The Australian National University
2600 ACT | Canberra | Australia

Australian National University

School of Computing

College of Engineering, Computing and Cybernetics (CECC)

Visual Prompting in In-context Learning

— 24 pt research project (S2/S1 2023–2024)

A report submitted for the course
COMP8800, Research Project

By:
Ke Yan

Supervisor:
Dr. Jing Zhang

June 2024

Declaration:

I declare that this work:

- upholds the principles of academic integrity, as defined in the University Academic Misconduct Rules;
- is original, except where collaboration (for example group work) has been authorised in writing by the course convener in the class summary and/or Wattle site;
- is produced for the purposes of this assessment task and has not been submitted for assessment in any other context, except where authorised in writing by the course convener;
- gives appropriate acknowledgement of the ideas, scholarship and intellectual property of others insofar as these have been used;
- in no part involves copying, cheating, collusion, fabrication, plagiarism or recycling.

June, Ke Yan

Abstract

This study introduces a novel approach to adapting pre-trained models to new tasks using visual prompting, a technique that modifies the input space instead of the underlying model parameters. This method enables the flexible application of a single model across various tasks without extensive retraining, traditionally required in transfer learning. By extending the concept of in-context learning from natural language processing to computer vision, this approach employs a prompt retriever framework to select optimal in-context pairs based on cosine similarity, effectively bridging the gap between textual and visual domains.

Additionally, the report explores advancements such as prompt enhancers and ensembling techniques, which refine inputs to improve model performance and robustness. Experimental evaluations on standard benchmarks demonstrate significant improvements over existing methods. Notably, the proposed model, equipped with learnable prompts, enhances mean Intersection over Union (mIoU) scores by 7.6% for foreground segmentation and 16.26% for single object detection compared to baseline models. These findings highlight the efficacy of visual prompting in enhancing the adaptability and efficiency of pre-trained models for a wide range of computer vision tasks, setting a new standard for model versatility and robustness in the field.

Table of Contents

Chapter 1

Introduction

Imagine teaching a skilled painter to explore new artistic styles from just a few examples. In the world of artificial intelligence, a similar challenge exists as we teach computer models to understand and perform tasks outside their initial programming. My work explores visual prompting, a technique akin to giving a computer vision model a new set of brushes designed to master unseen styles quickly and efficiently.

Machine learning models, a subset of AI, are specifically trained with data to excel at particular tasks. However, adapting these pre-trained models to new tasks is a fundamental challenge, traditionally solved through transfer learning. This approach involves using a model developed for one task as the starting point for another, applying its learned knowledge to new, related problems. Despite its effectiveness, transfer learning often requires extensive re-training with new data, which can be both computationally expensive and time-consuming.

An emerging solution to this challenge is the use of prompting. Prompting involves adapting pre-trained models to new tasks by modifying their input space with task-specific information. For instance, the sentences you type into ChatGPT are prompts that guide the AI to respond appropriately. Compared with traditional transfer learning, prompting has several advantages. First, it uses less training data, making it quicker and more efficient. Secondly, it allows the AI model to take on new tasks without changing its underlying structure, which means one single model can perform various downstream tasks without requiring a redesign each time.

In the area of Natural Language Processing (NLP), this concept is extended into what researchers describe as in-context learning. In large language models like GPT-3 Brown et al. (2020), in-context learning enables models to perform new tasks by using specific examples provided along with a query. These examples, known as in-context examples, serve as prompts that guide the model's responses. This method demonstrates how the principles of prompting are effectively applied to handle complex language tasks. The

(a) Colorization (b) Segmentation (c) Binary segmentation

Figure 1.1: (a) Colorization: Demonstrates the model's ability to add color to grayscale images. (b) Segmentation: Shows how the model can distinguish and outline different objects within a scene. (c) Binary Segmentation: Illustrates the model's capability to separate an object from its background using binary classification. Each example utilizes a grid-like visual prompt, with the model's prediction displayed in the bottom-right cell, showcasing the adaptability of the in-context learning model to varied visual tasks.

diverse capabilities of in-context learning models are demonstrated through the output results for different tasks as shown in Figure 1.1.

Although well-established in NLP, in-context learning is relatively new in the field of computer vision, particularly with the advent of large-scale language vision model like ViT Dosovitskiy et al. (2021) and CLIP Radford et al. (2021). While these model can achieve impressive results, they require millions of training data, making it impractical to transfer them to downstream tasks with traditional methods. As a result, visual in-context learning is important because its ability to efficiently transfer the model to different tasks by simply altering the input prompt.

Similar to textual prompting in NLP, visual prompting involves altering visual inputs by adding or modifying elements in an image to improve performance or reprogram the model for different tasks. The challenge of visual in-context learning lies in how to make model understand the prompt and to make prediction based on it. Recent studies such as Wang et al. (2023) Bar et al. (2022), have developed frameworks for visual in-context learning, as illustrated in Figure 1.2. These frameworks select a in-context prompt pair from training dataset, combine them with a query image to form a grid-like visual prompt.

1.1 Motivations

Visual prompting represents a significant departure from conventional model training techniques, offering a versatile and resource-efficient method for model adaptation. This

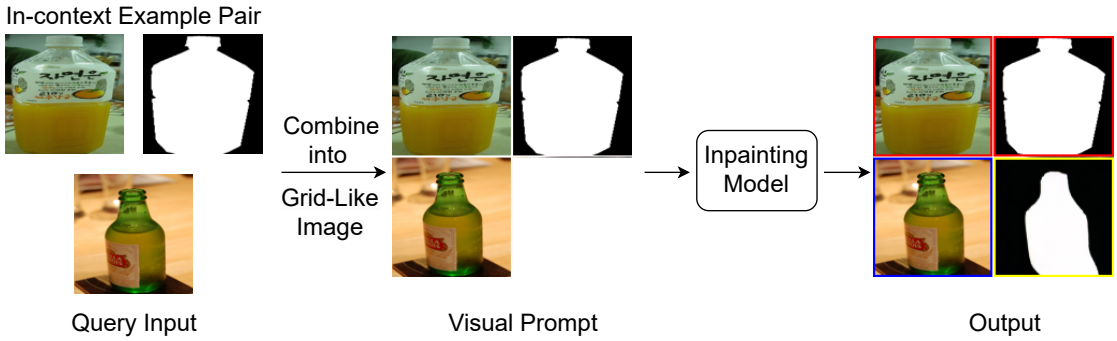

Figure 1.2: An overview of visual in-context learning: This method combines a query image and an in-context pair to create a four-cell grid canvas. An empty cell in the bottom-right corner of the grid is reserved for predictions. The canvas serves as the visual prompt and is fed into an inpainting model to generate the prediction, highlighted by a yellow border around the resulting cell.

approach can address critical bottlenecks in deploying AI solutions across diverse downstream tasks.

The traditional approach of transfer learning, while effective, is often computationally expensive and time-consuming, requiring extensive re-training with new data. Visual prompting, however, can adapt pre-trained models to new tasks with minimal additional data, making it a more efficient and scalable solution. This efficiency is particularly important in scenarios where data collection is difficult or costly.

Research indicates that prompts semantically similar to the query image tend to yield better performance Sun et al. (2023); Zhang et al. (2023b). Designing an effective visual prompt manually is a challenging task due to the complexity in image space. To address this, I employ a prompt retriever framework, as described in Zhang et al. (2023b); Wang et al. (2023). The core of the framework is a score function that assesses each in-context pair in the database, calculating the similarity between the example and the query image based on cosine distance. After the scoring process, the framework selects the in-context pairs with the highest scores to construct a visual prompt. The cosine distance measures how closely the features of the query and the in-context pair align, with features extracted by the image encoder of large scale models like CLIP Radford et al. (2021) or ViT Dosovitskiy et al. (2021).

However, the selected examples might not be optimal due to the finite size of database. This limitation has led to the development of a prompt enhancer, inspired by works such as Bahng et al. (2022); Chen et al. (2023); Oh et al. (2023). This enhancer overlays a learnable prompt onto the input image, refining the initially retrieved in-context pair. Similar to the visual in-context learning method, the enhanced in-context pair and query input are image are combined to create a visual prompt, which is then processed by an inpainting model as shown in Figure 1.2. These visual prompts act as hints, aiding the

model in learning more effectively and enhancing its performance.

To further enhance the model robustness and performance, prompt ensembling are used, inspired by Lester et al. (2021). This approach allows for multiple constructions of the visual prompt for the same in-context pair and query image. The resulting outputs are then averaged based on voting techniques to ensure the most effective prompt is selected. Additionally, the ensembling of multiple in-context examples can consistently improve performance, as demonstrated in subsequent figures.

By investigating and refining visual prompting techniques, this research aims to contribute to the development of more adaptable and efficient visual in-context models.

1.2 Contributions

This work makes the following contributions:

- Demonstrates how visual prompts can be effectively used to adapt models for tasks such as single object segmentation and detection.
- Develops and evaluates a prompt retriever framework that employs cosine similarity to select the most effective in-context pairs, optimizing the adaptation process.
- Explores and evaluates techniques such as prompt enhancer and ensembling to refine the visual prompts, ensuring models perform more optimally.
- Conducting extensive experiments and analyses to understand the influence of various parameters and the effectiveness of different models in the context of visual prompting. This rigorous analysis helps find the optimal configurations and strategies.

Chapter 2

Background

This chapter aims to provide the necessary background for understanding visual in-context learning and the role of prompting strategies in machine learning models, specifically in the visual domain. This includes explaining foundational concepts in machine learning, deep learning, and the evolution of models used in computer vision.

2.1 Machine Learning (ML)

Machine learning (ML) is a branch of artificial intelligence that focuses on building systems that learn from data to make decisions or predictions. These systems improve their performance on a given task over time with more data.

2.1.1 Deep Learning

Deep learning is a subset of machine learning involving neural networks with many layers. These models automatically learn to extract and represent features from raw data, enabling high performance in tasks such as image recognition.

2.1.2 Neural Network

Neural networks are computational models inspired by the human brain. They consist of interconnected layers of nodes (neurons) where each connection has a weight adjusted during training. These networks can model complex patterns in data.

