language-model-generated attribute-like side information for discriminative zero-shot learning on CUB. The model predicts attribute vectors from images, and a nearest-neighbour classifier operates in attribute space at test time. The experiments compare ground-truth attributes with several LLM-derived approximations.

4.7.1 Side Information Variants

The analysis includes two main sources of side information:

- **Ground-truth attribute label vectors**, either full real-valued CUB attribute-frequency vectors derived from the manual CUB attribute labels, or binarised variants with top-1, top-2, or confidence-based thresholds.
- **LLM-generated attribute lists**, where ChatGPT selects factually correct attributes from a candidate list, derives attributes from bird resources such

as WhatBird and AllAboutBirds, or maps GPT-4 textual descriptions into attribute vectors with GPT-3.5.

4.7.2 Results

Table 4.8 contains the best zero-shot classification accuracy achieved by APN for each side-information source.

Table 4.8: CUB zero-shot learning with APN under different forms of side information. The real-valued attribute-vector baseline uses the ground-truth (GT) CUB attribute labels aggregated to class-level vectors. The table reports the best zero-shot accuracy over training epochs.

Description	Best ZSL accuracy [%]
CUB GT real-valued attribute label vectors	71.9
Top 2 attributes per class, binarised	50.4
Top 1 attribute per class, binarised	46.9
Thresholded by confidence	48.3
ChatGPT fills attribute slots	19.7
ChatGPT selects all factually correct attributes	16.5
ChatGPT attributes from Wikipedia context	14.1
Attributes derived from GPT-4 textual descriptions	29.4
GPT-4-based attributes + GPT-3.5 preprocessing	34.0
Attributes derived from WhatBird bird guides	40.4
Attributes derived from AllAboutBirds field guide	30.0

Ground-truth CUB attribute labels provide the strongest performance, with 71.9% zero-shot accuracy when used as full real-valued class-level attribute vectors. Binarisation and sparsification gradually reduce performance, indicating that a small set of salient attributes retains substantial discriminative power but does not match the full ground-truth real-valued CUB attribute label vectors.

Naive LLM attribute prompts perform much worse. Slot filling and "select all correct features" prompts yield accuracies below 20%. More carefully engineered pipelines close part of the gap: GPT-4 descriptions with GPT-3.5 attribute extraction reach 29.4%, GPT-4-based attributes with GPT-3.5 preprocessing reach 34.0%, and WhatBird-derived attributes reach 40.4%. High-quality domain text improves LLM-derived attributes, but manual CUB attributes remain substantially stronger.

The APN experiment gives a negative result for direct attribute replacement. LLM-derived attribute vectors perform best when high-quality domain descriptions constrain the extraction process, but they remain well below manual attributes.

4.7.3 Implementation Details

APN experiments use the Attribute Prototype Network architecture for zero-shot learning. The global path maps images into attribute space for classification, while local attribute-aware prototypes identify attribute evidence within convolutional feature maps. Training uses the standard APN losses for classification and attribute alignment.

The CUB adaptation keeps the visual architecture and training protocol fixed while changing only the side-information vectors. Ground-truth side information uses CUB class attributes, either as full real-valued class-level vectors or binarised variants. LLM-derived vectors come from four pipelines: slot filling, selecting the correct attributes from a candidate list, extracting from encyclopaedic bird resources, and mapping GPT-4 descriptions into the CUB attribute space. Table 4.8 reports the best zero-shot accuracy obtained for each side-information source.

4.8 Enhancing CLIP with GPT-4 and Visual Descriptions

Contrastive pre-training for large-scale vision-language models yields strong open-set image classification performance. CLIP [1] and ALIGN [79] optimise generalisable image-text representations from web-scale image-text pairs. Conventional supervised training uses a closed set of classes, whereas CLIP pre-training uses natural-language supervision and produces a joint embedding space that is not tied to a fixed label set. CLIP constructs a classifier by inserting the class name into a fixed prompt template, such as "a photo of {class name}". Prompt engineering adapts CLIP to a target domain by modifying this template, for example "a photo of class name, a type of bird". Prompt engineering remains sensitive to domain expertise and phrasing, so small changes to prompts can produce large performance differences.

Fine-grained datasets contain classes distinguished by subtle visual differences. Strong zero-shot transfer on such datasets requires either pre-training on the specific class with aligned textual labels or additional side information in the form of visually descriptive text (VDT). VDT denotes sentences that describe shape, size, colour, environment, patterns, and composition for the target class. The experiments test whether VDT improves CLIP zero-shot accuracy relative to the standard prompt.

The first experiment uses VDT from target-domain information to construct class-conditional prompts. The CUB evaluation uses domain-expert sentences about bird species, available in dataset annotations.

Domain-expert annotations are costly and time-consuming. GPT-4 removes the need to write each class-conditional visual description manually. The prompt asks

for visual cues such as shape, colour, structure, and composition. Generated VDT sentences are inserted into prompt ensembles, encoded with the CLIP text encoder, and aggregated into classifiers for zero-shot classification. Prompt ensembling reduces sensitivity to individual prompt phrasing.

A lightweight adapter selects and aggregates the most informative LLM-generated visual descriptions for each dataset. Few-shot adaptation is evaluated in the Base-to-New setting, where training uses base classes, and evaluation uses novel classes across a benchmark of 12 datasets.

4.8.1 Review of CLIP and CLIP-Adapter

CLIP pre-training aligns images and texts in a shared embedding space through contrastive learning. After pre-training, zero-shot image classification compares image embeddings with text embeddings derived from class prompts, without fine-tuning on the target dataset.

CLIP inference uses cosine similarity $s(f,t) = \frac{f^\top t}{\|f\|_2\|t\|_2}$. Given an image $\mathbf{x} \in \mathbb{R}^{H\times W\times C}$, where H, W, and C denote height, width, and channel count, the vision encoder maps the image to features $f \in \mathbb{R}^D$. The text encoder maps the caption to text features $t \in \mathbb{R}^D$.

The prompt template generates one sentence for each of K classes, and the text encoder maps these sentences to the classifier weight matrix $\mathbf{W} \in \mathbb{R}^{D\times K}$. Prediction probabilities are computed by multiplying the image feature f by $\mathbf{W}$ and applying a softmax:

$$f = \text{Backbone}(\mathbf{x}), \qquad p_i = \frac{\exp(\mathbf{W}_i^\top f/\tau)}{\sum_{j=1}^{K}\exp(\mathbf{W}_j^\top f/\tau)}. \tag{4.1}$$

CLIP [1] uses domain-specific prompt templates for zero-shot transfer, such as "A photo of a class-name, a type of bird" for bird images. Prompt ensembling averages predictions across multiple prompt variants, improving zero-shot classification accuracy. VDT prompt ensembles extend this principle by adding visual descriptions to class prompts.

CLIP-Adapter [33] applies a learnable MLP adapter to image features and/or text features for few-shot transfer. In the few-shot setting with K classes and N labelled images per class, the labelled set is $\{(x_{c,n}, y_c)\}$ with $c \in \{1,\dots,K\}$ and $n \in \{1,\dots,N\}$. The prompt template H maps a class name to a textual prompt, and the text encoder g embeds prompts into the joint feature space. The classifier prototypes are $\mathbf{W} = [g(H(\text{classname}_c))]_{c=1}^{K}$. Given image features f and text features $\mathbf{W}$, adapters A_v and A_t produce adapted features, for example $\tilde{f} = f + \lambda_v A_v(f)$ and $\tilde{\mathbf{W}} = \mathbf{W} + \lambda_t A_t(\mathbf{W})$, where A_v and A_t are MLPs and λ_v, λ_t are scaling coefficients.

The hyperparameters α and β weight pretrained CLIP representations and fine-

tuned adapter outputs to reduce overfitting in CLIP-Adapter. Logits follow Equation 4.1, and cross-entropy over the training set $\{(x_{i,k}, y_{i,k})\}_{i=1,k=1}^{N,K}$ optimises A_v and A_t.

The *All* setting evaluates few-shot transfer on a held-out split containing images from the same K classes used during training. The Base-to-New setting [2] trains on base classes and evaluates on disjoint new classes. This setting measures generalisation to unseen classes within the same dataset.

4.8.2 Simple Few-Shot Adapters for Visual Sentences

A simple adapter uses VDT to improve few-shot transfer with CLIP on target datasets. Similar to CLIP-A Text, a small set of learnable parameters is appended to the text-encoder outputs, and cross-entropy is used to train the adapter. CLIP-A-self uses a self-attention layer over embeddings of multiple sentences for each class and averages the resulting representations to obtain the final classification vector.

VDT-Adapter constructs M prompts by appending each GPT-generated sentence $t_{m,k}$ for class k to the template $H(\text{classname}(y_{i,k}), t_{m,k})$. The CLIP text encoder maps these prompts to $\mathbf{W}^{\text{sent}} \in \mathbb{R}^{D\times M\times K}$. Here, $H(\cdot)$ denotes the prompt-templating function, classname$(\cdot)$ maps a class label $y_{i,k}$ to its textual name, and D is the CLIP text-feature dimensionality.

Self-attention [14] operates over all visually descriptive sentences. During training, attention weights select and aggregate sentences relevant to class identification. Sentence-level classification embeddings $\mathbf{W}^s \in \mathbb{R}^{K\times M\times D}$ serve as keys, queries, and values for the self-attention block B_{self}. Mean pooling over the output tokens produces the per-class classification vector $\mathbf{W}^\star \in \mathbb{R}^{K\times D}$.

$$\mathbf{W}_{\text{avg},k} = \frac{1}{M}\sum_{m=1}^{M} \mathbf{W}^s_{m,k}, \tag{4.2}$$

$$\{\mathbf{W}^a_{m,k}\}_{m=1}^{M} = B_{\text{self}}\left(\{\mathbf{W}^s_{m,k}\}_{m=1}^{M}, \{\mathbf{W}^s_{m,k}\}_{m=1}^{M}, \{\mathbf{W}^s_{m,k}\}_{m=1}^{M}\right), \tag{4.3}$$

$$\mathbf{W}_{a\text{-mean},k} = \frac{1}{M}\sum_{m=1}^{M} \mathbf{W}^a_{m,k}, \tag{4.4}$$

$$\mathbf{W}^\star_k = \beta\mathbf{W}_{a\text{-mean},k} + (1-\beta)\mathbf{W}_{\text{avg},k}. \tag{4.5}$$

The adapted classifier weights $\mathbf{W}^\star \in \mathbb{R}^{D\times K}$ aggregate the most discriminative VDT sentences for the dataset. Equation 4.1 then gives class probabilities, and the predicted class is the maximum-probability category.

Few-shot training optimises only the weights of B_{self} with a cross-entropy objective. All other parameters remain frozen.

4.8.3 Experiments

The VDT study evaluates visual sentence ensembles in two settings. First, zero-shot evaluation compares VDT prompt ensembles with default CLIP prompts across 12 benchmark datasets. Second, the few-shot Base-to-New evaluation compares adapters that use visual prompts with other transfer methods.

Datasets

The evaluation suite contains 11 image recognition datasets from [31] and the CUB bird species dataset [42]. The suite comprises generic object classification datasets (ImageNet, Caltech101), fine-grained object datasets (OxfordPets, StanfordCars, Flowers102, Food101, FGVCAircraft), SUN397 for scene recognition, UCF101 for action recognition, DTD for texture classification, EuroSAT for satellite imagery, and CUB for bird species identification.

VDT-Adapter zero-shot transfer evaluates all classes across datasets. In the Base-to-New setting, VDT-Adapter follows CoCoOp [2] and splits classes equally into non-overlapping base and new sets. The CUB setting uses the standard 150-base and 50-new split from the ZSL and few-shot literature. CLIP-A-self uses the 16-shot setting to facilitate comparison with prior methods.

Baselines

The zero-shot evaluation compares visual sentence ensembles with CLIP default prompts. The CLIP prompts for each dataset are tuned on the test set in the standard evaluation protocol.

DCLIP [80] uses GPT-3 to generate VDT for zero-shot transfer and serves as the most direct prior baseline. CoOp [31] optimises static prompts, and CoCoOp [2] optimises dynamic prompts for base-to-new transfer. The evaluation also includes CLIP-A [33] because of its architectural similarity to CLIP-A-self, which isolates gains from visual sentences rather than generic text-feature adaptation.

Training Settings

The implementation builds on the CoOp and CLIP-Adapter codebases.

All comparisons use the CLIP ViT-B/16 backbone. Reported CoOp and CoCoOp results for all datasets except CUB come from their respective papers. CUB experiments follow the same practice: context length is set to 4 and context initialisation is set to "a photo of" for comparability. CLIP-Adapter experiments are re-run on ViT-B/16 because the original paper does not report this backbone. For all adapter models, including VDT-Adapter, only the residual ratio α is tuned.

CLIP-Adapter uses the MLP-on-visual-encoder variant, which gives the strongest performance.

4.8.4 GPT-Generated Visual Sentences Improve Zero-Shot Transfer

Results in Table 4.10 indicate that the CLIP-GPT prompt ensemble outperforms default CLIP prompts for zero-shot transfer. Average accuracy across 12 datasets improves by approximately 2 percentage points. Gains exceed 5 percentage points on specialised fine-grained datasets such as CUB, SUN397, EuroSAT, and DTD, and exceed 2 percentage points on Oxford Flowers and Oxford Pets. Default class-name prompts provide insufficient semantic coverage across several datasets, whereas visually descriptive sentences yield stronger text classifiers. ImageNet and Caltech101 also improve by more than 4 and 1 percentage points, respectively. Transfer performance matches CLIP on most of the remaining datasets, except for UCF101. Inspection of generated UCF101 sentences indicates that several descriptions contain temporal rather than purely visual information, which likely explains the drop. A prior study defines a six-dataset subset for the ViT-B/16 encoder [80]. On this subset, GPT-4-generated VDT yields larger improvements than GPT-3-generated VDT on specialised datasets such as DTD and EuroSAT.

Table 4.9: Zero-shot top-1 accuracy (%) with Large Language Model generated Visually Descriptive Text (VDT) on six datasets, compared with prior methods. VDT generated by GPT-4 yields higher accuracy than VDT generated by GPT-3 on specialised datasets such as DTD, Oxford Pets, and EuroSAT.

Methods	EuroSAT	Food101	DTD	Oxford Pets	CUB	ImageNet	Average
CLIP	47.69	85.97	43.09	89.07	54.70	64.51	64.17
DCLIP [80]	48.82	**88.50**	45.59	86.92	**57.75**	68.03	65.93
CLIP-GPT	**54.86**	86.43	**50.15**	**91.54**	57.43	**68.92**	**68.21**

Table 4.10: Results of zero-shot transfer on 12 datasets with ViT-B/16.

Methods	EuroSAT	Caltech101	Oxford Flowers	Food101	FGVC Aircraft	DTD	Oxford Pets	Stanford Cars	Sun397	UCF101	CUB	ImageNet	Average
CLIP	47.69	93.75	70.69	85.97	**24.81**	43.09	89.07	**65.55**	62.61	**67.54**	54.70	64.51	64.16
CLIP-GPT	**54.86**	**94.51**	**73.40**	**86.43**	23.42	**50.15**	**91.54**	65.01	**67.24**	65.51	**57.43**	**68.90**	**66.53**

Across the zero-shot CLIP-GPT experiments, GPT-generated visual descriptions improve default CLIP prompts on average, with larger gains on several specialised datasets. The lower results on the exception datasets coincide with descriptions that

emphasise temporal or contextual information rather than visually discriminative properties.