2.1.3 Convolutional Neural Networks(CNNs)

CNNs are specialized neural networks designed for processing structured grid-like data such as images. They use convolutional layers to automatically and adaptively learn spatial hierarchies of features from visual data. CNNs have been highly successful in tasks

like image classification and object detection due to their ability to capture local patterns and textures in images. However, CNNs have limitations in capturing long-range dependencies within an image, leading researchers to explore alternative architectures.

2.2 Visual Machine Learning

2.2.1 Computer Vision Overview

Computer vision is a field of machine learning that enables computers to interpret and make decisions based on visual data such as images and videos. This field employs algorithms and models to recognize patterns or identify objects.

2.2.2 Transformer-based Methods in Computer Vision

The advent of Transformer models, initially for natural language processing tasks, has adapted in the visual domain, leading to the creation of Vision Transformers (ViT) Dosovitskiy et al. (2021).

Vision Transformers (ViT) Dosovitskiy et al. (2021)

ViTs represent images as sequences of patches and apply self-attention mechanisms to capture global dependencies among these patches. This method contrasts with CNNs by focusing on relationships over larger distances within the image. ViTs have demonstrated competitive performance compared to state-of-the-art CNN models and introduced new perspectives on how to process visual information.

Self-Attention Mechanism Self-attention mechanism in Transformer Vaswani et al. (2023) enables the model to focus on different parts of the image and understand the relationships between them, regardless of their spatial distance. This enables a more comprehensive representation of the visual content.

Advantages Over CNNs Transformers' ability to capture long-range dependencies gives them an advantage over CNNs, which primarily focus on local patterns. This broader view helps in understanding complex structures within images.

2.2.3 Tasks in Computer Vision

Object Detection

Object detection involves identifying and locating objects within an image or video, classifying them, and determining their precise boundaries, typically represented by bounding boxes.

Semantic Segmentation

Semantic segmentation classifies each pixel in an image into a category representing what is being depicted. Unlike object detection, it provides a pixel-wise breakdown of the image but does not differentiate between instances of the same object.

Single Object Segmentation

Single object segmentation, or instance segmentation, extends semantic segmentation by distinguishing between different instances of the same object. This task is crucial in scenarios where multiple occurrences of similar objects need to be individually recognized and analyzed.

2.3 Transfer Learning

Transfer learning is a technique where a model trained for one task is repurposed for another. More specifically, this method leveraging the knowledge from trained models including features, weights, and layers, making training model for new tasks significantly easier than training from scratch.

Transfer learning is particularly valuable when collecting large amounts of training data is costly or impractical. The effectiveness of this technique has popularized the use of pre-trained (PTR) models. These models are trained on diverse datasets, enabling them to be easily transferred to various downstream tasks.

2.4 In-Context Learning (ICL)

In-context learning effectively uses the model from transfer learning and PTR models with a rich pre-knowledge base. ICL is a method where a model uses input-output examples along with a query to make predictions. This approach enables the model to apply learned patterns from the examples directly to the query without retraining. The rich feature representations learned during pre-training enable the model to perform well with just a few examples which is known as few-shot learning. This approach and its variants, including zero-shot and one-shot learning, seek to generalize the model's ability to perform unseen tasks with a few or zero in-context examples, as demonstrated in GPT-3 Brown et al. (2020).

Few-shot Learning In few-shot learning, the model is provided with limited examples or shots for the task it needs to perform. These examples are used to guide the model in predicting on unseen data. More shots typically leads to higher prediction accuracy as the model has access to a wider range of task-specific knowledge.

Zero-Shot Learning Zero-Shot learning is an extreme case of Few-shot learning where the model is required to perform tasks it is not trained on. This approach relies heavily

on the model's ability to generalize from the existing knowledge base and conceptual descriptions of the new task it has been given.

One-shot Learning One-shot learning is a subset of few-shot learning where the model is given only a single example from which to learn a task. One-shot learning is crucial in fields such as facial recognition, where it is necessary to identify a person from a single image.

2.5 Prompt Engineering

Although transfer learning is very reliable in most situation, transfer a large scale PTR model with millions of parameters can be time consuming and requires task-specific knowledge. Prompting methods have emerged to alleviate the computational cost by modifying the input space of large-scale PTR models. One of the key advantages of prompting is that a single PTR model can be applied to many downstream tasks without altering the model itself while only the input space is modified. This approach theoretically enables the creation of a universal model capable of performing a wide range of downstream tasks efficiently.

2.5.1 Natural Language Prompting

In Natural Language Processing (NLP), prompting involves designing text inputs for pre-trained language models, such as GPT-3 Brown et al. (2020) or BERT Devlin et al. (2019). This technique allows these models to perform various downstream tasks without updating the model parameters. For instance, given a pre-trained model, the task might be to fill in missing words. Consider the example "I am so [MASK]." The effectiveness of this task depends on how well the underlined prompt is designed. However, manually creating these prompts can be time-consuming and requires extensive task-specific knowledge.

Prompt tuning Lester et al. (2021) introduces "soft prompts" prepended to the input layers. Those prompts are parameterized vectors learned through back-propagation while keeping the PTR model parameters fixed. Prefix tuning Li and Liang (2021) on the other hand, the learnable prompts are introduced at each encoder layer. For large scale models, fine tuning the whole model trains significantly more parameters compared with prompt learning

2.5.2 Visual Prompting

Visual prompting involves modifying the image input of pre-trained models (PTR), such as Vision Transformers (ViT) Dosovitskiy et al. (2021) or CLIP Radford et al. (2021). Techniques like learnable visual prompts Bahng et al. (2022) and image inpainting Bar et al. (2022) help models adapt to new tasks efficiently. Visual Prompting Bahng et al. (2022) (VP) constructs learnable visual prompts by merging borders with the original

image, effectively guiding the model to focus on specific parts of the image and improving performance on various tasks.

2.5.3 Visual In-context Learning

Visual in-context learning extends the concept of in-context learning to the visual domain by providing visual prompts alongside the input query. One of the earliest attempts at visual in-context learning is Flamingo Alayrac et al. (2022), which accepts text as input and can process both images and videos. Bar et al. (2022) is the first to generalize in-context learning to the visual domain via image inpainting. This approach constructs a grid-like image concatenated with input-output examples and an input query, transforming the visual prompting problem into an image inpainting or hole-filling problem. Since the Masked Autoencoder (MAE) He et al. (2021) is trained to reconstruct masked patches, it serves as a base model for VP to perform different downstream tasks without fine-tuning.

Similarly, Painter Wang et al. (2023) concatenates input-output pairs as input to indicate which task to perform. These prompts (input-output pairs) are either selected from the training set or generated.

Building on these foundations, visual in-context learning frameworks enable models to efficiently adapt to new tasks by leveraging example-based visual prompts.

2.6 Terminology

Loss Function A loss function is a metric used to determine the difference between the model's current predictions and the actual ground truth data. A common type of loss function used is the cross-entropy loss.

Cross Entropy Loss Cross-entropy loss is particularly effective when predicting the probability distribution across multiple classes. It is defined as:

$$L = -\sum_{i=1}^{N}\sum_{c=1}^{M} y_{i,c}\log(p_{i,c})$$

L is the overall loss for the entire dataset; N is the number of samples in the dataset; M is the number of classes; $y_{i,c}$ is a binary variable indicating whether the class label c is align with the observation i; $\log(p_{i,c}$ is logarithm of the predicted probability that observation i belongs to class c.

Since the loss increases as the predicted probability diverges from the actual label. It effectively penalizes deviations of the predicted probabilities from the actual class labels, driving the model's predictions towards the true probabilities.

Learning Rate The learning rate is a hyperparameter during training machine learning models, particularly when using gradient descent algorithms. It determines the size of the steps that the training algorithm takes in order to reach the minimum of the loss function.

Adam Optimizer Kingma and Ba (2017) Adam, Adaptive Moment Estimation, adjusts the learning rate for each parameter individually based on estimates of first and second moments of the gradients.

$$
\begin{aligned}
m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t, \\
v_t &= \beta_2 v_{t-1} + (1-\beta_2) g_t^2, \\
\hat{m}_t &= \frac{m_t}{1-\beta_1^t}, \\
\hat{v}_t &= \frac{v_t}{1-\beta_2^t}, \\
\theta_{t+1} &= \theta_t - \frac{\eta \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}.
\end{aligned}
$$

Where:

- m_t and v_t are estimates of the first moment (the mean) and the second moment (the uncentered variance) of the gradients.
- $\hat{m}_t$ and $\hat{v}_t$ are bias-corrected versions of m_t and v_t.
- β_1 and β_2 are the exponential decay rates for these moment estimates, typically close to 1.
- g_t is the gradient at time step t.
- θ_t is the parameter vector at time step t.
- η is the step size or learning rate.
- ϵ is a small scalar (e.g., 10^{-8}) added to improve numerical stability.

Adam is broadly used across different types of neural networks and has become a default choice for many applications due to its robustness in handling various optimization challenges, especially when the gradient can become sparse (as in NLP with large vocabularies).

Scheduler When training neural network, a scheduler is a method used to adjust learning rate over time. They help in preventing overfitting, ensuring faster convergence, and enabling the model to escape local minima. By dynamically adjusting the learning rate, schedulers contribute significantly to the effectiveness and efficiency of model training.

Cosine Annealing with Warm Restarts Scheduler Loshchilov and Hutter (2017) The Cosine Annealing with Warm Restarts scheduler is a learning rate scheduler used in training neural networks that adjusts the learning rate according to a cosine function between specified intervals, effectively "resetting" the learning rate periodically. This method is particularly useful for finding global minima in the optimization landscape and can lead to better overall performance by avoiding local minima.

It adjust the learning rate based on the following fomular:

$$\eta_t = \eta_{\min} + 0.5 \times (\eta_{\max} - \eta_{\min}) \times (1 + \cos(\frac{T_{\text{cur}}}{T_{\max}}\pi))$$

Where:

- $\eta_{\min}$ is the minimum learning rate.
- $\eta_{\max}$ is the maximum learning rate.
- T_{cur} is the current epoch count since restart.
- $T_{\max}$ is the total number of epoch between restarts.

The learning rate cycles between $\eta_{\min}$ and $\eta_{\max}$, following a cosine curve, which is reset after every epochs $T_{\max}$.

Numpy NumPy, short for Numerical Python, is a fundamental package for scientific computing in Python. NumPy is widely used in both academia and industry for various types of data analysis, scientific computing, and machine learning tasks due to its efficiency compared with standard python.

Chapter 3

Related Work

3.1 Visual In-context Learning

One of the earliest work in visual in-context learning is Flamingo Alayrac et al. (2022), which processes images and videos based on text inputs. This early model sets a precedent for subsequent developments in combining Large Language Models (LLMs) with vision models. Building on this, studies by Zhou et al. (2024) and Chen et al. (2023) further explore the integration of visual data and contextual learning.

Visual in-context learning has been significantly advanced by recent contributions from Zhu et al. (2023), Liu et al. (2023), and Puri (2023), which have developed frameworks that blend LLMs capabilities with image recognition capabilities, enhancing the model's ability to interpret and analyzing visual content alongside textual data. BLIP-2 Li et al. (2023), similar to Flamingo, focuses on vision-language tasks but introduces a critical enhancement: the Querying Transformer (Q-Former). This trainable module is designed to refine the output features from an image encoder. The Q-Former selectively feeds useful visual information into a frozen LLM while filtering out irrelevant details, optimizing the model's efficiency and effectiveness in handling complex vision-language tasks. Comparing the performance of BLIP-2 Li et al. (2023) and Flamingo Alayrac et al. (2022), both models integrate LLMs with vision models. However, BLIP-2 introduces a Querying Transformer that refines output features more effectively, optimizing efficiency for vision-language tasks. In contrast, Flamingo lacks such a mechanism to filter irrelevant visual information, resulting in less efficient processing.

In visual prompt learning, Bar et al. (2022) is the first to generalize in-context learning to visual domain via image implanting. This technique involves constructing a grid-like image that integrates input-output examples with a query image, effectively transforming visual prompting into a problem of image implanting or hole filling. With MAE He et al. (2021), which is adept at reconstructing masked image patches, Visual Prompting (VP)

uses a MAE-VQGAN model equipped with a VQGAN Esser et al. (2021) codebook to tackle various downstream tasks without the need for fine-tuning. This strategy allows the system to adapt to new tasks by simply modifying input prompts rather than undergoing extensive retraining.

Building on this concept, Painter Wang et al. (2023) uses the Vision Transformer Dosovitskiy et al. (2021) (ViT) as its base model, concatenating input-output pairs to dictate the specific tasks the model should perform. These prompts, either drawn from the training set or synthetically generated, guide the model in processing images. A derivative model, SegGPT Wang et al. (2023), further refines this approach for segmentation tasks by training the model on images where colors are arbitrarily assigned to different segments. This method trains the model to depend less on color and more on the contextual and structural elements of the images, enhancing its ability to generalize across diverse visual scenarios.

Visual Prompting (VP) Bahng et al. (2022) incorporates learnable prompts into the model's input. These prompts, designed as borders around the original image, are used for classification tasks in conjunction with CLIP. The integration of discrete text prompts and a hard-coded mapping strategy enables VP to effectively address classification challenges. VP demonstrates enhanced robustness against distribution shifts when used with CLIP, outperforming conventional vision models. Despite its successes, a significant limitation of VP lies in its rigid approach to optimizing text prompts for CLIP, which may not fully exploit the potential connection between text and visual data. This limitation highlights the necessity for more dynamic models between textual and visual elements, as explored in Section 3.4 for advanced prompt tuning techniques.

To enhance the effectiveness of visual prompts, Wu et al. (2023) integrates adversarial reprogramming techniques, refining prompts through data augmentation and size reduction of the original images to maintain the integrity of the original information, which is blurred in Bahng et al. (2022). Meanwhile, Huang et al. (2023) introduces the use of distinct prompts with specific subsets of data, accommodating the inherent diversity of large datasets. This move from generic to context-specific prompts represents a significant shift towards more semantically rich visual learning frameworks with more semantic relevant prompts, achieves better performance than Bahng et al. (2022).

In the studies conducted by Bar et al. (2022) and Wang et al. (2023), the selection of in-context examples within the visual grids relies on random selection, which may not always lead to the most effective prompts. This somewhat random approach can potentially limit the model's capacity to generalize across varying visual scenarios.

To refine this process, InMeMo Zhang et al. (2023a) leverages the framework established by Bahng et al. (2022) and the image implanting techniques from Bar et al. (2022) and Wang et al. (2023) to augment input-output pairs with two learnable prompts. This approach underscores the importance of prompt quality in visual in-context learning, utilizing the CLIP image encoder as a feature extractor to align in-context examples with query images closely. This method enhances relevance and precision in finding

the optimal prompt. However, while this approach marks a significant improvement, the reliance on a single feature extractor like CLIP may introduce biases based on the training data of the encoder itself.

SEEM Zou et al. (2023) extends these concepts into interactive and compositional inputs for segmentation tasks, integrating memory tokens to retain historical mask information alongside user corrections. This method uses an interactive learning environment that dynamically handles a wide range of segmentation tasks with minimal supervision.

Similarly, SAM Kirillov et al. (2023) is a foundational model for segmentation that responds to diverse prompt types including geometric forms and textual descriptions, including points, boxes, masks, or text. Supported by the fully automated annotated SA-1B dataset, SAM's training involves simulating various prompts to fine-tune its responses with the ground truth. Its capability for zero-shot transfer, similar to CLIP's flexibility within the DALL.E system, underscores its robustness and adaptability.

Both SEEM and SAM accept a variety of prompt types as input for both standardized and user-defined segmentation tasks. However, SEEM uses referred regions from external images as reference segmentation points that is different from traditional segmentation models.

Zhang et al. (2023b) is the first to explore the impact of visual in-context examples on model performance. Their work introduces two approach of an innovative unsupervised prompt retrieval method and a supervised neural network. This network leverages a learnable feature extractor, optimized through contrastive learning similar to CLIP, to enhance the selection of in-context examples that significantly improve performance, evidenced by a notable 1 to 3 mIoU increase even amidst distribution shifts. Crucially, their findings underscore the importance of semantic similarity between the query and in-context examples, aligning with established principles in Natural Language Processing (NLP) Liu et al. (2021).

Further advancing on visual in-context learning, Prompt-SeLF Sun et al. (2023) explores prompt selection and fusion. Their method employs an off-the-shelf model to calculate the spatial representation at the pixel level, subsequently predicting outcomes from eight different fusion images composed of various arrangements of input cells and the blank prediction cell similar to Bar et al. (2022). The fusion process, employing a 4/8 voting strategy, enhances performance, suggesting that integrating multiple visual prompts through ensemble methods can significantly refine prediction accuracy. Interestingly, while the ensemble method shows superior results, the study reveals that the standalone performance of their method closely matches that of Zhang et al. (2023b)'s supervised neural network without prompt ensembling.

The research also highlights a critical limitation when increasing the number of in-context examples. Specifically, placing seven examples within a limited 224×224 pixel space drastically reduces resolution and degrades performance, indicating a trade-off between the quantity of contextual information and image quality. To solve this, they

make multiple predictions, each from a different configuration of four grids relative to one example. These predictions are then analyzed through a prompt fusion ensemble strategy, aiming to optimize the final output.

3.2 Annotation prompts

CPT Yao et al. (2022) introduces a prompting method specifically designed for tasks related to visual grounding or Referring Expression Comprehension by transforming these tasks into a fill-in-the-blank problem. Each segment of an image is assigned a unique color, integrating both visual and textual prompts. This method not only simplifies the task of linking descriptive text to specific image parts but also significantly enhances the model's precision in visual recognition tasks. However, this approach may impose limitations on the model's flexibility by binding it to predefined color assignments.

Meanwhile, Shtedritski et al. (2023) addresses a critical shortfall in CLIP Radford et al. (2021), which typically processes the entirety of an image when matching it to a text description. This often results in suboptimal performance for tasks requiring detailed recognition of specific image areas, such as Referring Expression Comprehension (REC). Instead of finding the image crop that maximising the score with textual description Subramanian et al. (2022) this approach marks the relevant areas directly on the image, which enhance its ability to perform tasks that demand localized understanding. This strategy markedly improves keypoint identification and localization, surpassing traditional methods that rely on extracting and evaluating entire image crops based on textual alignment.

However, the annotation-driven approach could potentially lead to overfitting on specific marker types, possibly reducing the model's generalization ability across different visual scenarios or marker types.

3.3 Unified Prompting

Unified Prompt Tuning (UPT) Zang et al. (2022) addresses the limitations observed when separate prompt tuning methods are applied to text and vision components. This approach is particularly crucial in scenarios characterized by high intra-class visual variance or low inter-class text embedding variance. UPT utilizes a lightweight transformer network that generates both text and visual prompts, which are then strategically injected into various layers of dual encoders. This method leads to more robust and accurate model performance across varied tasks.

Meanwhile, Multi-modal Prompt Learning (MaPLe) Khattak et al. (2023) enhances the interaction between modalities through a coupling function that maps text prompts into vision prompts. This process involves a vision-to-language projection that converts text prompts into visual prompts that can be processed alongside image data. These prompts, now learnable tokens, are then integrated into deeper layers of the transformer.

MaPLe has found that limiting the depth of prompt injection optimizes performance by preventing unnecessary adjustments to all parameters. This approach not only boosts the model's efficiency but also significantly enhances its ability to generalize across different domains and adapt to new, unseen environments.

Both UPT and MaPLe represent critical steps forward in the development of more integrated and adaptable multi-modal systems.

3.4 Prompt tuning

In the field of NLP, initial experiments in prompt tuning, such as those by Lester et al. (2021) and Li and Liang (2021), introduced the concepts of prompt tuning and prefix tuning.