4.8.5 GPT-Adapters Improve Few-Shot Transfer Performance

Table 4.11 contains Base-to-New results for CLIP-A-self, CLIP, CoOp, and CoCoOp across 12 datasets. CLIP-A-self improves new-class accuracy over CoCoOp by 3 percentage points on average while maintaining base-class performance comparable to CoOp. CoCoOp uses a meta-network and an additional text-encoder forward pass, which increases training cost. On 16-shot ImageNet with ViT-B/16, CoCoOp requires up to 5 hours of training, whereas CLIP-A-self trains in about 10 minutes on an RTX 3090 GPU. Base-to-New generalisation is particularly strong on fine-grained and specialised datasets, with harmonic-mean gains on FGVC-Aircraft ($\sim$ 7.5%), EuroSAT ($\sim$ 7.4%), DTD ($\sim$ 5.8%), CUB ($\sim$ 4.3%), Flowers-102 ($\sim$ 4%), Stanford-Cars ($\sim$ 2.4%), and UCF101 ($\sim$ 2.4%). The reported gains show that VDT contributes useful side information in this few-shot transfer setting.

Table 4.11: Comparison of CLIP-A-self against CLIP, CoOp, and CoCoOp under the Base-to-New generalisation setting. For prompt-learning methods (CoOp and CoCoOp), prompts are learned from base classes in the 16-shot setting. Base and New report average top-1 accuracy on base and novel classes, respectively; H denotes the harmonic mean.

	Base	New	H
CLIP	68.45	73.89	71.05
CoOp	**82.39**	62.39	70.99
CoCoOp	79.35	71.89	75.37
CLIP-A	78.90	72.14	75.07
CLIP-A-self	82.12	**74.20**	**77.78**

CLIP-GPT zero-shot ensembles underperform the default CLIP prompt on FGVC-Aircraft, UCF101, and Stanford Cars. CLIP-A-self nevertheless outperforms CLIP-A and CoCoOp on these datasets in the few-shot transfer setting. Few-shot training optimises the self-attention mechanism to weight the most relevant VDT sentences, producing classifiers that transfer better to new classes.

The adapter results are strongest when a lightweight learned module selects among the generated descriptions instead of weighting every sentence equally.

4.8.6 Ablation Over Different GPT Models

OpenAssistant [81] and GPT-3.5 are evaluated against GPT-4 for VDT generation on CUB. Results in Table 4.12 indicate that performance improves with larger models. YuKoLA [82] associates larger language models with more accurate class

information and fewer hallucinations. OpenAssistant performs reasonably but produces inconsistent, noisy output that requires substantial post-processing. GPT-3.5 and GPT-4 produce concise dictionary-style sentences.

VDT-Adapter remains robust to noisy VDT side information. Base-to-New generalisation with OpenAssistant and GPT-3.5 approaches the GPT-4 result.

The ablation indicates that model quality affects zero-shot VDT generation, but the adapter reduces sensitivity to the exact choice of generator once descriptions contain sufficient visual signal.

Table 4.12: Comparison of GPT models for generating VDT side information. Larger models yield higher-quality VDT side information, but CLIP-A-self produces generalisable classifiers with smaller models, such as OpenAssistant. Default uses no side information. ZS denotes zero-shot accuracy; Base and New denote the seen and novel class splits; H denotes their harmonic mean.

Prompting	ZS	Base	New	H
Default	54.7	NA	NA	NA
OpenAssistant	56.0	78.3	69.8	73.80
GPT-3.5	55.7	78.1	70.6	74.16
GPT-4	**57.4**	**78.6**	**71.3**	**74.77**

Language Model Prompt Design

Prompt design experiments evaluate which information improves CLIP zero-shot domain transfer when appended to the class prompt.

CLIP [1] uses prompt design and prompt ensembling to improve zero-shot classification. Prompt ensembling aggregates predictions across multiple prompt variants to reduce variance. The CUB experiment compares visual and non-visual prompt ensembles using expert annotations. Zero-shot classification evaluates recognition without any class-specific training images.

Visual prompts use class-attribute vectors that describe colour, pattern, shape, and related properties for 28 annotated bird parts, yielding 312 attribute scores per class. The procedure selects the most salient attribute-value pair for each part to construct 28 Visual-GT prompts, such as "A photo of a Green Heron. The Green Heron has a greenish-black crown." Non-Visual-GT prompts use domain knowledge about vocalisations, migration, behaviour, and habitat, such as "A photo of a Green Heron. The Green Heron's call is a loud, harsh skeow."

The approach derives classification vectors for *Visual-GT* and *Non-Visual-GT* by averaging class-level sentence embeddings within the CLIP joint embedding space while respecting the 77-token limit of the text encoder. Results in Table 4.13 indicate

no improvement from *Non-Visual-GT* prompts relative to the default prompt, but *Visual-GT* improves accuracy by 4 percentage points.

Table 4.13: Comparison of visual and non-visual prompt ensembles for zero-shot domain transfer on the CUB dataset (Caltech-UCSD Birds-200-2011). GT denotes ground-truth class names; GPT denotes LLM-generated prompts.

Prompting	Default	Non-visual-GT	Visual-GT	Visual-GPT
Accuracy	54.7	53.0	57.7	57.4

Expert VDT improves CLIP zero-shot performance, but expert annotation does not scale easily. GPT language models enable scalable prompt generation through large-scale text pretraining and flexible instruction-following. GPT-4 generates visual descriptions for a target dataset, which then form prompt ensembles for CLIP.

Chain-of-thought prompting [83] motivates a two-stage generation strategy. First, the prompt asks for attributes needed to discriminate images among the K target classes. Second, the prompt asks for natural-language values of those attributes for every class.

Accuracy in the final row of Table 4.13 indicates that GPT-4-generated visual descriptions achieve performance comparable to expert-derived class-attribute descriptions. Across the benchmark suite, the procedure generates visually descriptive sentences in a scalable manner and uses them to construct prompt ensembles. A class-attribute vector denotes a set of human-annotated semantic properties for a class, and a prompt ensemble denotes a collection of prompts whose scores are aggregated for robustness.

4.9 Discussion and Summary

4.9.1 What Determines Whether Side Information Transfers

The experiments separate three properties that are often grouped together under "better text": the information in a description, the encoder that converts it to a vector, and the mechanism that aligns that vector with images. The results show that all three matter. Generated definitions can improve WordNet- or CLIP-graph agreement without improving image classification, because an intrinsically coherent text neighbourhood need not match the decision boundaries of the visual representation. Conversely, a short CLIP prompt can outperform a richer language-only description when its embedding already occupies a space trained jointly with images.

The CUB experiments make the role of representation format especially clear. CADA-VAE and APN were designed to consume consistent class-level attributes.

Full CUB vectors encode many complementary properties with a stable ordering and scale. Free-form generated descriptions may contain correct facts, but converting them into a small or noisy set of attribute slots removes information and changes the input distribution expected by these models. This explains why generated text can be useful yet remain below the full ground-truth attribute baseline. The limitation is not simply that the descriptions are too short or too long; it is that their information must be expressed in the form used by the downstream model.

VDT-Adapter provides the contrasting positive case. Its generated descriptions are encoded by CLIP, so text and images already share a multimodal representation, and the lightweight adapter can select among several candidate descriptions using labelled base-class examples. The method does not require generated prose to imitate a 312-dimensional expert attribute vector. It uses the prose in the form for which CLIP was trained, then learns which descriptions help the target task. Across the chapter, generated side information is therefore most effective when the encoder and downstream method preserve its visual content, rather than when text is treated as a direct substitute for any existing semantic input.

The four evaluations answer different questions. MiniImageNet tests whether projected text prototypes support recognition of unseen classes. CADA-VAE and APN test whether generated side information can replace CUB's expert attribute vectors. VDT-Adapter tests whether selecting from several generated descriptions can improve a CLIP classifier after adaptation. Together, the results show that generated text is most useful when the recognition method is designed to work with text, and is less effective when it must imitate structured expert attributes.

MiniImageNet and CUB results support three main observations. First, semantic-prior quality depends strongly on alignment between the text embedding and the visual domain, as seen with CLIP, and on prompt specificity, as seen with refined label phrases and ChatGPT captions. Second, definitions, captions, and attribute lists improve semantic graphs in truthfulness experiments, and some gains transfer to downstream image recognition. Third, experiments with CLIP-GPT prompts and VDT-Adapter indicate that visually descriptive sentences from a language model can strengthen CLIP-based zero-shot and few-shot transfer when used through prompt ensembles and a lightweight adapter. Naive LLM outputs can still reduce performance, especially in generalised ZSL settings.

Text-only resources improve prototype geometry, but the largest recognition gains occur when generated visual descriptions are encoded by CLIP and selected by VDT-Adapter. Text-only embeddings do not replace the benefit of image-text pre-training.

LLM-generated side information improves CLIP prompts and VDT-Adapter, but it does not outperform full CUB ground-truth attributes.

Chapter 5

Spatial-Semantic Alignment for Fine-Grained Classification

5.1 Introduction

Large vision–language models, such as Contrastive Language-Image Pre-training (CLIP), align visual and textual features in a shared embedding space. These models achieve strong performance across a broad range of object recognition tasks. Still, accuracy drops for fine-grained classification tasks, such as bird species identification in the Caltech-UCSD Birds 200 (CUB) dataset [42]. Fine-grained tasks depend on local, specific parts such as beak shape or wing pattern.

Chapter 4 introduced the Visually Descriptive Text Adapter (VDT-Adapter). VDT-Adapter adapts a vision–language model to a downstream task by adding lightweight learnable modules on top of global image embeddings. Specifically, it uses a large language model (GPT-4) to generate multiple visually descriptive sentences for each class, encodes these sentences with the CLIP text encoder, and applies a self-attention adapter to aggregate them into class prototypes. This design works well across several low-shot benchmarks, but the global pooling step acts as an information bottleneck, discarding spatial detail. Spatial detail may help to separate fine-grained classes that differ only in localised attributes.

Large language models such as GPT-4 [73] can generate detailed, part-level descriptions for fine-grained categories. We test whether aligning image patches with part-level descriptions improves transfer to unseen classes. For example, suppose training encourages the embedding of a patch that covers a beak to move closer to the text description "short, conical, black beak". In that case, the representation may generalise better to new classes.

In this chapter, we investigate whether aligning spatial visual features with part-level text descriptions can improve spatial transfer (the ability to generalise spatial

and part-level representations learned on base classes to unseen new classes). We pursue two complementary directions:

1. **Training-free interventions.** These methods keep CLIP parameters fixed. Visual prompting (see Section 2.8.2 for background), inference-time attention modifications based on CLIP Surgery [21], and refinement of text prompts using a large language model are evaluated.

2. **Trainable spatial-semantic alignment.** In these methods, a spatial adapter is added on top of the CLIP patch tokens. The adapter is trained with fine-grained supervision objectives that map image parts to part descriptions generated by a large language model. Objectives range from simple contrastive losses to classification tasks in which the correct part description must be selected from a pool of candidates across all classes.

The experiments in this chapter support one main conclusion: the tested contrastive embedding methods do not yield effective spatial transfer in the low-shot fine-grained setting. Visual prompting and text refinement often reduce accuracy. Direct training for part–text alignment improves the local objective on base classes, but does not improve recognition of new classes. The analysis identifies limited separation between fine-grained descriptions in the CLIP text embedding space as one possible constraint. Some linguistically distinct descriptions map to vectors with small angular margins, which weakens the supervision available for learning transferable local features.

5.2 Methodology

We adopt a few-shot transfer learning setup on the CUB dataset. In K-shot learning, the model receives K labelled examples per class during training. The dataset is partitioned into base classes, which the model uses for training, and new classes, which are held out entirely for evaluation. The goal is to learn a classifier from base classes that generalises to new classes. This base-to-new split follows the protocol in Chapter 4 and allows direct comparison across chapters.

The remainder of this chapter is organised as follows. Section 5.2.1 describes training-free interventions, *i.e.* no adapter training, and Section 5.2.2 introduces trainable spatial adapters and the methods used to supervise part–text alignment.

5.2.1 Training-Free Interventions

Before introducing new trainable components, we test whether a frozen CLIP model already encodes useful fine-grained information that simple interventions can use.

We consider three categories of intervention: visual prompting (modifying the input image), text refinement (modifying the class descriptions), and attention modification (changing how internal representations are aggregated during inference).

Visual Prompting

We first test whether simple changes to the input image can emphasise relevant anatomy. As discussed in Section 2.8.2, drawing a red circle around an object can increase the contribution of that region to CLIP's image–text similarity while preserving global context, achieving strong results on referring expression comprehension [18]. The hypothesis here is that CLIP may already encode fine-grained part information, but global pooling of the spatial embeddings dilutes it. Encircling the relevant region could increase the influence of local fine-grained details.

We apply image-level augmentations based on ground-truth part bounding boxes:

- **Red bounding box.** We draw a red box around anatomical regions such as the head or wing, following the visual prompting approach in [18].
- **Background blurring.** We blur image regions outside the target part, following the Fine-Grained Visual Prompting (FGVP) framework [52]. FGVP demonstrates that blurring regions outside a target mask outperforms coarser visual prompts such as coloured boxes or circles. The blur operation reduces reliance on weakly related pixels while preserving spatial coherence between the target region and its surrounding context. This approach outperforms prior visual prompting methods by 3–5 percentage points on average in referring expression comprehension [52]. We evaluate this technique on fine-grained bird classification by using ground-truth part annotations from the CUB dataset to define the non-blurred regions. See an example in Figure 5.1.

Language Model Guided Text Refinement

We also test whether refining the text prompts can improve zero-shot classification. In zero-shot classification, the model receives no labelled training examples for the target classes; instead, it relies solely on class names or descriptions encoded by the text encoder. We use a large language model to generate these descriptions and then feed the refined text into CLIP:

- **Content word filtering.** We strip each description to its core nouns and adjectives to remove stopwords that do not change the meaning.
- **Language model weighting.** We assign weights to specific parts, for example, a higher weight for the beak, based on language model analysis of easily confused species-pairs.

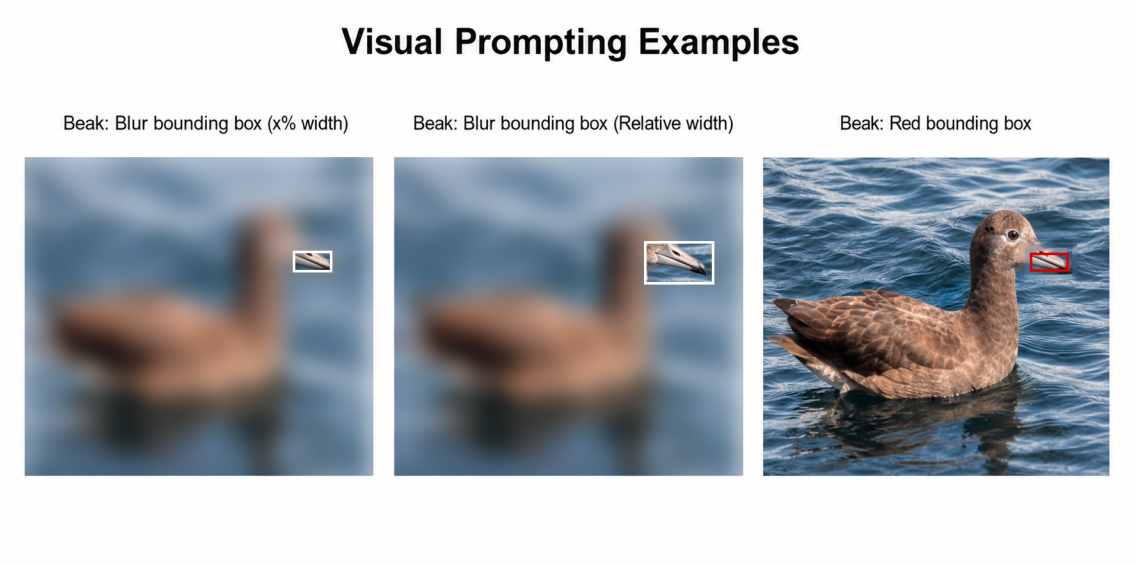


Figure 5.1: Visual prompting examples corresponding to the interventions in Table 5.1. The left panel shows background blurring, with only a small beak-centred, fixed-width bounding box remaining sharp. The middle panel shows the relative-width background-blur condition, in which a slightly larger beak-centred bounding box remains sharp. The right panel shows the unblurred image with a tighter red bounding box for the beak. Ground-truth part coordinates from the CUB dataset define the target regions, following the FGVP approach [52].

CLIP Surgery at Inference Time

Finally, we explore CLIP Surgery [21], which modifies the attention mechanism in the Vision Transformer backbone.

In a standard Vision Transformer, the image is divided into patches, each of which is encoded as a token. A special `[CLS]` (class) token is prepended to the sequence; after passing through multiple transformer layers, this token aggregates information from all patches and serves as the global image representation. Aggregation occurs through the self-attention mechanism, which computes attention weights from query and key projections of each token. The `[CLS]` token's query attends to patch keys, producing a weighted sum of patch values. This query–key interaction can smooth out local details, because globally salient regions dominate attention weights.

CLIP Surgery replaces query–key interactions in the final transformer layers with value–value interactions. Instead of computing where to attend (query–key), the method directly compares token contents (value–value). This design aims to preserve spatial distinctiveness by reducing the smoothing effect of learned attention patterns.

We integrate CLIP Surgery into a two-stage inference pipeline:

1. **Global stage.** We use CLIP global features to retrieve the top K candidate classes for each image. This step filters the label space based on coarse global

features.

2. **Local stage.** We re-rank these K candidates based on the alignment between spatial patch tokens (from CLIP Surgery) and part descriptions. Re-ranking allows fine-grained details to change the final prediction order (most likely based on the highest cosine similarity first).

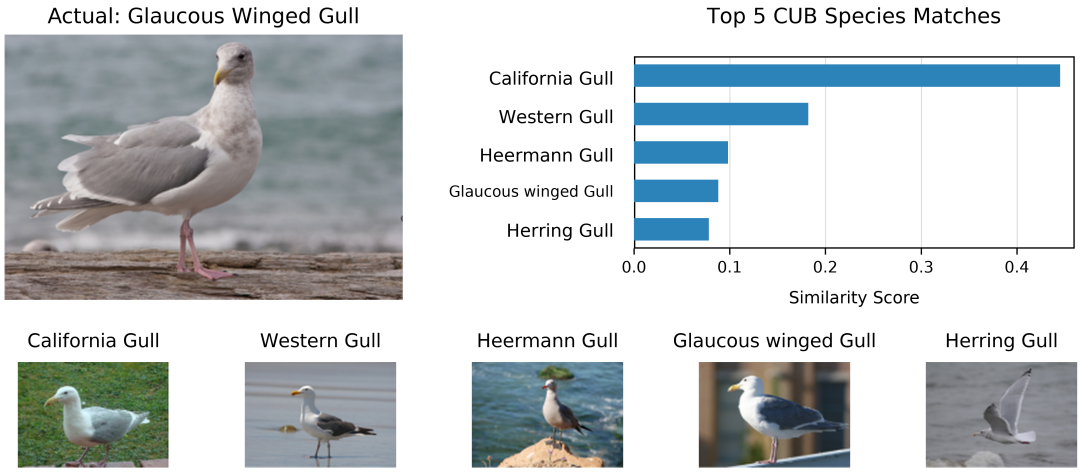


Figure 5.2: Fine-grained classification challenge on the CUB dataset. Top left: A Glaucous-winged Gull image. Top right: CLIP's top-5 predictions by similarity score, with California Gull ranked highest despite being incorrect. Bottom: Example images from each of the top-5 candidate species, illustrating their visual similarity. All five species share similar body shape, colouration, and posture, differing primarily in subtle details such as wingtip pattern, bill markings, and eye colour.

Figure 5.2 visualises the first stage of the two-stage inference pipeline. Although CLIP's global features misrank the correct species (Glaucous-winged Gull appears fourth), it successfully retrieves a shortlist of visually similar candidates across all 200 classes in the CUB dataset. We therefore use global CLIP embeddings to retrieve candidates and part-level scores to re-rank them. If a part-level classifier could reliably distinguish bill colour or wingtip pattern, for example, it could re-rank the candidates to place the correct class at the top.

5.2.2 Trainable Spatial Architecture

The training-free interventions above do not require parameter updates but cannot learn task-specific representations. We now introduce trainable components that extend the VDT-Adapter framework to exploit spatial information. The backbone remains a frozen CLIP ViT-B/16, and we augment it with learnable spatial adapters.

5.2.3 Visual Stream with Spatial Tokens

The CLIP image encoder (ViT-B/16) processes a 224×224 image by dividing it into a grid of 16×16 pixel patches, resulting in a sequence of $N = 196$ patch tokens (14×14 grid) plus a special class token. The internal Vision Transformer width is 768. After the final layer normalisation, the implementation applies CLIP's learned visual projection to each retained token, mapping the 768-dimensional transformer states into the 512-dimensional joint image–text space. Let $d = 512$ denote this projected dimension. Each projected patch token $z_i \in \mathbb{R}^d$ encodes the visual content of patch i, and $z_{cls} \in \mathbb{R}^d$ is the projected class token that aggregates global information:

$$Z = [z_{cls}, z_1, \ldots, z_{196}] \in \mathbb{R}^{197 \times d}$$

Standard adaptation methods (including the VDT-Adapter) use only the global z_{cls} token, which aggregates information from the entire image. This aggregation acts as an information bottleneck: although the 196 patch tokens preserve spatial layout and local texture, the single z_{cls} vector cannot represent all this detail. Fine-grained distinctions (*e.g.* whether a breast is spotted or streaked) may be lost in this representation. To recover spatial information, our modified architecture extracts the full sequence of spatial tokens $z_1, \ldots, z_N$ before global pooling. We introduce a Visual Adapter (a lightweight self-attention mechanism) that operates on these spatial tokens to produce adapted features $\hat{Z}_{spatial}$, preserving the spatial grid structure.

Fine-grained supervision requires mapping image regions to part descriptions, but part visibility varies across images: some images show only the head and back, while others reveal legs or tail. This variability complicates standard batch processing, which expects a fixed number of part–text pairs per image.

Using ground-truth part coordinates, we extract only the visible body parts for each image. For a visible part (e.g., "head"), we identify which patches cover that region. For instance, in Figure 5.3, the red annotation points mark the head locations labelled by the human annotator. The original ground truth data contains many body part annotations (*e.g.* eyes, crown, head, feet, etc.) that we roll up into six primary body parts. We select all spatial patch embeddings that contain point annotations. We average these patch embeddings to form a single visual part vector per primary part $\bar{z}_{\text{head}}$ (in this case, multiple patches covering the bird's head region), which we then pair with the corresponding text embedding for contrastive training. This ensures that the loss is calculated only on valid, visible regions.

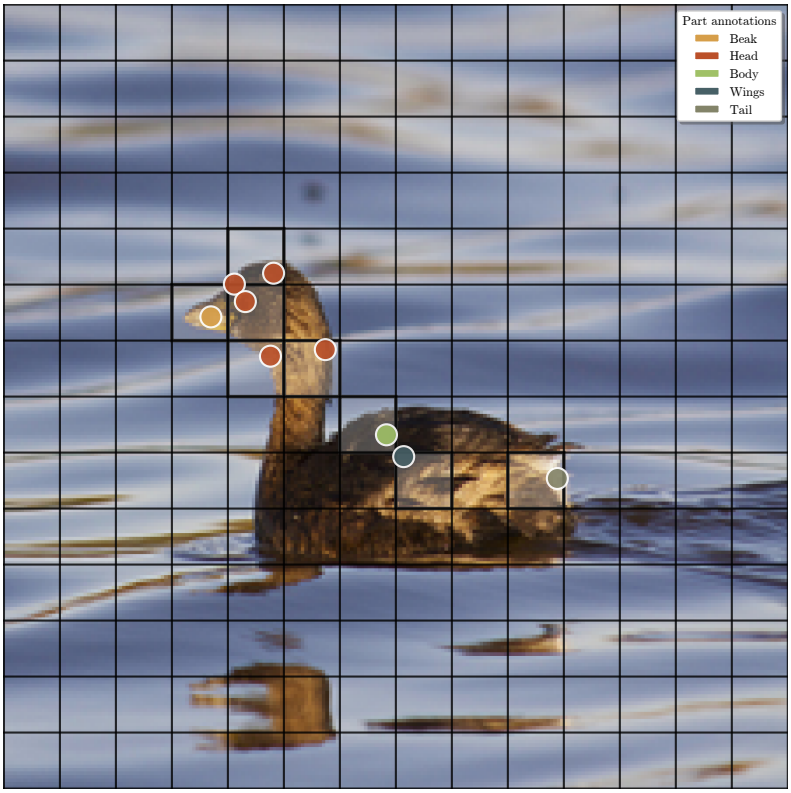


Figure 5.3: Visualisation of the packing mechanism. The image shows a bird from the CUB dataset with a 14×14 patch grid overlay. Coloured circles indicate ground-truth part locations (beak, head, body, wings, tail, legs). Light grey regions highlight patches selected for each visible part. Black boundary lines mark patch edges. The mechanism extracts only patches covering visible anatomical regions, ensuring contrastive training uses valid spatial correspondences.

5.2.4 Adapter Architectures

To modify both text and visual features, we employ two types of lightweight adapter modules. Recall that $d = 512$ is the CLIP embedding dimension for ViT-B/16.

- **Linear Adapter.** A two-layer MLP with a bottleneck: the input dimension d is projected down to $d/4 = 128$, passed through ReLU activation, and projected back to d. This adapter is applied to global text embeddings to adapt them to the target domain while preserving the semantic structure of the CLIP space.

- **Self-Attention Adapter.** This module wraps a multi-head attention block. It accepts a sequence of embeddings (either text descriptions or spatial visual tokens) and returns a sequence of the same shape. The adapter includes a residual interpolation parameter $\gamma \in [0, 1]$ that blends the attention output with the original input:

$$\hat{z} = \gamma \cdot f_{\text{attn}}(z) + (1 - \gamma) \cdot z,$$

  where f_{attn} denotes the attention transformation. When $\gamma = 0$, the adapter has no effect; when $\gamma = 1$, the output is fully determined by the attention block. Intermediate values allow the model to blend learned and pre-trained representations.

In our spatial alignment experiments, we use the Self-Attention Adapter on the visual side. It operates on the $N = 196$ spatial tokens extracted from the CLIP vision transformer. The adapter applies self-attention to these tokens and then averages the results to produce a spatially aware global feature. This allows the model to learn a pooling strategy optimised to align with GPT-4 text descriptions, rather than relying on the fixed global pooling used in the original CLIP architecture.

5.2.5 Spatial Alignment Objectives

We introduce a fine-grained loss $\mathcal{L}_{fg}$ to enforce local semantic consistency between visual patches and part descriptions. Let $\bar{z}_p \in \mathbb{R}^d$ denote the visual embedding for part p (obtained by averaging the relevant patch embeddings), and let $t_{c,p} \in \mathbb{R}^d$ denote the text embedding for the description of part p in class c. The fine-grained contrastive loss is:

$$\mathcal{L}_{fg} = -\frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \log \frac{\exp(\bar{z}_p \cdot t_{c,p}/\tau)}{\sum_{c'} \exp(\bar{z}_p \cdot t_{c',p}/\tau)}, \tag{5.1}$$

where $\mathcal{P}$ is the set of visible parts in the batch, τ is a temperature parameter, and the sum in the denominator runs over all classes c' that have a description for part p.

The experiments also explore *Supervised Contrastive Loss* (SupCon) [7] as an alternative. Unlike the standard CLIP loss, which treats each image-text pair as a unique class within a batch, SupCon uses class labels to pull together all embeddings (image and text) belonging to the same class while pushing apart those from different classes. This builds richer positive sets and may improve clustering of fine-grained concepts.

In our primary configuration, we minimise the contrastive loss between each visual part embedding $\bar{z}_p$ and its corresponding text embedding $t_{c,p}$. This treats the problem as a local alignment task: the visual "beak" region should align with the text "short conical beak," regardless of which other parts are visible.

Standard contrastive learning often relies on easy negatives, for example, distinguishing a beak from a wing, which differ obviously in texture. Fine-grained classification, however, requires distinguishing subtle variations within the same part category. To enforce this harder objective, we formulate an *all-parts classification task*: the correct description for part p must be selected from a candidate pool containing the unique descriptions of that part from all classes in the dataset. This objective requires distinguishing between "short conical beak" (Sparrow) and "long curved beak" (Curlew), thereby providing hard negative supervision.

We combine the global and fine-grained objectives:

$$\mathcal{L} = (1 - \lambda)\mathcal{L}_{fg} + \lambda\mathcal{L}_{global} \tag{5.2}$$

Here, $\mathcal{L}_{global}$ is the standard cross-entropy loss over class logits used by VDT-Adapter (i.e., the negative log-likelihood of the correct class given the softmax over image–text similarities), and $\lambda \in [0, 1]$ controls the trade-off between global and fine-grained supervision. When $\lambda = 1$, the model uses only global supervision; when $\lambda = 0$, it uses only fine-grained supervision.

5.3 Experiments and Results

We present results on the CUB dataset. For trainable methods, we use a 16-shot protocol: the model learns from 16 labelled examples per class during training. For zero-shot and inference-time interventions, we evaluate on the full test set without any training examples.

5.3.1 Training-Free Intervention Results

This subsection evaluates the training-free interventions from Section 5.2.1. We first test whether simple input manipulations can improve the CLIP zero-shot baseline of 55.25% accuracy on CUB.

5.3.2 Visual Prompting and Text Refinement

Table 5.1 summarises the effect of visual prompting and text refinement. Visual prompting consistently reduces performance. Drawing attention to parts via red boxes yields accuracies around 50%, which is lower than the baseline. The lower accuracy shows that, in this setting, the artificial markers disrupt CLIP's pre-trained image representation rather than directing it usefully to the selected part.

Background blurring causes a larger drop, to 36.67% accuracy. This result shows that CLIP depends on the unmodified global image context for fine-grained zero-shot recognition on CUB.

Text refinement produces a similar pattern. Removing bird names (Row 6 in Table 5.1) reduces accuracy to 7.36%. The large drop shows that class names contribute substantially in this prompt configuration; it does not establish how CLIP uses the remaining descriptive attributes. Language-model-based weighting of parts also fails to outperform simple averaging. Neither the tested visual modifications nor the tested text refinements improve the strongest zero-shot baseline on CUB.

Table 5.1: Zero-shot performance with visual prompting and textual interventions. Visual prompting reduces performance, and removing class names leads to low accuracy.

Ref	Method	Accuracy (%)
1	Baseline: CLIP Zero-shot	55.25
2	Baseline: Avg of GPT sentences	57.60
Visual Prompting		
3	Red bounding box (20% width) + GPT sentence parts	50.52
4	Blur bounding box (20% width) + GPT sentence parts	45.74
5	Blur bounding box (Relative width) + GPT sentence parts	36.67
Textual Ablations		
6	Avg of GPT sentences (names removed)	7.36
7	LLM Weighting (Median weights)	29.36

5.3.3 Inference-Time CLIP Surgery

We evaluate CLIP Surgery to test whether improved attention preservation helps fine-grained recognition. Table 5.2 shows that CLIP Surgery maintains the global

zero-shot baseline at about 55% accuracy. In contrast, relying on part-based alignment alone ($\alpha = 0$) yields only 9.01% accuracy. This result is comparable to the 7.36% accuracy obtained when removing class names from text prompts (Row 6 in Table 5.1). Both of the latter experiments lack class-level supervision and fall far short of the baseline that uses class names. The spatial similarity signal from CLIP Surgery alone is too weak to serve as a standalone classifier.

Blending the fine-grained score with the global score through margin gating (*i.e.* the two-stage pipeline) achieves 58.26% accuracy, nearly matching the VDT-Adapter zero-shot baseline of 58.41%. The fine-grained spatial score does not provide additional information to correct global errors; it mainly adds noise or simply repeats the global prediction.