Visual Prompt Tuning VPT Jia et al. (2022) extends the use of additional tokens the visual domain, alleviating the general fine-tuning cost. . VPT deploys prompts at strategic layers of the neural network: either just beyond the embedding layer (VPT-shallow) or across all layers (VPT-Deep). The deeper integration generally yields better performance, supporting the argument that prompts learn more effectively within latent spaces than directly at the input level. Empirical results demonstrate VPT's superiority over traditional fine-tuning methods.

Context Optimization (CoOp) Zhou et al. (2022b) automates the generation of text prompts instead of manual designing prompts, which could be a time-consuming process and demanding task-specific knowledge. While CoOp simplifies prompt creation and potentially enhances model robustness with fewer context tokens, it notably struggles with generalizing to unseen classes which is a drawback highlighting the need for improvements.

Building on CoOp, the instance-specific prompting method CoCoOp Zhou et al. (2022a) introduces input-conditional tokens that adjust based on individual image instances, promoting better generalizability across diverse data points. CoCoOp demands extensive computational resources due to its per-image instance processing approach, and its generalization performance still lags behind CLIP in several datasets, suggesting a crucial area for further optimization and resource efficiency.

To mitigate the overfitting limitation, Language-Aware Soft Prompting (LASP) Bulat and Tzimiropoulos (2023) introduces egularization techniques to align learnable prompts more closely with pre-designed, effective CLIP-like prompts. By minimizing the distance between the generated and the hand-crafted prompts along with a grouped representation optimized with subsets of text prompts LASP surpasses CLIP in a majority of classes. Additionally, LASP's performance is further boosted in virtual class settings, where class names not present in the base classes are included during training, along with layer normalization.

While CoOp Zhou et al. (2022b) aligns a image with one label, addressing single-label

classification tasks, **DualCoOp** Sun et al. (2022) extends to solve the multi-label recognition problem by learning two contrastive prompts for each class. This dual-prompt strategy enhances the model's ability to differentiate between multiple relevant labels for a single image, addressing the multi-label recognition challenge. Moreover, DualCoOp reformulates the last pooling layer of CLIP, and applies class-specific pooling to adaptively aggregate region features in the multi-label setting. However, DualCoOp faces challenges in zero-shot multi-label recognition, revealing a gap in its ability to predict unseen labels accurately.

3.5 Meta Learning

Meta learning, often described as "learning to learn", focuses on enhancing a model's ability to quickly adapt to new tasks by utilizing previously acquired knowledge.

Diversity-Aware Meta Visual Prompting DAM-VP Huang et al. (2023) addresses the limitation of using a single, generic prompt for datasets with large diversity. DAM-VP splits the dataset into smaller, manageable subsets, each associated with a unique prompt. These prompts are not arbitrary; they are initialized with a meta-prompt, leveraging accumulated knowledge from prior tasks to hasten the convergence of new prompts on subsequent datasets. During inference, the selection of prompts is strategically based on the feature distance between the input and the subsets, ensuring that the most semantically relevant prompts are utilized for each instance. While DAM-VP customizes prompts based on dataset diversity, the main idea is still find the optimal prompt for different inputs. In contrast, Zhou et al. (2022a) adopts a different strategy by extracting and leveraging detailed image-specific information to dynamically generate the most effective text prompt for each instance.

Chapter 4

Method

In this chapter, I present advanced methods to enhance the accuracy and efficiency of visual in-context learning by building upon existing frameworks and introducing novel approaches. The primary contribution is the innovative combination of prompt engineering, prompt ensembling, and prompt enhancing techniques to optimize performance. I introduce a approach for selecting and enhancing prompts, ensuring high relevance of in-context examples to the query image. This involves using state-of-the-art models like VQGAN-MAE for robust visual representations, further improved by prompt enhancers that add visual elements to better align in-context examples with the query image. Additionally, the prompt retriever leverages pre-trained feature extractors to automatically find the most suitable in-context examples, while prompt ensembling combines multiple prompts' outputs to enhance prediction robustness and accuracy. These contributions improve performance across various computer vision tasks, demonstrating the effectiveness of the ensembling and enhancing techniques for more accurate visual in-context learning.

4.1 Prerequisite

This section delves into the foundational models used in this research.

4.1.1 VQGAN Esser et al. (2021)

VQGAN (Vector Quantized Generative Adversarial Network) is a generative model known for producing high-resolution images by combining the strengths of transformers and CNNs. As discussed in the sections on Transformers (see section 2.2.2) and CNNs (see section 2.1.3), transformers excel in capturing long-range interactions and achieving state-of-the-art performance across various tasks, while CNNs focusing on local interactions. VQGAN leverages both capabilities by integrating a codebook mechanism to

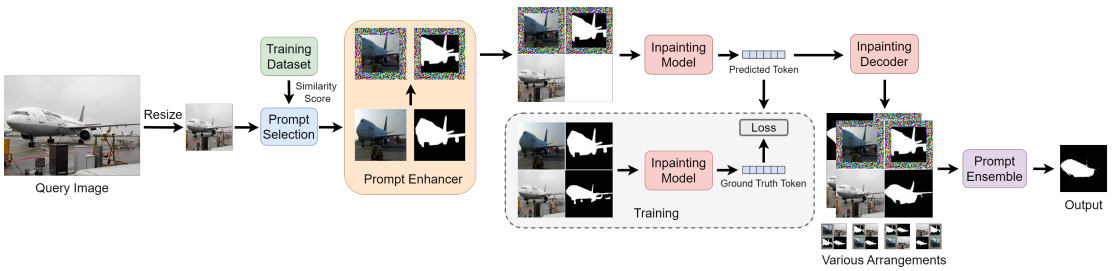

Figure 4.1: Overall framework of proposed method. First, the query image is resized into 111×111 size. Secondly, from prompt selection, a good in-context pair will be selected. Thirdly, with prompt enhancer, the selected in-context pair will be enhanced. The in-context example and query are concatenate into a four-cell grid image, with an empty cell on the bottom right.

encode visual tokens effectively.

In this study, I use a VQGAN model Bar et al. (2022) pre-trained on the ImageNet and Bar et al. (2022)'s Figure Datasets. This model features a codebook with a vocabulary size of 1024 vectors, which allows for a rich representation of visual data.

VQGAN Codebook Archetecture

Encoding VQGAN maps the input image into latent space, finding the nearest vector in the codebook. This vector serves as a discrete representation of the image's visual content, simplifying the model's complexity while maintaining essential details.

Decoding During decoding, the vector in the latent space is processed by the decoder to reconstruct the image. This step aims to generate an output image that reproduces the original image's visual aspects, leveraging the learned representations in the codebook.

4.1.2 MAE He et al. (2021)

MAE(Masked Autoencoder) is a neural network based on ViT Dosovitskiy et al. (2021) that is primarily used for self-supervised learning tasks. Designed to process large-scale image data efficiently, MAE does not require labeled data for training. MAE masks random parts of input images and encodes the visible parts. The decoder's job is to reconstruct the original image focusing on masked areas. The training objective is to minimize the construction error between original images and reconstructed images with loss calculated only on the masked sections.

During pretraining, only non-masked tokens are fed into the encoder, which is more efficient during training. Bar et al. (2022) uses checkpoint pretrained on ImageNet and further pretrained on Bar et al. (2022)'s dataset.

4.1.3 MAE-VQGAN

MAE-VQGAN Bar et al. (2022), a hybrid model combining Masked Autoencoder (MAE) and Vector Quantized Generative Adversarial Network (VQGAN), leverages the strengths of both architectures to enhance visual inpainting. Unlike traditional autoencoders that directly predict pixel values, MAE-VQGAN employs a softmax layer to assign probabilities to visual tokens, efficiently managing ambiguities through a probabilistic approach. During training, images are transformed into patches with random portions masked, and visible patches are encoded into latent vectors matched to the nearest vector in the VQGAN codebook. The decoding process uses these vectors to reconstruct the image, focusing on accurately predicting visual tokens for the masked sections using cross-entropy loss.

4.2 Visual Prompting Inpainting

Visual prompting in inpainting utilizes a specially designed visual prompt, which is a grid-like image made up of several image-label pairs and a query image. The primary objective is to predict the label of a query image, designated as y_q, with an input query image x_q and a set of prompt examples $S = \{(x_i, y_i)\}_{i=1}^{N}$, where x_i is the example image and y_i is the corresponding example label.

For this task, I employ the MAE-VQGAN model, an approach in Bar et al. (2022). This model is particularly suited for handling images where parts of the image are masked and need to be "in-painted" using context from the unmasked areas.

To use MAE-VQGAN inpainting an masked image, I first need to transform the set of example pairs S and the query image x_q into a format suitable for inpainting. This transformation presented by a mapping function f as described in Equation 4.1, converts the the these inputs into a new image and a corresponding mask:

$$[x_{vp}, m] = f(S, x_q) \tag{4.1}$$

x_{vp} is the transformed visual prompt and m is the mask area that inpainting model tries to predict.

The goal of inpainting model is to analyze the visual prompt x_{vp} and mask m as input and to generate an output y_{vp} that fills in the masked areas. The process is guided by the inpainting function g as described in Equation 4.2:

$$y_{vp} = g(x_{vp}, m) \tag{4.2}$$

Once the model has generated y_{vp}, the predicted label for the query image y_q can be extracted based on the areas specified by the mask m.

4.3 Visual In-context learning

Visual in-context learning leverage input query image x_q and a set of n in-context examples to predict the result y_q:

$$y_q = f_{ICL}(x_q|(x_i, y_i)_{i=1}^n) \tag{4.3}$$

This form of learning is highly dependent on the selection of in-context examples, since these examples significantly affects the model's capacity to learn and generalize, as highlighted by research Zhang et al. (2023b) Brown et al. (2020).

In this research, I focus primarily on optimizing visual in-context learning through the following method:

- Separate Processing with Ensembling: Each in-context example and the query image is processed independently as a discrete visual prompt. Subsequently, the model's predictions are aggregated to enhance accuracy and robustness.

Another viable method is concatenation of examples, where multiple in-context examples are merged into a single grid. However, this approach can significantly reduce the resolution of each example as more examples are added to the grid. The reduction in resolution may lead to a loss of critical visual details necessary for the model to make accurate predictions.

For a comprehensive discussion on the method, see the Section 4.6 on Prompt Ensembling .

4.4 Prompt Retriever

Zhang et al. (2023b) asserts that selecting an optimal in-context example is essential for improved prediction performance, suggesting that the best in-context example should be the one most contextually similar to the query image. Inspired by this, I introduce prompt retriever, automating the selection of the most appropriate prompt for a given query image x_q by defining this process as an optimization problem. This selection process is defined as an optimization problem where x^* is chosen by maximizing the score function f_θ:

$$x^* = arg \max_{x_n \in S} (x_n, x_q) \tag{4.4}$$

f_θ is a scoring function parameterized by θ, designed to assess similarity between the query image x_q and potential prompt images x_n from the dataset S. This scoring relies on feature vectors extracted by robust feature extractors like CLIP Radford et al. (2021) or ViT Dosovitskiy et al. (2021). The choice of extractor is important as it directly

Algorithm 1: Pseudo Code of Prompt Similarity Computation

Data: Images in the training dataset S; Images in the validation dataset V; The off-the-shelf feature extractor *img_encoder*; The source file names *source_names*; The target file names *target_names*

Result: A *.json* file containing all the query images name and its corresponding top-50 most similar prompt images names

```
Function Compute_similarity():
    /* initialize variables                                          */
    source_features ← torch.tensor([]).cuda()
    target_features ← torch.tensor([]).cuda()
    similarity_idx_dict ← dict()
    /* compute the features through image encoder                    */
    for img in S do
        source_features ← source_features + [img_encoder.forward_features(img)]
    for img in V do
        target_features ← target_features + [img_encoder.forward_features(img)]
    /* flatten the features                                          */
    source_features ← source_features.reshape(source_features.shape[0], -1)
    target_features ← target_features.reshape(target_features.shape[0], -1)
    target_norms ← norm(target_features, axis=1)
    /* Batch compute the cosine similarity to fit in the limited
       memory                                                        */
    batch_size ← 500
    for batch_start ← 0 to len(source_features) by batch_size do
        batch_end ← min(batch_start + batch_size, len(source_features))
        batch_features ← source_features[batch_start : batch_end]
        /* Compute dot product and normalize                         */
        dot_product ← dot(batch_features, transpose of target_features)
        source_batch_norms ← norm(batch_features, axis=1)
        similarity ← dot_product / dot(source_batch_norms, target_norms.T)
        /* Sort similarity and store results                         */
        for idx ← 0 to len(similarity) do
            similarity ← similarity[idx]
            sorted_indices ← argsort(similarity)[::-1]
            top_indices ← sorted_indices
            top_image_names ← [target_names[i] for i in top_indices]
            source_image_name ← source_names[batch_start + idx]
            similarity_idx_dict[source_image_name] ← top_image_names[:50]
    return similarity_idx_dict
```

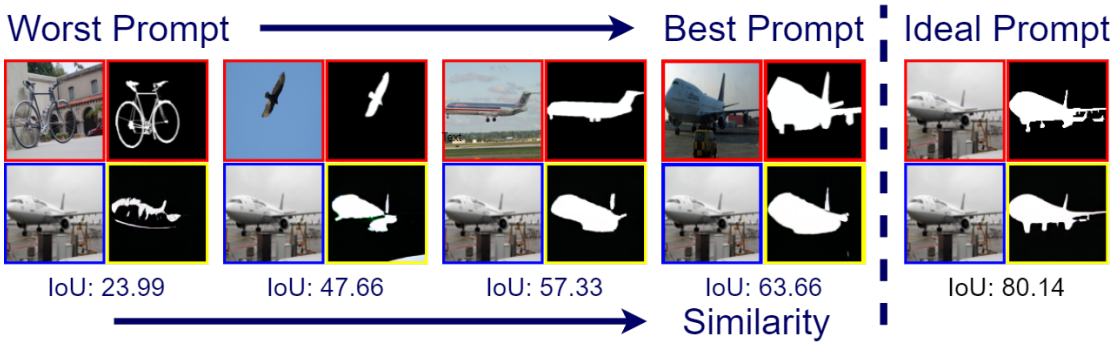

Figure 4.2: This example demonstrates the importance of image similarity in prompt selection for single object segmentation tasks. Performance improves with increasing similarity between the prompt image and the query image, with the ideal scenario being identical images. This illustrates prompt selection can enhance segmentation accuracy.

affects the quality of the feature representation and consequently the effectiveness of the prompt selection.

If the context requires selecting only one optimal prompt ($N = 1$), x^* is chosen directly. For $N > 1$, the top-n scoring prompts x_n are selected based on their scores.

The prompt retrieval process involves computing the cosine similarity between the query image x_q and each training example. This computation is computationally intensive and exceeds the current GPU memory when dealing with large datasets (fold-2 of pascal-5i dataset 4.7). To manage this, the similarity calculations are conducted in batches of 500 to optimize both computational resources and memory usage, as illustrated in Algorithm 1.

4.5 Prompt Enhancer

Despite careful prompt selection from existing datasets, finding the optimal prompt for each query image remains a challenge due to the finite size of the datasets. To bridge this gap, the prompt enhancer augments the retrieved in-context pairs to better align them with the query image.

The concept of embedding a visual prompt as a learnable parameter into a frozen large vision model was first introduced by Bahng et al. (2022). This technique mitigates domain shifts by enabling the use of source domain inputs for target domain tasks without the need to retrain the model.

Building on this foundation, Zhang et al. (2023a) introduced pixel-level prompts that wrap around the edges of images. This enhancement, similar to Bahng et al. (2022)'s initial approach, significantly boosts the effectiveness of visual inpainting models.

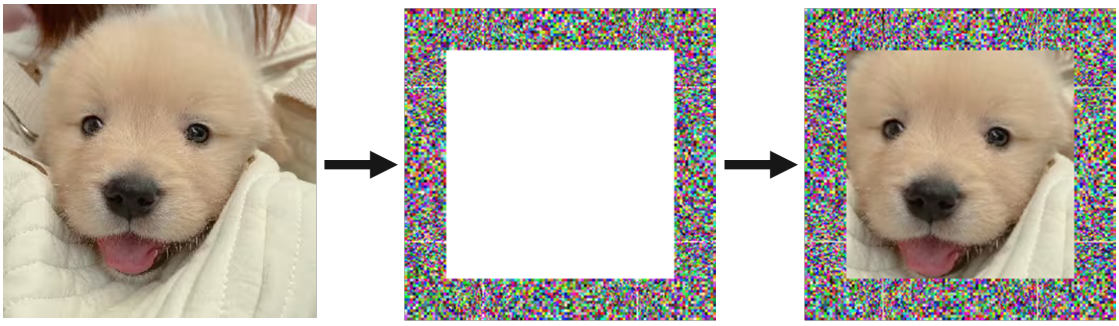

Figure 4.3: Prompt enhancer modifies prompts by adding visual elements such as borders, specifically tailored for each task.

Expanding on this concept, Huang et al. (2023) proposes the use of varied prompts for different data subsets rather than a single generic prompt for the entire dataset. This approach recognizes the diversity within large datasets and tailors the prompts to better suit the specific characteristics of each data subset.

The prompt enhancer adds a prompt parameterized by Φ to each in-context example (x, y) retrieved from the prompt retriever, generating a prompted pair:

$$x' = x + \delta t_\Phi$$
$$y' = y + \delta t_\Phi$$

where δ is a scaling factor and t_Φ comprises some pixels as the prompt while the other pixels remain zero.

These prompts are designed to be input-agnostic, with each task utilizing a uniquely trained prompt. By embedding learnable prompts within in-context pairs, the method not only directs the model towards the intended task but also increases the similarity between in-context examples and the query image, enhancing performance.

4.6 Prompt Ensembling

Inspired by strategies in Natural Language Processing (NLP) Lester et al. (2021), prompt ensembling is employed to enhance the robustness and accuracy of model predictions by integrating the outputs from multiple prompts. This technique not only improves prediction reliability but also removes potential biases or errors inherent in single prompt predictions.

Prompt ensembling involves:

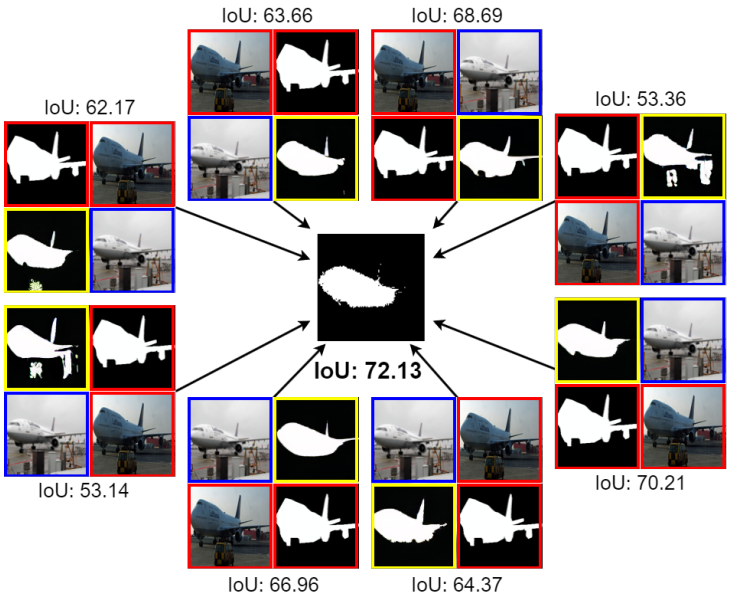

Figure 4.4: Different grid arrangements lead to varied model outputs. By employing a technique called prompt ensembling, where the final pixels are determined by majority voting across eight different grid arrangements, the quality of the final prediction is consistently enhanced.

- Constructing diverse visual prompts with different configurations of the mapping function f as described in Equation 4.1.
- Applying an inpainting function g as described in Equation 4.2 to each prompt to generate preliminary predictions.
- Combining the predictions from each prompt using methods like voting or a weighted average.