Table 5.2: Inference-time spatial–semantic alignment results with CLIP Surgery.

Method	Accuracy (%)
CLIP Zero-shot	55.33
VDT-Adapter ZS (Baseline)	58.41
Pure Part-Based Inference	
CLIP-Surgery (Spatial & Text Tokens)	9.01
CLIP-Surgery (Top-5 Re-ranking Only)	34.85
Blended Inference	
Weighted Tokens ($\alpha = 0.9$)	55.64
Margin Gating (threshold 0.1)	58.26

5.3.4 Trainable Spatial Alignment Results

We evaluate trainable adapters with spatial losses to test whether learned part–text alignment improves upon the global baseline.

Impact of Spatial Loss Weighting

We first investigate adding a spatial alignment loss to the frozen vision backbone, without yet introducing a trainable vision adapter. In this setting, we train only the text adapter but regularise learning with a CLIP-style contrastive loss between the fixed spatial tokens and the adapted text descriptions.

We parameterise this experiment with λ from Equation 5.2. When $\lambda = 1$, the model uses only global cross-entropy (the baseline); lower values of λ increase the relative weight of the spatial alignment term.

Table 5.3 summarises the results. The baseline ($\lambda = 1.0$, no spatial loss) achieves 77.8% on base classes and 61.6% on new classes. Introducing a modest spatial

alignment term ($\lambda = 0.4$) yields minor but consistent improvements, raising base accuracy to 79.4% and new accuracy to 62.6%. However, relying solely on spatial loss ($\lambda = 0.0$) leads to model collapse: accuracy drops to 1% across both splits. Collapse occurs because the spatial loss alone encourages part–text alignment but does not directly supervise class predictions, so the model fails to learn class-discriminative features.

Table 5.3: Effect of spatial loss weighting λ on base and new class accuracy (frozen vision encoder). $\lambda = 1.0$ corresponds to the baseline (no spatial loss). Lower λ increases the weight of the spatial alignment term.

Spatial Loss Weight (λ)	Base Acc (%)	New Acc (%)
1.0 (Baseline)	77.8	61.6
0.5	78.6	62.5
0.4	**79.4**	**62.6**
0.0 (Pure Spatial Loss)	1.0	1.0

We also observe that the language adapter is critical. Configurations that remove the language adapter and rely solely on the vision adapter with spatial loss fail to recover strong classification performance, confirming that the language adapter is essential for adapting the CLIP text space to the fine-grained bird taxonomy.

The Trade-Off Between Base and New Accuracy

Table 5.4 presents the performance of spatial losses in combination with the global VDT-Adapter. The baseline VDT-Adapter reaches 77.8% accuracy on base classes and 61.6% on new classes.

When a vision adapter is trained only with the spatial loss (Row 4), base accuracy drops to 61.0%, while *new* accuracy rises to 73.0%. The two metrics therefore move in opposite directions in this configuration. Joint losses with moderate spatial weight ($\lambda = 0.4$–0.5) give small improvements on both base and new classes in the reported runs.

Table 5.4: Comparison of global and spatial alignment objectives (16-shot). Base refers to seen classes, and New refers to unseen classes.

Method	Base Acc (%)	New Acc (%)
VDT-Adapter (Global Baseline)	77.8	61.6
+ Spatial CLIP Loss ($\lambda = 0.5$)	78.6	62.5
+ Spatial CLIP Loss ($\lambda = 0.4$)	**79.4**	**62.6**
Vision Adapter + Spatial Loss Only	61.0	73.0

5.3.5 Ablation Studies

We perform detailed ablations to understand the architectural and data factors limiting performance.

Adapter Architecture: MLP vs. Self-Attention

In the fine-grained alignment experiments, the choice of adapter architecture significantly affects how well the training data can be fitted. We compared Self-Attention adapters against Multi-Layer Perceptron (MLP) adapters for the task of aligning spatial patches to part descriptions.

- **Self-Attention Adapters:** Self-Attention adapters did not fit the training set closely, saturating at approximately 20-30% fine-grained training accuracy. We hypothesise that optimisation is more difficult for Self-Attention adapters in this low-data regime, though we did not conduct a detailed analysis.
- **MLP Adapters:** In contrast, simple MLP adapters operating on each token independently proved much easier to train. With sufficient depth (3 layers) and width, MLP adapters achieved $> 98\%$ fine-grained accuracy on the training set.

The MLP has enough capacity to fit the training pairs, while the lower test accuracy shows that this fit does not transfer to the test data (test accuracy saturated around 70%).

Integration of Spatial Features

We investigate how best to combine the learned spatial features with the pre-trained global CLIP representation. We compare three strategies for constructing the final image embedding:

1. **Replacement:** Discarding the CLIP CLS token and using only the averaged output of the spatial adapter.
2. **Pooling:** Including the CLS token in the spatial adapter's input pool.
3. **Residual:** Keeping the frozen CLIP CLS token and adding the averaged output of the spatial adapter as a residual correction.

The *Residual* configuration outperformed the others, particularly on new classes (achieving $\sim$ 68.8% accuracy compared to $\sim$ 58% for replacement strategies in comparable runs). In these runs, retaining the frozen global CLS token gives higher new-class accuracy than replacing it, while adding the spatial output as a residual performs best among the three tested integration strategies.

Impact of Cropping

Finally, we evaluated the baseline VDT-Adapter's sensitivity to input preprocessing. Standard random resized cropping reduces base accuracy from $\sim 78\%$ (uncropped) to $\sim 68\%$ (cropped). Cropping changes the visible context, object scale, and image statistics together, so this experiment does not identify which change accounts for the lower accuracy.

5.4 Discussion

The experiments show that a straightforward spatial task can be defined and fitted by an adapter using base classes. However, this fit does not translate to better recognition of unseen classes. Two observations are relevant to this gap.

5.4.1 Limited Separation Between Text Embeddings

In probing experiments conducted as part of this study, we computed pairwise cosine similarities between CLIP embeddings of GPT-4-generated part descriptions. Some linguistically distinct descriptions (e.g., "brown with black bars" vs "streaked brown upperparts") have cosine similarities above 0.95. Table 5.5 reports the broader CLIP-LabelText distribution, with a mean pairwise similarity of 0.62, a standard deviation of 0.11, and a 90th percentile of 0.74. For the fine-grained bird descriptions, similarity values often exceed 0.90.

Table 5.5: Summary statistics of pairwise cosine similarity for CLIP text embeddings. Fine-grained bird descriptions cluster tightly in embedding space.

Embedding	Dim	Mean	Std	P50	P90	Max
CLIP-LabelText	512	0.624	0.110	0.649	0.737	0.942

These high similarities leave small angular margins between some part descriptions. In the tested contrastive objective, such margins provide less separation between candidate text targets and may make part-level discrimination more difficult.

5.4.2 Diffuse Patch–Token Similarity

In the inspected CLIP Surgery maps, activations associated with one annotated part sometimes extend into nearby regions with similar texture. This qualitative observation motivates the denser comparison below but does not establish a general limitation of contrastive models.

To investigate the lack of fine-grained alignment, we visualise dense, scaled similarity maps between projected image patches and text tokens. The displayed values are CLIP similarity logits: the cosine similarities after normalisation, multiplied by the scale used in the visualisation. They are therefore not raw cosine values, which must lie between -1 and 1. Figure 5.4 shows the scores between patches covering the head region (rows) and individual text tokens from the description "black with purplish blue sheen, red facial skin and small crest" (columns). The orange boxes highlight semantically meaningful colour tokens: "black" and "red".

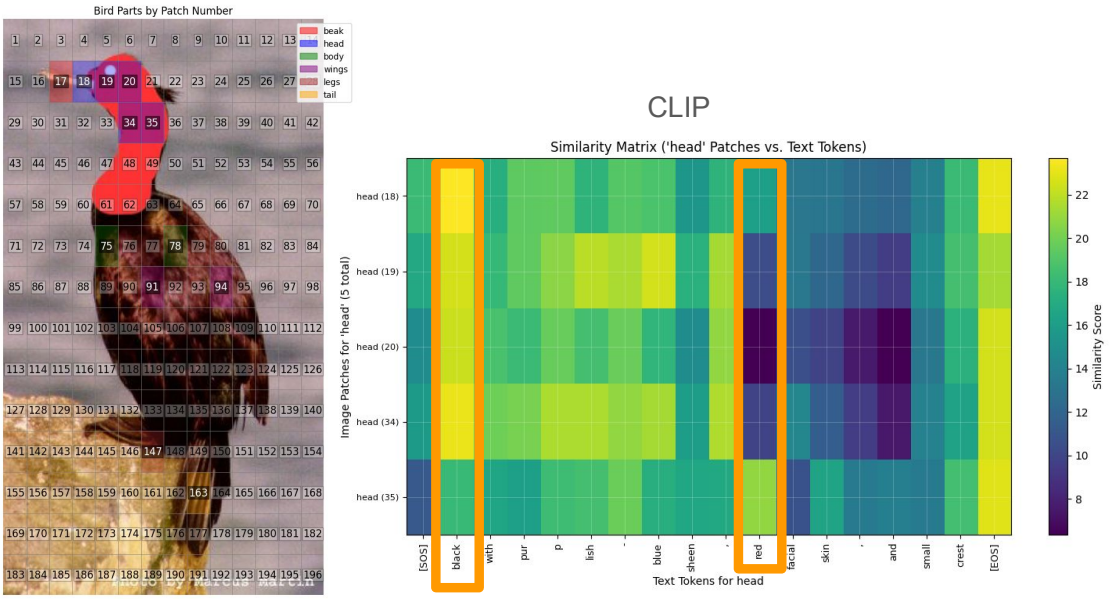


Figure 5.4: Dense scaled-similarity map for standard CLIP. Left: A random bird image from the CUB dataset with a synthetically modified red head. The image has a 14×14 patch grid overlay depicting the spatial token boundaries from the CLIP vision transformer. Coloured regions indicate ground-truth part annotations (beak in red, head in blue, body in green, wings in purple, legs in orange, tail in yellow). Right: Scaled similarity logits between the five head patches (rows) and text tokens from the head description (columns). Orange boxes highlight the "black" and "red" tokens. All head patches show similar activation patterns across tokens, indicating that CLIP does not localise colour-specific information to the patches that actually contain those colours.

In standard CLIP (Figure 5.4), all five head patches exhibit similar similarity profiles across text tokens. The "black" token (left orange box) activates moderately across all patches, as does the "red" token (right orange box), although the head has been modified to be red. For this example, the five selected head patches have similar profiles across text tokens and provide little evidence of patch-specific colour separation within the selected head region.

Figure 5.5 shows the same analysis using CLIP Surgery on the synthetically modified image. With CLIP Surgery, the "red" token activates more strongly over the modified head patches and the "black" token weakens. The displayed scaled sim-

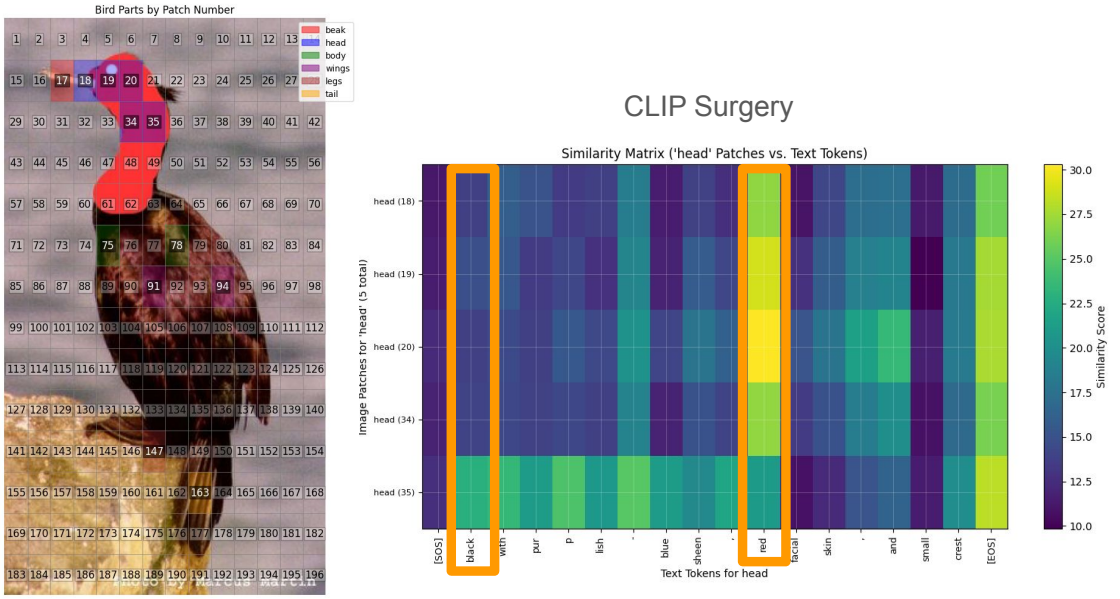


Figure 5.5: Dense scaled-similarity map for CLIP Surgery on the same synthetically modified image. The orange boxes again highlight the "black" and "red" tokens. Relative to standard CLIP (Figure 5.4), the "red" token has higher scores across the artificially reddened head patches, while the "black" token scores are lower. This example shows greater colour-token differentiation after CLIP Surgery on the synthetic modification.

ilarity logits also rise from approximately 22 to 30; these values should be compared within the map and not interpreted as raw cosine similarities. We therefore tested whether this effect improves fine-grained classification.

Figure 5.6 provides a direct side-by-side comparison of CLIP and CLIP Surgery similarity maps on an *unmodified* natural image from the CUB dataset. The orange boxes highlight the initial colour description tokens. In standard CLIP (left), the activation pattern is relatively uniform: all head patches respond similarly to the sequence "black with purplish...". In CLIP Surgery (right), the pattern is marginally more structured, with head patch 18 showing slightly stronger activation for "black" and "with". The difference between methods is smaller on this unmodified image than on the synthetic image with artificially saturated colours (Figure 5.5). The inspected patches do not show consistent patch–token separation for the subtler natural colours.

These visualisations are consistent with the weak transfer results but do not by themselves identify the cause. The quantitative experiments show limited separation in both text targets and patch-level scores under the tested setup.

5.4.3 Overfitting to Seen Part–Text Pairs

We test the all-parts objective, in which the vision adapter must distinguish the correct part description from a unique list of part descriptions for all bird classes in

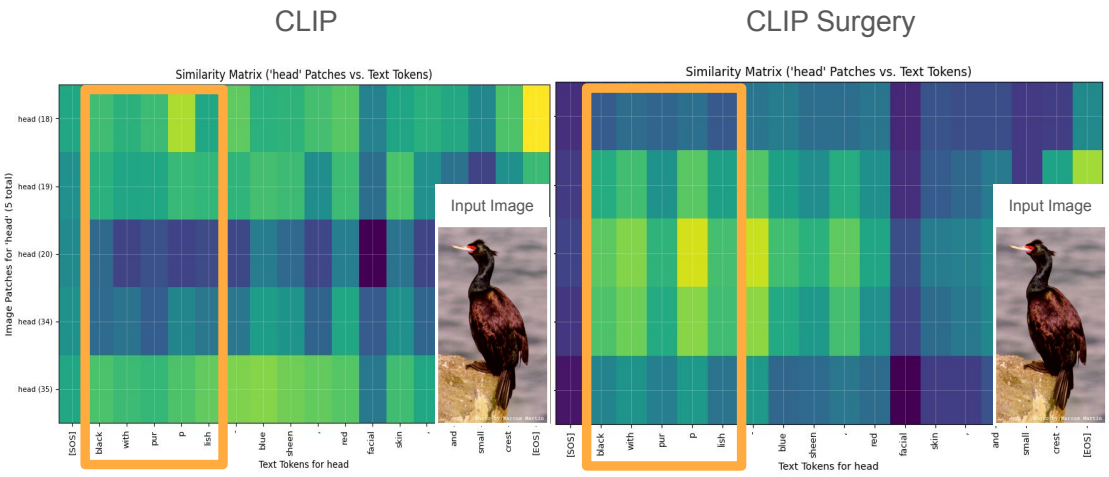


Figure 5.6: Side-by-side comparison of dense scaled-similarity maps for CLIP (left) and CLIP Surgery (right) on an *unmodified* natural image. Orange boxes highlight the first five text tokens ("black", "with", "pur", "p", "ish"). In standard CLIP, all head patches show relatively uniform activation across these tokens. CLIP Surgery produces slightly more differentiated patterns, but the effect is less pronounced than on the synthetic image (Figure 5.5). Neither method achieves clean patch-token alignment on natural images where colour boundaries are subtler.

CUB. Table 5.6 highlights a key finding: the adapter fits this task on the training set (75.2% fine-grained accuracy) but does not generalise to the test set (28.4%) or improve global classification.

The train–test gap is consistent with overfitting to the specific part–text pairs of seen classes. The fitted representation associates the visual "Red-winged Blackbird wing" with its specific text embedding, but does not transfer this association to new species under the tested protocol.

Table 5.6: Overfitting to seen part–text pairs. The adapter fits fine-grained alignment on training data (high Train FG Acc) but fails to generalise (low Test FG Acc).

Method	Train FG Acc	Test FG Acc	Global Base	Global New
Global Baseline	-	-	61.0	73.0
All-Parts FG ($\tau = 0.1$)	**75.2%**	28.4%	61.0	73.0
All-Parts FG (Surgery)	30.2%	17.4%	62.5	42.7

5.4.4 Oracle Analysis

To determine whether the limitation lies in the rescoring strategy or in the quality of the spatial model, we perform an oracle experiment. We simulate a re-scorer that

correctly selects the actual class from the top-5 predictions with a fixed probability.

As shown in Table 5.7, a fine-grained classifier with 60% accuracy would raise global accuracy to 85.5%, well above the baseline. Our learned re-scorer achieves a correct switching rate of only 33.7% and often reverses correct predictions to incorrect ones. The rescoring design can improve accuracy, but the learned spatial score is not reliable enough.

Table 5.7: Top-5 re-scoring results. Simulated rows demonstrate the potential gain if spatial alignment were robust.

Method	Final Acc (%)	Correct Switches (%)
Global Baseline	79.0	-
Learned Re-scorer	79.1	33.7
Simulated Oracle (60% correct)	85.5	49.2
Simulated Oracle (80% correct)	**87.5**	**53.0**

5.5 Interpretation of the Spatial Results

The experiments identify where the proposed two-stage classifier succeeds and where it fails. Global CLIP features already retrieve plausible candidates: the correct class is often present in the top-five set even when it is not ranked first. The oracle experiment then shows that local re-scoring could raise final accuracy from 79.0% to 85.5% if the fine-grained decision were correct 60% of the time. Candidate retrieval and the re-scoring rule are therefore not the principal limitation. The measured local classifier is the principal limitation: its correct-switch rate of 33.7% is too low to offset the correct global decisions that it reverses.

The training results locate the failure more precisely. The all-parts model reaches 75.2% fine-grained training accuracy but only 28.4% test accuracy. It can fit the supplied part–description pairs, so insufficient classifier capacity is not the main problem. The learned correspondence does not transfer to unseen images. Two measurements explain this poor transfer. First, distinct part descriptions can have cosine similarity above 0.95 in CLIP's text space, leaving little margin between targets such as "spotted breast" and "striped breast". Second, the patch maps remain similar across neighbouring tokens and patches, especially on unmodified images. When both sides of a contrastive pair provide weak separation, an adapter can memorise training associations without learning a stable local decision rule.

The visual interventions support the same conclusion from a different direction. Red boxes and background blur alter the image distribution while also removing or obscuring global context, and every tested variant underperforms the unmodified

global baseline. These results show that preserving CLIP's global representation is necessary in this setting. Local evidence should supplement that representation, not replace it with a manipulated crop or mask. CLIP Surgery improves colour-token differentiation in the synthetic red-head example, but the smaller effect on the natural image and the weak classification result show that this local change is not sufficient for fine-grained transfer.

Together, the results specify useful requirements for a successor method. It needs text targets with larger margins between confusable attributes, local visual features that retain part-specific colour and pattern, and a fusion rule that defaults to the global prediction when the local score is uncertain. The oracle gains justify continuing with two-stage inference, while the train–test gap argues against adding further capacity to the same alignment objective without first improving the representations being aligned.

5.6 Chapter Summary

In this chapter, we investigate spatial–semantic alignment for fine-grained classification using CLIP and modifications to the VDT-Adapter. We tested training-free interventions (visual prompting, text refinement, CLIP Surgery) and trainable spatial adapters with part-level supervision. Neither approach outperforms strong global baselines on the CUB dataset.

The tested text descriptions have small pairwise margins in the CLIP text space, and the tested spatial-alignment objectives do not transfer beyond the training pairs. Some linguistically distinct descriptions (e.g., "spotted breast" vs "striped breast") have cosine similarity above 0.95, which reduces the separation between candidate targets in the contrastive objective.

Future investigation could evaluate explicit hierarchical reasoning. One possible direction is a visual question-answering pipeline in which sequential, targeted questions are generated (e.g., "What colour is the beak?" and "Are the wing bars present?") and each question is answered independently by a vision model. This style of reasoning decomposes the classification problem into verifiable subtasks, potentially achieving fine-grained separation without relying on a single shared embedding space that can collapse subtle distinctions. Other possible investigations include fine-tuning the text encoder on fine-grained corpora (not only for few-shot adaptation) or using attribute-specific projection heads.

Chapter 6

Predicting Vision-Language Model Performance Using LLM-Generated Counterfactuals

6.1 Introduction

Recent machine learning research has shifted from many task-specific models towards large general models, often termed Foundation Models [9]. These models, including BERT [71], DALL-E [84], CLIP [1], and the GPT family [75, 73], are trained on very large and diverse datasets, often with self-supervised objectives. They support many downstream tasks with the same set of parameters. In *Computer Vision*, this trend changes the earlier ImageNet-based pipeline [60, 85], where practitioners first pretrain a model on a large labelled dataset and then fine-tune it for each new task [49, 86, 4].

Vision-Language Foundation Models (VLFMs), such as CLIP [1] and ALIGN [79], learn a joint representation of images and text. Users map images and natural language prompts into a shared embedding space. They then measure similarity in this space to perform *zero-shot* image recognition and image–text retrieval [57, 33]. A typical prompt is "a photo of a {label}". This prompt acts as a simple classifier without extra training data [87].

The quality of zero-shot predictions of a Vision-Language Foundation Model depends on the data that support training of the model [88]. Web-scale datasets often follow a Zipf-like long-tailed distribution of concepts [58, 57]. Frequent categories receive many examples, while rare concepts appear only a few times. Work on scaling laws shows that models then exhibit log-linear performance scaling [57]. Performance on rare concepts improves slowly, even as total data size grows. As a result, many concepts remain under-represented, and their performance on fine-grained or

domain-specific tasks remains poor.

This data imbalance affects who can benefit from artificial intelligence. Many application areas lack datasets that meet their local needs, particularly in domains such as agriculture, health, and culture [89]. Foundation Models often rely on data that over-represent some regions and topics while under-representing others. When people apply these models in new or specialised settings, performance can drop, and local users may have little control over data or model design. Efforts to broaden datasets help, but still struggle to capture detailed domain knowledge [89].

A practical question follows. A practitioner may have a new dataset, for example, a fine-grained food dataset or a small medical image dataset, and a set of candidate Vision-Language Foundation Models. The practitioner would like to know which model is suitable before labelling a large test set. Running a full evaluation requires many labels and significant time. In many settings, this cost is not acceptable.

This chapter proposes a method for predicting zero-shot transfer performance of a Vision-Language Foundation Model on a new image classification dataset. The method uses one labelled example per class and measures how strongly the model separates a plausible caption from closely related counterfactual captions. A Large Language Model (LLM) generates a plausible caption and several challenging but incorrect descriptions, which we call *counterfactual prompts* (see Fig. 6.1). These similarity scores, together with standard CLIP-style prompt scores, are used as inputs to a linear regression model that predicts dataset-level zero-shot accuracy. The goal is to provide an inexpensive tool to support early decisions on model selection, particularly in under-resourced or specialised domains. Earlier chapters of this thesis focus on adapting and improving models for few-shot learning (Chapters 3 to 5). In contrast, this chapter focuses on estimating the performance of an existing Vision-Language Foundation Model before any adaptation, which complements the methods in the rest of the thesis and builds on concepts from the literature review in Chapter 2.

6.2 Methodology

Our method estimates how distinctive the representation of a given Vision-Language Foundation Model is for a specific domain, that is, a set of classes. We aim for high data efficiency, so we assume one labelled image per class. The methodology follows three stages as shown in Fig. 6.1. In this section, we describe each stage and explain how we derive the features for the regression model.

We probe the local geometry of the shared image–text embedding space of the Vision-Language Foundation Model as follows. The model maps each input image and each text string to a vector, which we call an embedding. Cosine similarity

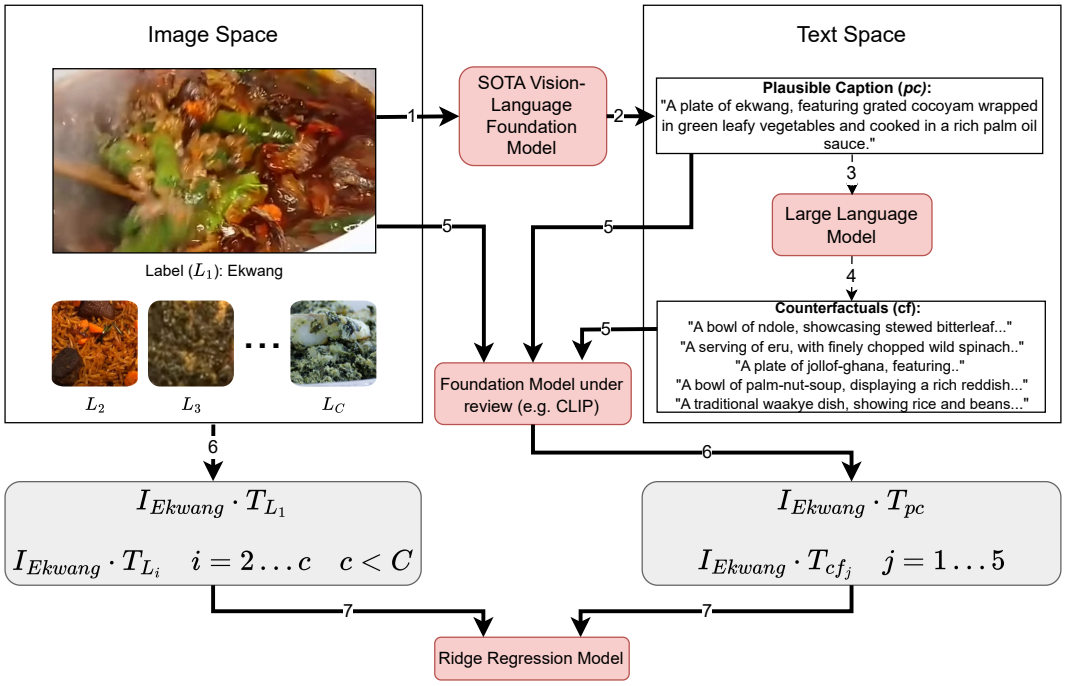


Figure 6.1: Method overview. The figure shows the three-stage pipeline that predicts zero-shot accuracy of a Vision-Language model on a target domain, using the "Ekwang" class from the African Food dataset as an example. Counterfactual probing: a single representative image (Step 1) provides a plausible caption T_{pc} (Step 2) and a set of counterfactual captions $\{T_{cf_i}\}$ from a Large Language Model (Steps 3–4). Similarity scoring: the Vision-Language Foundation Model under evaluation provides embeddings for the image ($\mathcal{I}$) and all captions (Step 5). $\{T_{L_i}\}$ represents label prompts in the vanilla CLIP [1] prompt style, for example "a photo of a {label}", with the class name inserted. These prompts are embedded by the Vision-Language Foundation Model under evaluation, OpenCLIP-ViT-B/16 [90] in the experiments. Two sets of similarity scores are as follows: one for vanilla CLIP-style label prompts and one for Large Language Model-generated captions (Step 6). Performance prediction: the similarity scores form the input to a Ridge Regression model (Step 7). This model estimates zero-shot accuracy of the Vision-Language Foundation Model on the full test set using only one labelled image per class.

between two normalised embeddings is the dot product between the corresponding vectors, which lies between -1 and 1. Higher values indicate that the model encodes the two inputs as more similar.

Image-to-Language anchoring. We randomly select one image per class from a diverse set of commonly used datasets. A multimodal Large Language Model, for example, GPT-5-Nano [72], receives the image and its ground-truth label. The output is a plausible caption T_{pc} that describes the content of the image. The caption acts as a natural-language summary of the class.

Generating counterfactuals. With T_{pc} as an anchor, we use a text-based Large Language Model to generate N counterfactual captions T_{cf_i}. The prompt instructs the model to produce descriptions that are semantically related to T_{pc} but that match other visually confusable classes in the same dataset. This step creates a small set of hard negatives for each class. In our experiments we set $N = 5$.

Vision-Language Foundation Model-based similarity scoring. Using the Vision-Language Foundation Model that we evaluate, for example, OpenCLIP [90], we compute normalised embeddings for the image I_j, the plausible caption T_{pc}, and all counterfactual captions T_{cf_i}. We then measure cosine similarity, which yields a similarity score $s_{pc_j} = I_j \cdot T_{pc}$ and a list of counterfactual scores $s_{cf_{ij}} = I_j \cdot T_{cf_i}$. These scores reflect how close the image is to each caption in the embedding space.

The method also uses standard zero-shot similarity scores from label prompts in the vanilla CLIP [1] prompt style, with prompts of the form "a photo of a {label}". These prompts yield c text embeddings T_{L_i}. In the experiments, $c = N$, so the implementation randomly selects the same number of label-based prompts as counterfactual captions. Each query image corresponds to one class and therefore yields 12 similarity scores as features for the Ridge Regression model. Method design choices appear in Section 6.4.

The final stage uses the twelve similarity-based features to estimate zero-shot accuracy of the Vision-Language Foundation Model. We train a linear regression model on a diverse collection of datasets, where ground-truth targets are measured as the zero-shot accuracy of the Vision-Language Foundation Model on full test sets. We use Ridge Regression because it constrains coefficient magnitudes via L2 regularisation to address feature correlation, as is the case in our setting. Once trained, the model is applied to a new domain using only one labelled image per class, providing a data-efficient estimate of transfer performance.