Given there are different ways to construct the the mapping function f, ensembling can be performed across diverse configurations, thereby maximizing the utilization of the MAE-VQGAN model's capabilities as demonstrated in Figure 4.4. This includes generating eight distinct grid arrangements to accommodate the inpainting model, each providing a unique prediction perspective. These predictions are then synthesized through a simple majority voting mechanism where a pixel in the final image is determined by a threshold of at least half (4 out of 8) of the predictions.

This study explores two distinct ensembling methodologies:

1. Ensembling on Multiple Examples: This method treats multiple examples as an individual image prompt. The outputs from each prompt are then ensembled, ensuring that the resolution of each prompt is maintained, which is crucial for maintaining detail and accuracy in the final output.

2. Ensembling on Diverse Mapping Functions: This approach involves applying different configurations of the mapping function f. The results from these diverse mappings are then aggregated to derive a robust and comprehensive prediction.

4.7 Dataset

PASCALVOC 2012 Everingham et al. is a dataset in the field of computer vision for object detection, classification, and segmentation tasks. It has been a benchmark on evaluating algorithms in these areas.

Pascal-5i dataset Shaban et al. (2017) is derived from PASCALVOC 2012 and SDS Hariharan et al. (2014), tailored for few-shot learning tasks. In Pascal-5i, it is about learning to recognize or segment objects using less than 5 examples.

The "$5i$" in Pascal-5i stands for 5 different folds in the datasets. From the original twenty semantic classes in PASCALVOC, 5 classes are sampled to form the test lable-set $L_{test} = \{4i + 1, ..., 4i + 5\}$, where i is the fold index. The remaining fifteen form the training label-set L_{train}.

Training set D_{train} includes all the image-mask pairs from both PASCALVOC and SDS training sets that contain at least one pixel of the semantic classes from L_{train}. The mask in D_{train} are modified that pixels not belonging to L_{train} are marked as background class $l_{\emptyset}$.

Test set contains image-label pairs from PASCALVOC validation set, corresponding to the test label-set L_{test}.

This setting of Pascal-5i dataset is designed to test the generalization ability of model from limited data to new unseen classes.

4.8 Downsteam Computer Vision Tasks

Foreground Segmentation. The objective of foreground segmentation is to distinguish and separate the foreground from the background in a query image. I uses Pascal-5i 4.7 dataset for training and evaluation. To assess performance, the pixels in the output image are classified into two categories: foreground (white) and background (black). The mean Intersection over Union (IoU) metric (see section 5.1.1) is employed to quantify the accuracy of the segmentation.

Single Object Detection. This task focuses on identifying a single object within an image. Following the approach used in Bar et al. (2022), the Pascal VOC 2012 dataset (see section 4.7) is employed, selecting images that contain only one object, with the stipulation that the annotation mask covers less than 50% of the image area.

4.9 PadPrompter Architecture

The PadPrompter is a specially designed neural network module designed for prompt enhancer. It is designed to apply a visual prompt to input images through the addition of learnable padding on all four sides. This modification enhances the input data with task-specific prompt, aiding in the model's training and inference processes.

The PadPrompter is configured with a parameter called 'pad size', which defines the thickness of the padding added around the input image. The standard size for input images is set at 224×224 pixels.

During inference, the PadPrompter dynamically adds the padding according to input images according to the task specified.

Chapter 5

Evaluation

5.1 Benchmark Set

5.1.1 Intersection over Union(IoU)

Intersection over Union (IoU) is a widely utilized metric in the field of computer vision, especially for evaluating the accuracy of object detection and segmentation models. The IoU metric measures the overlap between the predicted mask and the ground truth.

5.1.2 Calculation

The IoU is calculated as follows:

$$IoU = \frac{\text{Area of Intersection}}{\text{Area of Union}} \tag{5.1}$$

Here, the **Intersection** is the area where the predicted and the actual (ground truth) segmentations overlap, while the **Union** is the total area covered by both the predicted segmentation and the ground truth.

5.1.3 Merits

IoU is scale-invariant, meaning it provides a fair comparison even if the objects being segmented are of different sizes. Whether the object occupies a large portion of the image or a small portion, IoU will consistently measure the proportion of the overlap relative to the size of the object. Furthermore, IoU is particularly useful because it directly measures how much of the object is captured in the segmentation regardless of its shape. Due to its effectiveness, IoU has become a standard benchmark in the computer vision community. Its widespread use allows for easier comparison across different studies and benchmark datasets.

5.2 Hardware Setup

Table 5.1: This table lists the hardware used in the evaluation.

Component	Specification
Operating System	Ubuntu 22.04.3 LTS
CPU	i9-13900KF, 24 cores
GPU	NVIDIA GeForce RTX 4090
RAM	32GB

The system in Table 5.1 operates on Ubuntu 22.04.3 LTS with a powerful Intel i9-13900KF CPU that features 24 cores. It is equipped with an NVIDIA GeForce RTX 4090 GPU, which is highly capable for deep learning tasks, and 32GB of RAM to handle large datasets.

5.2.1 Library Version

Table 5.2: This table lists the library versions used in the evaluation.

Component	Specification
Cuda	12.1
torch	2.2.2
torchvision	0.17.2
torchaudio	2.2.2
pytorch_lightning	2.2.1
timm	0.8.3.dev0

The setup in Table 5.2 includes Cuda 12.1 for GPU acceleration, and PyTorch 2.2.2 along with its ecosystem libraries like torchvision, torchaudio, and pytorch_lightning, for training deep learning models. The 'timm' library version 0.8.3.dev0 supports various image models for benchmarking and feature extraction.

5.2.2 Model Version

The models are downloaded via 'timm' library images in Table 5.3.

Table 5.3: This table lists the model versions used in the evaluation.

Model	Version
CLIP	vit_large_patch14_224_clip_laion2b
ViT	vit_base_patch16_224.dino
ResNet	ResNet-18 & ResNet-50

5.3 Evaluated Method

5.3.1 Implementation Details

Evaluation All experiments use MAE-VQGAN Bar et al. (2022) as the pre-trained large-scale vision model. Compared methods include visual in-context learning (VP Bar et al. (2022), prompt-SelF Sun et al. (2023), InMeMo Zhang et al. (2023a)).

5.3.2 Implementation Details

For foreground segmentation, the model is trained separately for each fold of the training set 4.7, treating each fold as an independent task. This approach allows for the development of a unique, learnable prompt for each fold. In contrast, for single object detection, the model is trained on the entire training dataset. During testing, each image in the test set is treated as a query image to retrieve an in-context pair from the training set.

Images are resized to 224 $\times$ 224 pixels, and a learnable prompt that occupies x pixels from each edge is applied, resulting in model 4.9 containing $((224^2) - (224 - 2 \times x)^2)3$ parameters for each color channel in each pixel. Subsequently, images are resized again to 111×111 pixels to form the canvas. In-context pairs (x', y'), along with a query image x_q and an empty image mask m, are arranged in a grid with certain arrangements of different mapping function 4.1.

The implementation uses the PyTorch framework, and the model is trained over 100 epochs using the Adam optimizer Kingma and Ba (2017). Training begins with a learning rate of 40, which is adjusted according to a cosine annealing warm restarts schedule Loshchilov and Hutter (2017). This training regimen is executed efficiently on a single NVIDIA GeForce RTX 4090 GPU, with a batch size of 32.

5.4 Empirical Results

5.4.1 Comparison with State-of-the-Art

I compared my model with prior visual in-context learning methods methods in Table 5.4. My analysis reveals that my model achieved the SOTA, surpassing the previous SOTA in both downstream tasks. Specifically, my model, denoted as "Mine" in the

Table 5.4: Comparison of mIoU scores across different feature extraction methods for single object segmentation. Highest scores in each fold are marked in bold. Each method was evaluated without the use of a prompt enhancer.

Model	Fold-0	Fold-1	Fold-2	Fold-3	Mean	Det.
Random Bar et al. (2022)	27.49	30.21	25.88	24.06	26.91	25.45
baseline	35.05	37.17	33.59	32.59	34.60	27.05
prompt-SelF Sun et al. (2023)	42.48	43.34	39.76	38.50	41.02	29.83
InMeMo Zhang et al. (2023a)	41.65	47.68	42.43	**40.80**	43.14	43.21
Mine	**42.65**	**47.91**	**43.28**	40.26	**43.25**	**43.31**

table, achieves the highest mean Intersection over Union (mIoU) scores across nearly all the folds and maintains superior consistency in detection rates when compared to other state-of-the-art models such as InMeMo Zhang et al. (2023a) and prompt-SelF Sun et al. (2023)..

In the individual fold assessments, my model not only tops the charts in Folds 0, 1, and 2 with mIoU scores of 42.65, 47.91, and 43.28 respectively, it also competes closely in Fold-3. Although it does not achieve the highest score in Fold-3, the slight difference does not overshadow its overall superior performance with a mean mIoU of 43.25 and a remarkable detection rate of 43.31.

The results also demonstrate a substantial improvement over the generic baseline and significantly outperform the random model described by Bar et al. (2022).

5.4.2 Overfitting Analysis

As shown in Figure 5.1, the training loss (blue line) starts high and decreases sharply within the first few epochs, indicating rapid initial learning. After approximately 10 epochs, the training loss continues to decrease more gradually, reaching a relatively stable state but with minor fluctuations as training progresses.

Conversely, the validation IOU (orange line) initially increases sharply, suggesting improved model performance on the validation set during the early epochs. Around 10 epochs, the validation IOU stabilizes and fluctuates around a certain value, indicating the model's ability to generalize reasonably well to unseen data.

After about 100 epochs, the validation IOU shows a slight decreasing trend while the training loss continues to decrease, suggesting that the model might be starting to overfit the training data. This is a common indication that the model may be beginning to overfit the training data, learning the training set's noise or specific details rather than generalizing well to new data.