Table 6.1: Overview of Datasets Used

Dataset	Task	Classes	Focus
CIFAR-10 [91]	Object classification	10	Ten common object categories (airplane, automobile, bird, etc.) in low-resolution natural scenes.
CIFAR-100 [91]	Object classification	100	One hundred fine-grained classes grouped into 20 super-classes (e.g., maples, lobsters).
Oxford 102 Flowers [64]	Flower classification	102	Photographs of 102 flower species with diverse backgrounds, lighting, and viewpoints.
Food-101 [65]	Food classification	101	Real-world dishes (e.g., pizza, sushi, ramen) with varied preparation and noisy backgrounds.
Caltech-101 [61]	Object classification	101	Images of objects from 101 categories (e.g., faces, motorcycles, crabs) with varying scale, position, and background clutter.
Oxford-IIIT Pet [62]	Pet breed classification	37	Cat and dog breeds with diverse poses, backgrounds, and bounding box annotations.
DTD [67]	Texture classification	47	Texture attributes (e.g., striped, dotted, wrinkled) from various materials and conditions.
EuroSAT [68]	Land-use classification	10	Sentinel-2 patches of land-use categories (e.g., residential, forest, river) across Europe.
GTSRB [92]	Traffic-sign class.	43	German road signs in real driving scenarios with weather, occlusion, and angle changes.
FGVC Aircraft [43]	Aircraft-model class.	100	Commercial and military aircraft variants distinguished by subtle features. Regarded as a fine-grained image classification dataset.
ImageNet-v2 [93]	Distribution-shift test	998	Re-collected version of ImageNet-1k to measure generalisation under data distribution shifts.
MNIST [94]	Handwritten digit class.	10	Classic benchmark for handwritten digit recognition, consisting of centred, low-res digits 0–9.
STL-10 [95]	Object classification	10	A benchmark for semi-supervised learning with higher-res images and a large unlabeled set.
ImageNet-O [96]	Robustness/OoD evaluation	200	Images from 200 object categories not included in ImageNet-1k, used to test out-of-distribution handling.
African Food [97]	Food classification	6	Traditional dishes (e.g., jollof rice, injera), highlighting cultural representation in food datasets.
Beans [98]	Plant disease class.	3	Images of healthy and diseased bean leaves for agricultural disease diagnosis in smallholder farms.

6.3 Experiments and Results

Datasets. We evaluate our method on a diverse suite of datasets (see Table 6.1), covering generic and fine-grained classification, specialised domains, and datasets from under-represented domains. Specifically, we include two datasets, African Food and Beans, to test performance prediction on underrepresented domains.

Implementation details. We evaluate the OpenCLIP-ViT-B/16 model pretrained on LAION-400M [99]. We use the GPT-5-Nano Large Language Model [72] for caption and counterfactual generation. We set $N = 5$, which is the number of counterfactuals per class. Our Ridge Regression model trains on eleven datasets, which appear as blue points in Fig. 6.2, and we test it on five hold-out datasets.

We report two complementary evaluation metrics to quantify both relative ranking quality and absolute accuracy:

- **Pearson-r:** Pearson correlation coefficient that measures linear association between predicted and true accuracies in the range $[-1, 1]$; higher values indicate stronger linear agreement.
- **RMSE:** Root mean squared error between predicted and true accuracies (percentage scale here); lower values indicate better absolute agreement.

We randomly select 30% of the datasets as test sets to evaluate our Ridge Regression model. We also preselect African Food and Beans for the test set to ensure that we test our method on underrepresented domains. Table 6.2 reports the full-test-set CLIP zero-shot accuracy, our 1-shot *PreLabellingProbe* estimates, and absolute errors. The estimates closely track the ground-truth accuracies (Pearson-r = 0.96; RMSE = 10.37), with low error on African Food and moderate underestimation on Beans and CIFAR-10.

The central claim is that the one-example-per-class method provides a useful estimate of zero-shot performance of a Vision-Language Foundation Model. As shown in Fig. 6.2, the predicted and ground-truth zero-shot accuracies show a strong linear correlation across datasets and lie close to the identity line.

Table 6.2: Full test set CLIP zero-shot accuracy vs. PreLabellingProbe (Percentage Scale). Pearson-r = 0.96. Root mean squared error = 10.37.

Dataset	CLIP ZS Accuracy (Full testset)	PreLabellingProbe (1-shot)	Error
African Food	38.24	41.22	2.98
Beans	39.84	26.12	-13.72
Caltech101	89.25	84.86	-4.39
CIFAR10	91.68	76.41	-15.27
Food101	83.76	74.38	-9.38

The held-out datasets include African Food [97] and Beans [98]. Across the held-out datasets, predicted and measured accuracies have Pearson $r = 0.96$. These estimates can inform whether full annotation is warranted and how fine-grained those annotations should be.

6.4 Ablation and Discussion

The ablation quantifies the contribution of each component in *PreLabellingProbe*. It compares three settings: (i) only Large Language Model-generated captions and counterfactuals, (ii) only vanilla CLIP-style label-prompt scores, and (iii) the full combined variant. This isolates the source of gains and tests whether the signals are complementary.

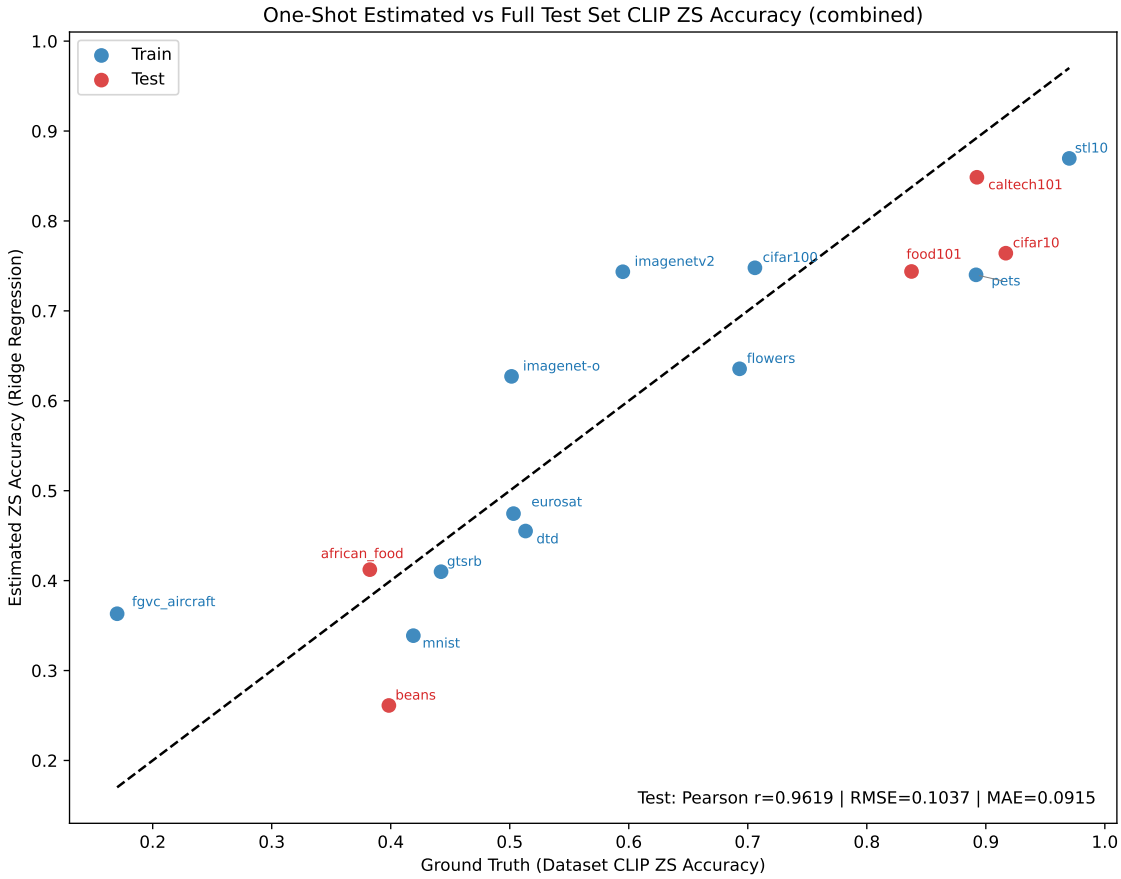


Figure 6.2: Ground-truth versus predicted zero-shot accuracy. The scatter plot depicts the method's predictions across 16 datasets. The x-axis shows the actual zero-shot accuracy of OpenCLIP-ViT-B/16, calculated on each dataset's full test set, and the y-axis shows predicted accuracy using only a single labelled image per class. The dashed black line indicates perfect agreement. Blue points are datasets used to train the Ridge Regression model; red points are unseen test datasets. The strong test correlation demonstrates accuracy and generalisation, including on specialised datasets such as *African Food* and *Beans*.

The central hypothesis is that the combination of Large Language Model-generated counterfactuals and label-prompt scores provides a more effective probe of the Vision-Language Foundation Model than either signal alone.

Vanilla CLIP-style prompt scores refer to the prompt template introduced with CLIP [1], not to a separate CLIP implementation. All ablation scores use the same evaluated Vision-Language Foundation Model, OpenCLIP-ViT-B/16 [90] in the experiments. The label-prompt scores use "a photo of a {class_name}" and "a photo of a {other_class_name}", where "{other_class_name}" is randomly sampled from the other classes in the dataset. The rest of the pipeline remains unchanged to ensure a fair comparison.

Table 6.3: Test performance comparison of variants

Variant	Pearson-r	RMSE
LLM-generated only	0.849	0.145
Vanilla CLIP-style label prompts only	0.947	0.150
PreLabellingProbe	**0.962**	**0.104**

The results in Table 6.3 show that the combined *PreLabellingProbe* variant achieves the strongest agreement with ground-truth accuracies, with the highest Pearson-r and the lowest RMSE. This indicates a complementary signal between Large Language Model-derived features and vanilla CLIP-style label-prompt features.

The cost of evaluating a new dataset using *PreLabellingProbe* is low. For the African Food dataset with six classes, Large Language Model caption and counterfactual generation takes about one minute and twenty-three seconds in total, or roughly fourteen seconds per class, with an API cost of about $0.006. CLIP similarity score computation and Ridge Regression inference take less than five seconds on a central processing unit such as the AMD Ryzen 9 7950X, which makes the pipeline practical on standard hardware.

6.5 Operational Use of the Estimate

Pearson correlation and RMSE describe different uses of the predictor. The full method's Pearson correlation of 0.96 shows that it ranks easier and harder dataset–model combinations consistently. Its RMSE of 0.10 means that an individual prediction can still differ from measured accuracy by a practically important amount. PreLabellingProbe should therefore be used for screening and resource allocation, not as a replacement for the final labelled evaluation required to report model performance.

The ablation explains why both inputs are retained. Label-prompt scores alone achieve a high correlation of 0.95 but an RMSE of 0.15. They provide a stable measure of how strongly the model matches an image to its correct class label, but do not measure how clearly that class is separated from plausible alternatives. The LLM-only variant compares generated captions with counterfactual class descriptions and has a lower correlation of 0.85 when used by itself. Combining the two signals reduces RMSE to 0.10. The label-prompt score measures positive class fit, while the counterfactual comparisons measure relative class confusion; together they describe both whether the target class matches and whether competing classes match almost as well.

In practice, the estimate can precede annotation in a staged workflow. One representative labelled image per class is used to run the probe. Classes with weak positive scores or small margins over counterfactuals can then receive additional annotation or targeted inspection, while clearly separated classes require less immediate attention. After that intervention, a conventional labelled test set remains necessary for the final accuracy estimate. This use matches the method's evidence: it identifies where a model is likely to struggle early enough to guide data collection, while keeping the distinction between predicted and measured performance explicit.

6.6 Chapter Summary

We introduce a data-efficient method to estimate dataset-level zero-shot performance of Vision-Language Foundation Models. Using a single labelled example per class, we generate counterfactual captions with a Large Language Model to probe how well latent representations within the Vision-Language Foundation Model separate a class from related alternatives. Experiments show accurate performance estimation across diverse datasets, including specialised and under-represented domains. The method estimates performance before full annotation or model adaptation.

Chapter 7

Conclusion

This thesis describes methods for adapting and evaluating Vision-Language Models (VLFMs), with a focus on Contrastive Language-Image Pre-training (CLIP), when labelled data are scarce and validation resources are limited. The work starts from the observation that CLIP provides a strong zero-shot baseline. Nevertheless, CLIP adaptation to new domains faces three main obstacles: current methods require large, labelled datasets to tune hyperparameters, text prompts are too simplistic, and support for fine-grained spatial reasoning is limited. In addition, practitioners often lack reliable methods for estimating model performance when they cannot afford large annotated test sets.

To address these problems, the thesis introduces a set of methods that intervene at different points in the adaptation and evaluation pipeline. First, HOSO-Adapter provides a validation-free adapter that respects strict few-shot constraints. Second, semantic knowledge from Large Language Models is incorporated at the level of global class descriptions through CLIP-GPT and VDT-Adapter, and at the level of local part descriptions through spatial alignment experiments. Third, LLM-generated counterfactuals are used to predict CLIP's performance from a few labelled samples.

The following sections summarise the main findings for the four research questions posed in Chapter 1, discuss broader implications and limitations, and outline future directions for this line of work.

7.1 Summary of Contributions and Findings

7.1.1 RQ1: Validation-Free Adaptation

How can one adapt CLIP in a strict few-shot setting without a separate validation set?

Chapter 3 examines common practice in few-shot adaptation and shows that

many methods rely on validation sets or implicit test-set tuning. These practices violate the strict few-shot protocol and do not reflect realistic deployment, where only a small support set is available.

Key findings:

- The **Hold-One-Shot-Out (HOSO)** strategy reserves one example per class from the support set and uses this 1-shot cache to tune the blending ratio between CLIP and the adapter.
- Optimising the blending ratio on the 1-shot cache, while training the adapter on the remaining support examples, acts as a **dynamic regulariser**. When performance on the hold-out examples deteriorates, the contribution of the CLIP prior increases and adapter overfitting is reduced.
- **HOSO-Adapter** reaches state-of-the-art accuracy in the validation-free setting and improves on the CLIP-Adapter baseline by about four percentage points on average across 11 datasets.
- In 8-shot and 16-shot settings, HOSO-Adapter improves on an "Oracle" CLIP-Adapter that tunes the blending ratio on the test set with a grid search. At those shot counts, the learned ratio gives higher average accuracy than the tested static ratios.

7.1.2 RQ2: LLM-Based Global Enhancement

How can Large Language Models generate side information for CLIP that improves zero-shot and few-shot transfer at the level of global class descriptions?

Chapter 4 addresses the limitation of using short class names as prompts. It examines whether richer natural-language descriptions from LLMs yield text embeddings that align more closely with visual structure.

Key findings:

- Analysis of text embeddings shows that prompts based on LLM-generated visual descriptions form semantic graphs that match visual similarity structure better than standard dictionary definitions or WordNet hierarchies.
- In the zero-shot setting, the **CLIP-GPT** ensemble aggregates multiple LLM-generated descriptions for each class. This ensemble improves accuracy over default CLIP prompts by about two percent on average, with gains above five percent on fine-grained datasets such as CUB, DTD and EuroSAT.
- In the few-shot setting, the **Visually Descriptive Text Adapter (VDT-Adapter)** uses a self-attention mechanism to select relevant descriptions and

combine them into a single text representation. VDT-Adapter improves **Base-to-New** generalisation and exceeds CoCoOp by roughly three percent on unseen classes.

7.1.3 RQ3: Fine-Grained Spatial Alignment

Can explicit alignment between spatial visual features and LLM-generated fine-grained part descriptions improve few-shot generalisation on image classification tasks?

Chapter 5 explores whether the embedding space of CLIP supports reliable alignment between local image patches and detailed attribute descriptions, such as different beak shapes or plumage patterns. The chapter evaluates both training-free interventions and trainable spatial adapters on the CUB dataset.

Key findings:

- Training-free spatial interventions, such as adding red bounding boxes or blurring parts of the image, do not improve performance and often degrade accuracy. Visual prompting reduces zero-shot accuracy from 55.25% to approximately 50%, while background blurring drops accuracy to 36.67%. Together, these results show that the tested modifications disrupt CLIP's pre-trained representation rather than improve spatial localisation.
- A trainable spatial adapter can match patches to text descriptions with 75.2% accuracy on the training set, but its accuracy drops to 28.4% on the test set. The train–test gap is consistent with overfitting to the training correspondences.
- Some distinct fine-grained descriptions, such as "spotted breast" and "striped breast", have cosine similarities above 0.95 in the CLIP text space. The resulting small margins between candidate targets limit the supervision available for discriminative local features.
- Adding a spatial alignment loss with moderate weight ($\lambda = 0.4$) to the global objective yields small but consistent improvements, raising base-class accuracy from 77.8% to 79.4% and new-class accuracy from 61.6% to 62.6%.
- Under the tested CUB setup, the standard CLIP embedding space does not provide enough part-level separation for the evaluated spatial-transfer methods.

7.1.4 RQ4: Performance Prediction

Can CLIP's zero-shot accuracy on a full test set be estimated from one labelled image per class and LLM-generated counterfactuals?

Chapter 6 shifts focus from adaptation to evaluation. It considers the case where a practitioner needs to select or configure a model for a new dataset but cannot afford to label a large test set.

Key findings:

- The **PreLabellingProbe** method uses an LLM to generate a caption and a set of hard negative counterfactual captions for a single labelled image per class.
- The margin between the similarity score that CLIP assigns to the true caption and to the counterfactual captions is a strong predictor of dataset-level performance.
- A simple linear regressor trained on these margins estimates the full test-set zero-shot accuracy with a **Pearson correlation of 0.96** across a diverse set of datasets, using one labelled example per class.

7.2 Discussion and Speculation

The discussion in this section is more speculative and extends beyond the strict empirical evidence in the previous chapters. It reflects the intuition developed throughout this thesis and outlines possible directions for future research.

7.2.1 Role of the Pre-Trained CLIP Prior

The performance of HOSO-Adapter in Chapter 3 suggests that a single hold-out example per class can provide a useful signal for tuning the blending ratio. One explanation is that the zero-shot CLIP model provides a stable reference, while the adaptation step calibrates deviations from that model. The experiments do not test whether a 1-shot cache would be sufficient when learning a model from scratch.

The cache measures whether increasing reliance on the adapter improves held-out classification relative to retaining more of the pre-trained representation. The results suggest that one example per class can provide a useful signal for this scalar parameter, although they do not establish that the same cache size is sufficient for other hyperparameters.

7.2.2 Limited Fine-Grained Separation in the Text Encoder

The Chapter 5 results identify limited separation in the CLIP text encoder as one factor in the weak spatial-transfer performance. Large language models can produce

distinct descriptions of fine-grained attributes, but the CLIP text encoder maps some of these descriptions to vectors with small angular margins.

The tested embeddings place terms such as "spotted" and "striped" close together, leaving small angular margins between some candidate descriptions. Future work should test whether text encoders trained to separate fine-grained attributes provide a stronger signal for dense spatial alignment.

7.2.3 The Role of Counterfactuals in Robustness

The performance prediction study in Chapter 6 suggests that robustness is associated with a wide margin between correct and incorrect captions. Counterfactual captions provide a practical way to probe this margin. When a caption describing a bowl of Jollof rice and a counterfactual caption describing a bowl of fried rice receive similar CLIP scores, the model does not clearly separate the two concepts in its embedding space.

The same counterfactual margins could also guide **active learning** by identifying samples for which the model assigns similar scores to plausible and incorrect descriptions. An acquisition function could request annotations for these samples rather than sampling images at random.

7.2.4 Limitations

The methods and findings in this thesis have several limitations that inform the scope of the conclusions.

First, HOSO-Adapter in Chapter 3 provides minimal gains on datasets where CLIP already aligns well with the target domain, such as ImageNet, where the optimal blending ratio favours the zero-shot prior and adaptation offers little additional benefit. The method also requires at least two labelled examples per class, since one must be reserved for the cache, which limits applicability in the extreme 1-shot regime.

Second, the LLM-generated side information in Chapter 4 relies on the factual accuracy and visual relevance of language model outputs. The experiments show that LLM-generated descriptions do not always match ground-truth attribute annotations, and that hallucinated or generic descriptions can introduce noise. The methods also assume access to a capable LLM, which may not be feasible in all deployment settings.

Third, the spatial alignment experiments in Chapter 5 focus on a single fine-grained dataset (CUB) and use ground-truth part annotations that are not available in most practical settings. The negative results clarify the limitations of the tested methods but do not offer a working solution for fine-grained spatial transfer.

Fourth, the performance prediction method in Chapter 6 shows moderate underestimation errors on some datasets, with RMSE of 10.37 percentage points and individual errors exceeding 15 percentage points on CIFAR-10 and Beans. The method also assumes that a single representative image per class captures class-level difficulty, which may not hold for datasets with high intra-class variability.

Finally, all methods in this thesis build on the CLIP architecture with frozen encoders. The findings may not transfer directly to newer vision–language models with different pre-training objectives or architectures, such as models that use generative rather than contrastive objectives.

7.2.5 Future Outlook

The results across Chapters 3 to 6 show that strong image and text representations do not by themselves guarantee effective cross-modal alignment. The remaining challenge is to align the two modalities while preserving both coarse- and fine-grained information.

Several concrete research directions emerge from this work:

Hierarchical alignment. Instead of a single global similarity score, future models could first align objects, then align parts and attributes, and finally align textures or contexts. This behaviour would mirror the polytomous keys that biologists use to identify species. The failure modes observed in Chapter 5 highlight how difficult this goal remains for current embeddings and provide concrete diagnostics and benchmarks for future work.

Generalising the HOSO principle. The HOSO strategy of using held-out examples as a micro-validation set could extend beyond the blending ratio to other critical hyperparameters. Learning rates, regularisation strengths, and early stopping criteria could all benefit from data-dependent tuning on a 1-shot cache. The same principle might apply to prompt selection in methods like CoOp, where the cache could guide which learned prompts to retain. Additionally, any adapter method that interpolates between a frozen prior and learned parameters could adopt a similar decoupled optimisation scheme.

Active learning with counterfactuals. The performance prediction method in Chapter 6 identifies images and classes where the model exhibits small margins between actual and counterfactual descriptions. These margins could guide active learning strategies that focus annotation effort on the most ambiguous regions of the embedding space.

7.3 Concluding Remarks

This thesis shows that Vision-Language Foundation Models provide a strong starting point for visual recognition, but that careful adaptation and evaluation strategies are still required when labelled data are scarce.

Chapter 3 introduces HOSO-Adapter, a validation-free adapter that operates within strict few-shot constraints. In the reported experiments, the pre-trained CLIP model provides a stable reference from which a single hold-out example per class can tune the blending ratio. This decoupled optimisation strategy acts as a dynamic regulariser and could be tested for other hyperparameters in future work.

Chapter 4 demonstrates that language-based side information from LLMs improves both zero-shot and few-shot transfer. The CLIP-GPT ensemble and VDT-Adapter show that richer text prompts yield text embeddings that align more closely with visual structure, with gains of up to five percentage points on fine-grained datasets. These methods provide a scalable alternative to manual attribute annotation.

Chapter 5 investigates whether explicit spatial alignment between image patches and part-level text descriptions can improve fine-grained recognition. The experiments reveal that trainable spatial adapters can fit training correspondences but struggle to generalise, and that training-free visual prompting interventions degrade performance. These results motivate further investigation into the conditions under which spatial supervision benefits vision-language adaptation.

Chapter 6 presents PreLabellingProbe, a data-efficient method that estimates zero-shot performance using only one labelled image per class. The strong correlation between predicted and measured accuracy (Pearson $r = 0.96$) indicates that practitioners can assess a model's suitability before committing to full-scale annotation.

Together, these methods support low-data adaptation and evaluation.

Bibliography

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

[2] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. "Conditional Prompt Learning for Vision-Language Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. New Orleans, LA, USA: IEEE, June 2022, pp. 16795–16804. ISBN: 978-1-66546-946-3. DOI: 10.1109/CVPR52688.2022.01631. (Visited on 12/11/2023).

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems* 25 (2012).

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, June 2016, pp. 770–778. ISBN: 978-1-4673-8851-1. DOI: 10.1109/CVPR.2016.90. (Visited on 10/16/2023).

[5] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. "A Simple Framework for Contrastive Learning of Visual Representations". In: *Proceedings of the 37th International Conference on Machine Learning*. Vol. 119. Proceedings of Machine Learning Research. PMLR, 2020, pp. 1597–1607. URL: https://proceedings.mlr.press/v119/chen20j.html.

[6] Kilian Q Weinberger and Lawrence K Saul. "Distance metric learning for large margin nearest neighbor classification." In: *Journal of machine learning research* 10.2 (2009).

[7] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. "Supervised Contrastive Learning". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 18661–18673. (Visited on 11/21/2023).

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=YicbFdNTTy.

[9] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, et al. *On the Opportunities and Risks of Foundation Models*. July 2022. arXiv: 2108.07258 [cs]. URL: https://crfm.stanford.edu/report.html (visited on 12/04/2023).

[10] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. *Representation Learning with Contrastive Predictive Coding*. Jan. 2019. DOI: 10.48550/arXiv.1807.03748. arXiv: 1807.03748 [cs, stat]. URL: https://arxiv.org/abs/1807.03748 (visited on 01/31/2024).

[11] Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Bin Xiao, Ce Liu, Lu Yuan, and Jianfeng Gao. "Unified Contrastive Learning in Image-Text-Label Space". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. New Orleans, LA, USA: IEEE, June 2022, pp. 19141–19151. ISBN: 978-1-66546-946-3. DOI: 10.1109/CVPR52688.2022.01857. (Visited on 12/13/2023).

[12] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. "Efficient Estimation of Word Representations in Vector Space". In: *International Conference on Learning Representations, Workshop Track*. 2013. DOI: 10.48550/arXiv.1301.3781. arXiv: 1301.3781 [cs.CL]. URL: https://arxiv.org/abs/1301.3781.

[13] Jeffrey Pennington, Richard Socher, and Christopher Manning. "Glove: Global Vectors for Word Representation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, 2014, pp. 1532–1543. DOI: 10.3115/v1/D14-1162. (Visited on 12/02/2025).

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. "Attention Is All You Need". In: *Advances in Neural Information Processing Systems*. Vol. 30. 2017. arXiv: 1706.03762 [cs.CL]. URL: https://papers.nips.cc/paper/7181-attention-is-all-you-need.

[15] Nils Reimers and Iryna Gurevych. “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*. 2019, pp. 3982–3992. DOI: 10.18653/v1/D19-1410.

[16] Yuxin Jiang, Linhan Zhang, and Wei Wang. “Improved Universal Sentence Embeddings with Prompt-based Contrastive Learning and Energy-based Learning”. In: *Findings of the Association for Computational Linguistics: EMNLP 2022*. 2022, pp. 3021–3035. DOI: 10.18653/v1/2022.findings-emnlp.220.

[17] Catherine Chen, Kevin Lin, and Dan Klein. “Constructing Taxonomies from Pretrained Language Models”. In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Online: Association for Computational Linguistics, 2021, pp. 4687–4700. DOI: 10.18653/v1/2021.naacl-main.373. (Visited on 12/02/2025).

[18] Aleksandar Shtedritski, Christian Rupprecht, and Andrea Vedaldi. “What Does CLIP Know about a Red Circle? Visual Prompt Engineering for VLMs”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 11987–11997. DOI: 10.1109/ICCV51070.2023.01101.

[19] Chong Zhou, Chen Change Loy, and Bo Dai. “Extract Free Dense Labels from CLIP”. In: *Computer Vision – ECCV 2022*. Lecture Notes in Computer Science. Springer, 2022, pp. 696–712. DOI: 10.1007/978-3-031-19815-1_40.

[20] Yongming Rao, Wenliang Zhao, Guangyi Chen, Yansong Tang, Zheng Zhu, Guan Huang, Jie Zhou, and Jiwen Lu. “DenseCLIP: Language-Guided Dense Prediction with Context-Aware Prompting”. In: *CVPR*. 2022.

[21] Yi Li, Hualiang Wang, Yiqun Duan, Ji-Heng Zhang, and Xiaomeng Li. “A Closer Look at the Explainability of Contrastive Language-Image Pre-training”. In: *Pattern Recognition* 162 (2025), p. 111409. DOI: 10.1016/j.patcog.2025.111409. URL: https://linkinghub.elsevier.com/retrieve/pii/S003132032500069X.

[22] Sule Bai, Yong Liu, Yifei Han, Haoji Zhang, Yansong Tang, Jie Zhou, and Jiwen Lu. “Self-Calibrated CLIP for Training-Free Open-Vocabulary Segmentation”. In: *IEEE Transactions on Image Processing* 34 (2025), pp. 8271–8284. DOI: 10.1109/TIP.2025.3639996. URL: https://doi.org/10.1109/TIP.2025.3639996.

[23] Luming Tang, Davis Wertheimer, and Bharath Hariharan. "Revisiting Pose-Normalization for Fine-Grained Few-Shot Recognition". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020, pp. 14352–14361. URL: https://openaccess.thecvf.com/content_CVPR_2020/html/Tang_Revisiting_Pose-Normalization_for_Fine-Grained_Few-Shot_Recognition_CVPR_2020_paper.html.

[24] Gregor Geigle, Radu Timofte, and Goran Glavaš. "African or European Swallow? Benchmarking Large Vision-Language Models for Fine-Grained Object Classification". In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 2653–2669. DOI: 10.18653/v1/2024.emnlp-main.154. URL: https://aclanthology.org/2024.emnlp-main.154/.

[25] Beichen Zhang, Pan Zhang, Xiaoyi Dong, Yuhang Zang, and Jiaqi Wang. "Long-CLIP: Unlocking the Long-Text Capability of CLIP". In: *Computer Vision – ECCV 2024*. Ed. by Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol. Cham: Springer Nature Switzerland, 2025, pp. 310–325. ISBN: 978-3-031-72983-6.

[26] Yanghao Li, Haoqi Fan, Ronghang Hu, Christoph Feichtenhofer, and Kaiming He. "Scaling Language-Image Pre-training via Masking". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 23390–23400.

[27] Liangliang Cao, Bowen Zhang, Chen Chen, Yinfei Yang, Xianzhi Du, Wencong Zhang, Zhiyun Lu, and Yantao Zheng. *Less Is More: Removing Text-regions Improves CLIP Training Efficiency and Robustness*. May 2023. DOI: 10.48550/arXiv.2305.05095. arXiv: 2305.05095 [cs]. URL: https://arxiv.org/abs/2305.05095 (visited on 12/05/2023).

[28] Sheng Shen, Chunyuan Li, Xiaowei Hu, Yujia Xie, Jianwei Yang, Pengchuan Zhang, Zhe Gan, Lijuan Wang, Lu Yuan, et al. "K-lite: Learning transferable visual models with external knowledge". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 15558–15573.

[29] George A Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J Miller. "Introduction to WordNet: An on-line lexical database". In: *International journal of lexicography* 3.4 (1990), pp. 235–244.

[30] Lijie Fan, Dilip Krishnan, Phillip Isola, Dina Katabi, and Yonglong Tian. "Improving CLIP Training with Language Rewrites". In: *Advances in Neural Information Processing Systems*. Vol. 36. 2023.

[31] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. "Learning to Prompt for Vision-Language Models". In: *International Journal of Computer Vision* 130.9 (Sept. 2022), pp. 2337–2348. ISSN: 0920-5691, 1573-1405. DOI: 10.1007/s11263-022-01653-1. arXiv: 2109.01134 [cs]. URL: https://doi.org/10.1007/s11263-022-01653-1 (visited on 12/07/2023).

[32] Sarah Pratt, Rosanne Liu, Ali Farhadi, and Ian Covert. "What Does a Platypus Look Like? Generating Customized Prompts for Zero-Shot Image Classification". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 15691–15701.

[33] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. "CLIP-Adapter: Better Vision-Language Models with Feature Adapters". In: *International Journal of Computer Vision* 132.2 (2024), pp. 581–595. DOI: 10.1007/s11263-023-01891-x.

[34] 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: *Computer Vision – ECCV 2022*. Lecture Notes in Computer Science. Springer, 2022, pp. 493–510. DOI: 10.1007/978-3-031-19833-5_29.