The periodic fluctuations in both training loss align with the expected behavior of cosine annealing with warm restarts 2.6, where the learning rate is periodically reduced and then

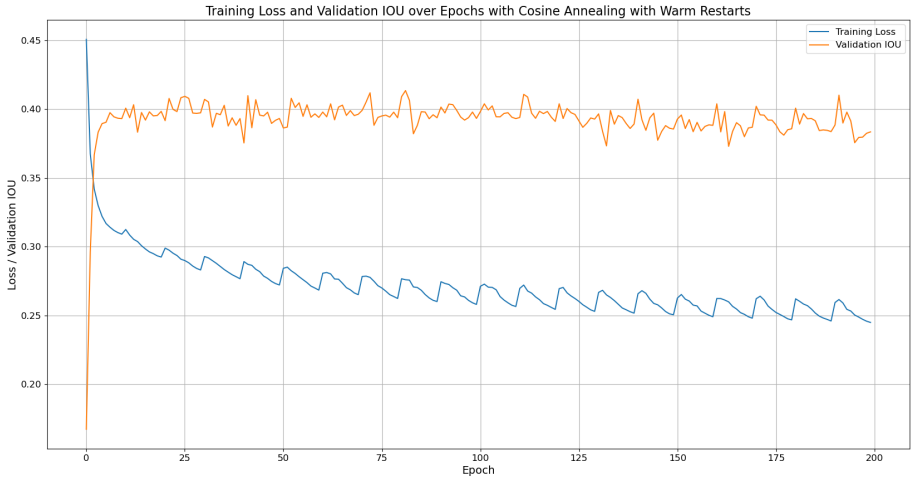

Figure 5.1: Training Loss and Validation IOU over 200 Epochs using Cosine Annealing with Warm Restarts.

increased, promoting better convergence and potentially escaping local minima. This strategy helps to balance minimizing training loss and maintaining reasonable validation performance, thus mitigating some degree of overfitting.

5.4.3 Prompt Selection

The feature extractor in prompt selection is crucial for extracting features from both the prompt and query images and identifying the most suitable prompt based on these features. It is essential that the feature extractor effectively captures and conveys semantic information across various regions of the image. To ensure optimal performance, I conducted experiments with several feature extractors, comparing their ability to discern relevant features for effective prompt selection.

To conduct a comprehensive comparison, I selected three widely-used feature extractors: ViT Dosovitskiy et al. (2021) and CLIP Radford et al. (2021), both based on transformer architectures, and ResNet He et al. (2015), which utilizes a CNN architecture. Each was chosen for its distinct approach to processing and understanding image features, allowing us to assess their effectiveness across a range of visual prompting scenarios.

The results presented in Table 5.5 reveal key insights into the performance of different feature extractors used for single object segmentation. Notably, ResNet-50 consistently performs better than ResNet-18 in all tests, showing its stronger ability to manage complex image details for segmentation tasks. Furthermore, while CLIP consistently surpasses the performance of ViT, it is slightly outmatched by ResNet-50 in terms of mean mIoU scores.

I have selected CLIP for my prompt selection tasks. This decision is based on CLIP's unique ability to align text and image representations, making it particularly suited for tasks where understanding the semantic context provided by prompts. This is especially critical for in-context learning, where the relevance and applicability of prompts significantly influence task performance.

Table 5.5: Comparison of mIoU scores across different feature extraction methods for single object segmentation. Highest scores in each fold are marked in bold. Each method was evaluated without the use of a prompt enhancer.

Selection Method	Fold-0	Fold-1	Fold-2	Fold-3	Mean (mIoU)
Random Bar et al. (2022)	27.49	30.21	25.88	24.06	26.91
CLIP	35.05	37.17	**33.59**	**32.59**	34.60
ViT	32.01	35.91	33.44	32.39	33.43
ResNet-18	35.97	36.98	32.62	31.88	34.36
ResNet-50	**36.03**	**37.24**	32.91	32.89	**34.76**

5.4.4 Prompt Ensembling

Table 5.6: Comparison of mIoU scores across different learning methods for single object segmentation. Highest scores in each fold are marked in bold. Each method was evaluated without the use of a prompt enhancer. Voting 4 indicates that a pixel is considered part of the segmentation if at least four arrangements agree.

Method	Fold-0	Fold-1	Fold-2	Fold-3	Mean (mIoU)
Without Ensembling	35.05	37.17	33.59	32.59	34.60
With Ensembling(voting 4)	35.67	39.48	35.85	35.80	36.70

The method using prompt ensembling consistently outperforms the approach without ensembling across all folds. This indicates that combining predictions from multiple prompts can significantly enhance model accuracy by incorporating a broader range of perspectives and fully explore the potential of inpainting model.

As illustrated in Figure 5.2, employing ensembling strategies markedly enhances segmentation performance. The outcomes reveal that a moderate voting threshold—specifically between 3 to 5—strikes an optimal balance. This range is effective at filtering out noise and inconsistencies between different layout arrangements, thus stabilizing the segmentation output. Voting thresholds lower than 3 tend to include more noise, whereas thresholds higher than 5 may overlook essential details. This insight underscores the utility of an intermediate threshold in ensembling strategies to achieve the best trade-off between sensitivity and specificity in image segmentation tasks.

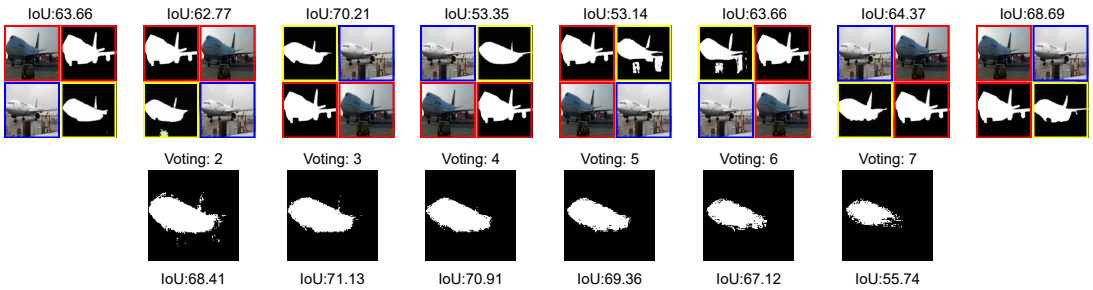

Figure 5.2: The figure demonstrates the effectiveness of various ensembling strategies using different voting thresholds. The first row displays the segmentation outputs under different layout arrangements, while the second row shows the aggregated results from the ensembling process. "Voting: 2" indicates that a pixel is considered part of the segmentation if at least two arrangements agree.

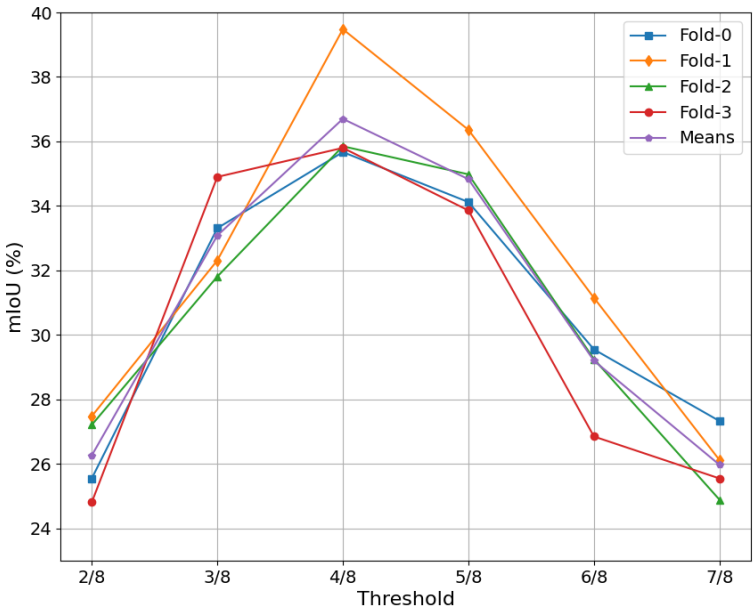

Figure 5.3: Performance analysis of different voting thresholds in ensemble segmentation. The thresholds range from 2/8 to 7/8, where the numerator indicates the minimum number of agreeing arrangements required out of eight to confirm pixel inclusion in the final segmentation mask.

Figure 5.3 illustrates the impact of various voting thresholds on the mean Intersection over Union (mIoU) scores across different dataset folds. The optimal performance is

achieved at a threshold of 4/8, where exactly half of the arrangements must agree to include a pixel in the final segmentation output. This threshold effectively balances sensitivity and specificity, allowing for a more accurate and reliable segmentation while minimizing the impact of outlier arrangements. This detailed analysis confirms the observation from Figure 5.2, emphasizing that a balanced threshold can substantially improve segmentation performance across diverse dataset conditions.

5.4.5 Visual Prompt Design

Visual prompt here refers to the prompts generated by the prompt enhancer 4.5. Choosing the appropriate prompt is crucial to the performance. Bigger prompt means more model parameters making the model less efficient and potential occlusion on the detection objects.

Visual Prompt Size Choosing the appropriate prompt size is crucial to optimizing model performance in visual tasks. The following analysis demonstrates the impact of prompt size on the effectiveness of the model for specific applications. I performed complete experiment on various prompt sizes to assess their performance impact. These experiments were designed to identify the optimal prompt size for both single object detection and segmentation tasks. For single object detection, the performance across different prompt sizes is detailed in Table 5.7. Similarly, the results for segmentation tasks are presented in a separate table, Table 5.8.

Table 5.7: The mIoU scores for varying padding sizes of prompt enhancer t_Φ on detection task. The highest score in each fold is marked in bold.

Padding Size	Detection
0	27.05
1	30.01
5	40.72
10	41.10
15	41.54
20	42.71
25	43.35
30	43.52
35	40.53
40	42.46
45	**44.37**
50	43.40
55	43.93
60	41.57

Table 5.8: The mIoU scores for varying padding sizes of prompt enhancer t_Φ on segmentation task. The highest score in each fold is marked in bold.