[35] Julio Silva-Rodríguez, Sina Hajimiri, Ismail Ben Ayed, and Jose Dolz. "A Closer Look at the Few-Shot Adaptation of Large Vision-Language Models". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, USA: IEEE, June 2024, pp. 23681–23690. ISBN: 979-8-3503-5300-6. DOI: 10.1109/CVPR52733.2024.02235. (Visited on 10/20/2025).

[36] Naoki Kato, Yoshiki Nota, and Yoshimitsu Aoki. "Proto-Adapter: Efficient Training-Free CLIP-Adapter for Few-Shot Image Classification". In: *Sensors* 24.11 (Jan. 2024), p. 3624. ISSN: 1424-8220. DOI: 10.3390/s24113624. (Visited on 10/16/2025).

[37] Zexun Xie, Min Xu, Shudong Zhang, and Lijuan Zhou. "RCAT: Retentive CLIP Adapter Tuning for Improved Video Recognition". In: *Electronics* 13.5 (Jan. 2024), p. 965. ISSN: 2079-9292. DOI: 10.3390/electronics13050965. (Visited on 10/16/2025).

[38] Omiros Pantazis, Gabriel Brostow, Kate Jones, and Oisin Mac Aodha. "SVL-Adapter: Self-Supervised Adapter for Vision-Language Pretrained Models". In: *Proceedings of the British Machine Vision Conference*. 2022.

[39] Zhengfeng Lai, Joohi Chauhan, Brittany N. Dugger, and Chen-Nee Chuah. "Bridging the Pathology Domain Gap: Efficiently Adapting CLIP for Pathology Image Analysis with Limited Labeled Data". In: *Computer Vision – ECCV 2024*. Ed. by Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol. Cham: Springer Nature Switzerland, 2025, pp. 256–273. ISBN: 978-3-031-73039-9. DOI: 10.1007/978-3-031-73039-9_15.

[40] Ekin D. Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. "RandAugment: Practical Automated Data Augmentation with a Reduced Search Space". In: *Advances in Neural Information Processing Systems*. Vol. 33. 2020, pp. 18613–18624.

[41] David Berthelot, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. "ReMixMatch: Semi-Supervised Learning with Distribution Matching and Augmentation Anchoring". In: *International Conference on Learning Representations*. 2020. URL: https://openreview.net/forum?id=HklkeR4KPB.

[42] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. *The Caltech-UCSD Birds-200-2011 Dataset*. Tech. rep. CNS-TR-2011-001. California Institute of Technology, 2011.

[43] Subhransu Maji, Esa Raichur, Raquel Urtasun, Trevor Darrell, and Andrea Vedaldi. "Fine-grained visual classification of aircraft". In: *Proceedings of the British Machine Vision Conference*. BMVA Press. 2013.

[44] Edgar Schonfeld, Sayna Ebrahimi, Samarth Sinha, Trevor Darrell, and Zeynep Akata. "Generalized Zero- and Few-Shot Learning via Aligned Variational Autoencoders". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 8247–8255.

[45] Muhammad Ferjad Naeem, Yongqin Xian, Luc Van Gool, and Federico Tombari. "I2DFormer: Learning Image to Document Attention for Zero-Shot Image Classification". In: *Advances in Neural Information Processing Systems*. Vol. 35. 2022.

[46] Muhammad Ferjad Naeem, Muhammad Gul Zain Ali Khan, Yongqin Xian, Muhammad Zeshan Afzal, Didier Stricker, Luc Van Gool, and Federico Tombari. "I2MVFormer: Large Language Model Generated Multi-View Document Supervision for Zero-Shot Image Classification". In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Vancouver, BC, Canada: IEEE, June 2023, pp. 15169–15179. ISBN: 9798350301298. DOI: 10.1109/CVPR52729.2023.01456. (Visited on 05/07/2024).

[47] Wenjia Xu, Yongqin Xian, Jiuniu Wang, Bernt Schiele, and Zeynep Akata. "Attribute prototype network for zero-shot learning". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 21969–21980.

[48] Han Zhang, Tao Xu, Mohamed Elhoseiny, Xiaolei Huang, Shaoting Zhang, Ahmed Elgammal, and Dimitris Metaxas. "SPDA-CNN: Unifying Semantic Part Detection and Abstraction for Fine-Grained Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, June 2016, pp. 1143–1152. ISBN: 978-1-4673-8851-1. DOI: 10.1109/CVPR.2016.129. (Visited on 05/09/2024).

[49] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. "Microsoft COCO: Common Objects in Context". In: *Proc. Eur. Conf. Comput. Vis. (ECCV)*. 2014, pp. 740–755.

[50] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. "Few-Shot Learning via Embedding Adaptation With Set-to-Set Functions". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, USA: IEEE, June 2020, pp. 8805–8814. ISBN: 978-1-7281-7168-5. DOI: 10.1109/CVPR42600.2020.00883. (Visited on 12/02/2025).

[51] Jinhao Li, Haopeng Li, Sarah Monazam Erfani, Lei Feng, James Bailey, and Feng Liu. "Visual-Text Cross Alignment: Refining the Similarity Score in Vision-Language Models". In: *Proceedings of the 41st International Conference on Machine Learning*. Vol. 235. Proceedings of Machine Learning Research. PMLR, 2024, pp. 28018–28039. URL: https://proceedings.mlr.press/v235/li24ag.html.

[52] Lingfeng Yang, Yueze Wang, Xiang Li, Xinlong Wang, and Jian Yang. "Fine-Grained Visual Prompting". In: *Advances in Neural Information Processing Systems*. Vol. 36. 2023.

[53] Choubo Ding and Guansong Pang. "Zero-Shot Out-of-Distribution Detection with Outlier Label Exposure". In: *International Joint Conference on Neural Networks*. 2024, pp. 1–8. DOI: 10.1109/IJCNN60899.2024.10650173.

[54] Sepideh Esmaeilpour, Bing Liu, Eric Robertson, and Lei Shu. "Zero-Shot Out-of-Distribution Detection Based on the Pre-trained Model CLIP". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 36.6 (June 2022), pp. 6568–6576. ISSN: 2374-3468, 2159-5399. DOI: 10.1609/aaai.v36i6.20610. arXiv: 2109.02748 [cs]. (Visited on 07/31/2025).

[55] Swayambhu Saxena, Necati Cihan Camgoz, Jean-Yves Franceschi, Niloofar Mireshghallah, Chris J. Pal, Marcin Drozdzal, and Andreea Romero-Soriano. "Predicting the Performance of Foundation Models via Agreement-on-the-Line". In: *Adv. Neural Inf. Process. Syst. (NeurIPS)*. Vol. 36. 2023, pp. 1–17.

[56] Ravi Shekhar, Anirudh Venkatesh, Tilman Baumgärtner, Elia Bruni, Barbara Plank, Raffaella Bernardi, and Raquel Fernandez. "FOIL It! A New Dataset for Fine-Grained Object-Oriented Image-Language Mismatch Detection". In: *Proc. Conf. Empir. Methods Nat. Lang. Process. (EMNLP)*. 2017, pp. 261–270.

[57] Vishaal Udandarao, Ameya Prabhu, Adhiraj Ghosh, Yash Sharma, Philip H. S. Torr, Adel Bibi, and Matthias Bethge. "No "Zero-Shot" Without Exponential Data: Pretraining Concept Frequency Determines Multimodal Model Performance". In: *Advances in Neural Information Processing Systems*. Vol. 37. 2024.

[58] George K. Zipf. *The Psycho-Biology of Language: An Introduction to Dynamic Philology*. Cambridge, MA: MIT Press, 1965.

[59] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, et al. "Overcoming Catastrophic Forgetting in Neural Networks". In: *Proceedings of the National Academy of Sciences* 114.13 (Mar. 2017), pp. 3521–3526. DOI: 10.1073/pnas.1611835114. (Visited on 11/14/2025).

[60] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "Imagenet: A Large-Scale Hierarchical Image Database". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2009, pp. 248–255.

[61] Li Fei-Fei, R. Fergus, and P. Perona. "Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Worskshops (CVPRW)*. 2004, pp. 178–178.

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

[63] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. "3D Object Representations for Fine-Grained Categorization". In: *2013 IEEE International Conference on Computer Vision Workshops*. 2013, pp. 554–561. DOI: 10.1109/ICCVW.2013.77.

[64] 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*. IEEE. 2008, pp. 722–729.

[65] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. "Food-101–mining 101 food categories through web scraping". In: *European conference on computer vision*. Springer. 2014, pp. 445–461.

[66] Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. "SUN Database: Large-scale Scene Recognition from Abbey to Zoo". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2010, pp. 3485–3492.

[67] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. "Describing Textures in the Wild". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014, pp. 3606–3613.

[68] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. "Introducing EuroSAT: A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification". In: *IEEE International Geoscience and Remote Sensing Symposium*. 2018, pp. 204–207. DOI: 10.1109/IGARSS.2018.8519248.

[69] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. *UCF101: A Dataset of 101 Human Actions Classes From Videos in the Wild*. Tech. rep. CRCV-TR-12-01. Center for Research in Computer Vision, University of Central Florida, 2012. arXiv: 1212.0402 [cs.CV]. URL: https://www.crcv.ucf.edu/data/UCF101.php.

[70] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. "Mpnet: Masked and permuted pre-training for language understanding". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 16857–16867.

[71] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Ed. by Jill Burstein, Christy Doran, and Thamar Solorio. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4171–4186. DOI: 10.18653/v1/N19-1423. URL: https://aclanthology.org/N19-1423/.

[72] OpenAI. *Models*. Accessed 2025-10-27. URL: https://platform.openai.com/docs/models.

[73] OpenAI. *GPT-4 Technical Report*. 2023. arXiv: 2303.08774 [cs.CL].

[74] Alex Fang, Gabriel Ilharco, Mitchell Wortsman, Yuhao Wan, Vaishaal Shankar, Achal Dave, and Ludwig Schmidt. "Data determines distributional robustness in contrastive language image pre-training (clip)". In: *International Conference on Machine Learning*. PMLR. 2022, pp. 6216–6234.

[75] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, et al. "Language Models are Few-Shot Learners". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[76] Grant Z. Higginbotham and Nathan S. Matthews. *Prompting and In-Context Learning: Optimizing Prompts for Mistral Large*. May 2024. DOI: 10.21203/rs.3.rs-4430993/v1. (Visited on 05/29/2024).

[77] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Delong Chen, et al. "Survey of Hallucination in Natural Language Generation". In: *ACM Computing Surveys* 55.12 (Dec. 2023), pp. 1–38. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3571730. arXiv: 2202.03629 [cs]. (Visited on 05/29/2024).

[78] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. "Large language models are zero-shot reasoners". In: vol. 35. 2022, pp. 22199–22213.

[79] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, et al. "Scaling up visual and vision-language representation learning with noisy text supervision". In: *International conference on machine learning*. PMLR. 2021, pp. 4904–4916.

[80] Sachit Menon and Carl Vondrick. "Visual Classification via Description from Large Language Models". In: *ICLR* (2023).

[81] Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi-Rui Tam, Keith Stevens, Abdullah Barhoum, Nguyen Minh Duc, Oliver Stanley, et al. "OpenAssistant conversations - democratizing large language model alignment". In: *Proceedings of the 37th International Conference on Neural Information Processing Systems*. NIPS '23. New Orleans, LA, USA: Curran Associates Inc., 2023.

[82] Jifan Yu, Xiaozhi Wang, Shangqing Tu, Shulin Cao, Daniel Zhang-Li, Xin Lv, Hao Peng, Zijun Yao, Xiaohan Zhang, et al. "Kola: Carefully benchmarking world knowledge of large language models". In: *International Conference on Learning Representations*. Vol. 2024. 2024, pp. 44594–44637.

[83] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. "Chain-of-thought prompting elicits reasoning in large language models". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 24824–24837.

[84] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. "Zero-Shot Text-to-Image Generation". In: *Proceedings of the 38th International Conference on Machine Learning*. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, pp. 8821–8831. URL: https://proceedings.mlr.press/v139/ramesh21a.html.

[85] Olga Russakovsky et al. "ImageNet Large Scale Visual Recognition Challenge". In: *Int. J. Comput. Vis.* 115.3 (2015), pp. 211–252.

[86] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C. Lawrence Zitnick. "Microsoft COCO Captions: Data Collection and Evaluation Server". In: *arXiv preprint arXiv:1504.00325* (2015). DOI: 10.48550/arXiv.1504.00325. arXiv: 1504.00325 [cs.CV]. URL: https://arxiv.org/abs/1504.00325.

[87] Muhammad Awais, Muzammal Naseer, Salman Khan, Rao Muhammad Anwer, Hisham Cholakkal, Mubarak Shah, Ming-Hsuan Yang, and Fahad Shahbaz Khan. "Foundation Models Defining a New Era in Vision: A Survey and Outlook". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 47.4 (Apr. 2025), pp. 2245–2264. DOI: 10.1109/TPAMI.2024.3506283. arXiv: 2307.13721 [cs.CV]. URL: https://doi.org/10.1109/TPAMI.2024.3506283.

[88] Ioana Bica, Anastasija Ilić, Matthias Bauer, Goker Erdogan, Matko Bošnjak, Christos Kaplanis, Alexey A. Gritsenko, Matthias Minderer, Charles Blundell, et al. "Improving Fine-Grained Understanding in Image-Text Pre-Training". In: *Proceedings of the 41st International Conference on Machine Learning*. Vol. 235. Proceedings of Machine Learning Research. PMLR, July 2024, pp. 3974–3995. arXiv: 2401.09865 [cs.CV]. URL: https://proceedings.mlr.press/v235/bica24a.html (visited on 05/09/2024).

[89] Abdul-Hakeem Omotayo, Ashery Mbilinyi, Lukman Ismaila, Houcemeddine Turki, Mahmoud Abdien, Karim Gamal, Idriss Tondji, Yvan Pimi, Naome A. Etori, et al. "The State of Computer Vision Research in Africa". In: *Journal*

of Artificial Intelligence Research 81 (Sept. 2024), pp. 43–69. ISSN: 1076-9757. DOI: 10.1613/jair.1.16653. arXiv: 2401.11617 [cs]. URL: https://www.jair.org/index.php/jair/article/view/16653 (visited on 07/29/2025).

[90] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. "Reproducible Scaling Laws for Contrastive Language-Image Learning". In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 2818–2829. DOI: 10.1109/CVPR52729.2023.00276. arXiv: 2212.07143 [cs]. URL: https://openaccess.thecvf.com/content/CVPR2023/html/Cherti_Reproducible_Scaling_Laws_for_Contrastive_Language-Image_Learning_CVPR_2023_paper.html (visited on 08/01/2025).

[91] Alex Krizhevsky. *Learning multiple layers of features from tiny images*. Tech. rep. University of Toronto, 2009.

[92] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. "The German traffic sign recognition benchmark: a man vs. machine". In: *The 2012 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2012, pp. 1–8.

[93] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. "Do ImageNet classifiers generalize to ImageNet?" In: *International conference on machine learning*. PMLR. 2019, pp. 5389–5400.

[94] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[95] Adam Coates, Honglak Lee, and Andrew Y Ng. "An analysis of single-layer networks in unsupervised feature learning". In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2011, pp. 215–223.

[96] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. "Natural Adversarial Examples". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 15262–15271.

[97] Momodou Omeiza, John See, Wui-Seong Liew, and Hong Tat Ewe. "The African Food Dataset (AFD)". In: *Pattern Recognition. ICPR International Workshops and Challenges: Virtual Event, January 10–15, 2021, Proceedings, Part VII*. Springer. 2022, pp. 201–216.

[98] Makerere University AI Lab. *Beans Dataset*. https://github.com/AI-Lab-Makerere/beans-dataset. A dataset of bean leaf images for disease classification. 2021.

[99] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Jenia Jitsev, and Aran Komatsuzaki. "LAION-400M: Open Dataset of CLIP-Filtered 400 Million Image-Text Pairs". In: *NeurIPS 2021 Workshop on Data-Centric AI*. 2021.