Padding Size	Fold-0	Fold-1	Fold-2	Fold-3
0 *(baseline)*	35.05	37.17	33.59	32.59
10	39.02	45.06	41.23	40.66
20	40.17	**46.64**	**41.79**	39.63
30	**41.34**	45.78	41.11	**41.06**
40	25.91	28.11	26.21	25.39
50	24.49	30.42	26.37	27.25

The tables outline the impact of different padding sizes on the model's performance in detection and segmentation. For detection task, increasing the padding size generally enhances the model's accuracy, as more contextual information around the object is included, helping the model make better predictions. The highest accuracy is achieved with a padding size of 45, where the mIoU peaks at 44.37. However, for segmentation, padding sizes of 20 and 30 achieve comparably high performance, indicating that a smaller padding can be just as effective for this task.

I have chosen a padding size of 30 for practical use. This choice balances between providing sufficient contextual detail for accurate detection and segmentation and maintaining operational efficiency. It ensures that the model operates swiftly and conservatively in terms of resource usage, which is vital in real-world scenarios where processing speed and efficient resource management are crucial. In detection tasks, the performance at a padding size of 30 is comparable to the peak performance observed at a padding size of 45, further validating this choice for both detection and segmentation tasks.

Additionally, a smaller padding size like 30 is more likely to generalize better across various operational conditions. It avoids potential overfitting that larger padding sizes might cause when they include too much irrelevant information or noise from the surroundings or occlude the segmentation object. This makes the model more robust and reliable across different environments and object types.

5.4.6 Prompt Enhancer

The use of a prompt enhancer significantly boosts performance for the specific grid arrangement the model is initially trained on. However, this comes at the expense of reducing the model's generalization ability across other grid configurations. This effect is illustrated in Figure 5.4, where different grid layouts are tested with and without the prompt enhancer.

When the enhancer is applied, the model's performance on the trained layout sees noticeable improvement as shown in Table 5.8 and 5.7. However, for alternative layouts,

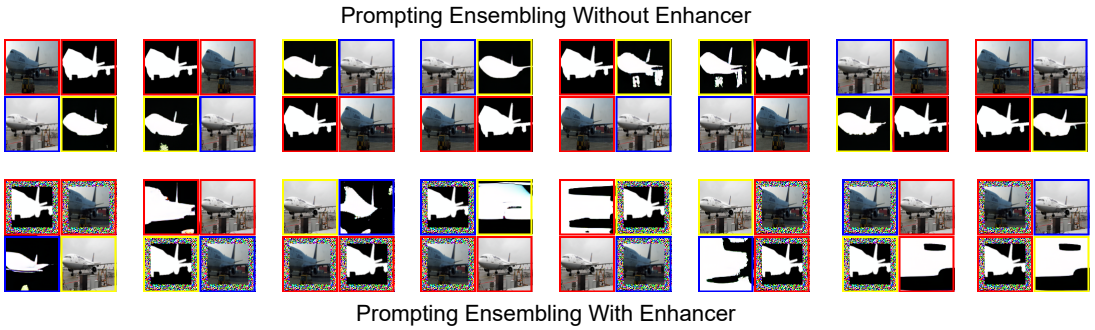

Figure 5.4: The cell with yellow border is the model prediction result for each arrangements. When with the enhancer, the output for different layouts other than the model trained for is meaning less. However, without the enhancer, the output for different layouts are good.

the prediction results become significantly worse, suggesting that the enhancer might be overfitting the model to a specific configuration. This limitation is evident as the model, when equipped with the enhancer, fails to interpret the visual prompts effectively if they deviate from the trained arrangement.

Conversely, without the enhancer, the model demonstrates robust generalization capabilities. The predictions across different layouts remain consistently accurate, underlining the model's versatility and its ability to adapt to varying visual contexts without the constraint of the enhancer.

This observation indicates a trade-off between specialized performance enhancement and broad-based model adaptability.

5.4.7 More Visual In-context Examples

In the realm of Visual In-context Learning, the principle that multiple in-context examples generally yield better results than a single example—akin to few-shot learning in Natural Language Processing (NLP)—is well-documented. This parallels findings in NLP, where incorporating multiple context examples significantly boosts learning efficacy, particularly in complex interpretation tasks.

As depicted in Figure 5.5, employing a greater number of highly similar in-context examples consistently improves performance. This strategy underscores the importance of careful prompt selection in visual in-context learning. It highlights how integrating multiple predictions can significantly refine the accuracy of query image predictions.

The variability in performance also brings to light the fact that prompts selected during the retrieval process might not always be optimal. Therefore, using a broader array of in-context examples is crucial for enhancing the robustness of the model. This approach not only compensates for potential inaccuracies of individual examples but also intro-

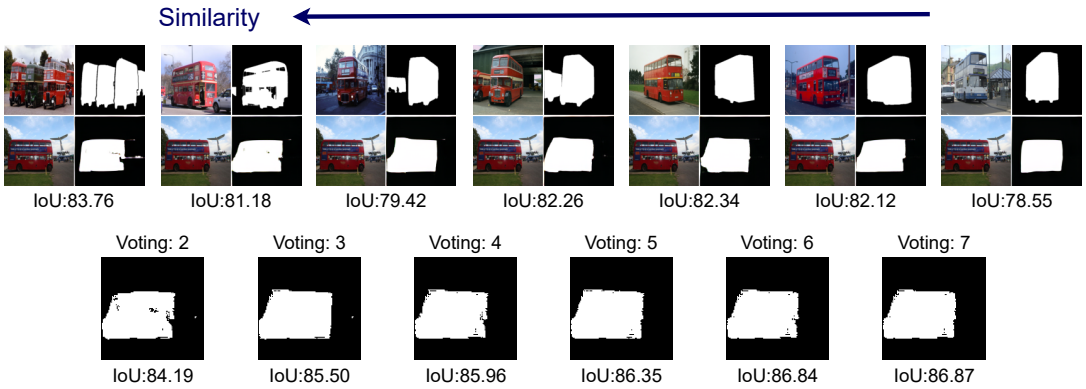

Figure 5.5: Increasing the number of similar in-context examples and employing ensembling strategies can significantly enhance IoU scores for visual in-context learning. The examples are arranged by decreasing similarity, showing the direct correlation between similarity and segmentation accuracy. Voting: 2 means using the first two results for ensembling.

duces a level of redundancy, safeguarding the segmentation results against diverse input variations and enhancing the overall reliability of the system.

By drawing on multiple examples and leveraging ensembling techniques, visual in-context learning can achieve higher precision and stability, mirroring successful strategies employed in NLP.

5.4.8 Cross-Fold Generalization

During the training phase, a unique prompt is tailored for each fold, optimized specifically for that subset. This specialization is reflected in the performance metrics across different folds, as detailed in Table 5.9, which illustrates the generalization capabilities of prompts trained fold-wise. The data indicates that each fold-specific prompt significantly enhances performance on the fold it was trained on compared to other folds.

Furthermore, the results suggest that a prompt trained for a particular class or fold does not generalize effectively to other classes. However, in comparison to a baseline model with no padding (padding size 0) in Table 5.8, all configurations with a prompt (padding size 30) yield superior IoU results, demonstrating the effectiveness of using tailored prompts in enhancing model accuracy and robustness.

This analysis reveals that while fold-specific prompts considerably boost performance within their respective folds, their generalization to other folds is limited.

Table 5.9: Generalization performance of model prompts trained and tested across different folds. The rows represent the fold the prompt enhancer was trained on, and the columns indicate the target fold for testing. The best performances within each training fold are highlighted in bold. This setup evaluates the ability of prompts to generalize across different subsets of data under the condition of using a prompt padding size of 30.

	Fold-0	**Fold-1**	**Fold-2**	**Fold-3**
Fold-0	**41.34**	39.20	33.64	37.36
Fold-1	37.25	**45.78**	36.02	37.84
Fold-2	39.62	43.41	**41.11**	36.84
Fold-3	36.52	41.85	36.30	**41.06**

Table 5.10: Domain shift analysis: 'Pascal → Pascal' indicates both in-context pairs and query images sourced from PASCAL; 'COCO → Pascal' indicates in-context pairs from COCO and query images from PASCAL. The baseline scores are our reproduction.

Dataset Transfer	**Method**	**Fold-0**	**Fold-1**	**Fold-2**	**Fold-3**	**mIoU**
Pascal → Pascal	Baseline	35.05	37.17	33.59	32.59	34.60
Pascal → Pascal	My Model	41.78	47.79	42.59	41.03	44.45
COCO → Pascal	Baseline	33.72	36.66	32.68	30.38	33.36
COCO → Pascal	My Model	38.18	42.92	40.69	37.74	39.88

5.4.9 Domain Shift Analysis

In real-world scenarios, models frequently encounter data that differ significantly from the data they were trained on. Such domain shifts can lead to decreased accuracy and robustness, making it crucial to evaluate how well a model can adapt to new environments. This evaluation is particularly relevant for applications where deploying a model across different geographic regions or different conditions is necessary.

To rigorously test the model's adaptability, I utilize the COCO citelin2015microsoft dataset for training in-context pairs while sourcing the query images from the PASCAL VOC dataset. The COCO dataset is divided into four subsets, each denoted as COCO-5i Zhang et al. (2023b) mirrors the categories of Pascal-5i.

The experimental configuration, termed 'COCO → Pascal,' involves using in-context pairs from a version of the COCO dataset adapted for compatibility with PASCAL VOC's structure, referred to as COCO-5i. This approach ensures that the categories between the two datasets align, providing a fair ground for evaluation. The results, summarized in Table 5.10, illustrate how the model performs under this cross-dataset

condition.

From the table, it is evident that both the baseline and My Model perform better when both training and testing occur within the same dataset (Pascal → Pascal). However, the performance degradation is less severe for My Model, indicating superior generalization capabilities compared to the baseline. When the training happens on COCO and testing on Pascal, there is a noticeable drop in performance for both models, but My Model shows a lesser decline, suggesting it handles domain shifts better than the baseline setup.

Chapter 6

Concluding Remarks

6.1 Conclusion

This study delves into the key aspects that influence visual in-context learning, focusing on prompt selection, enhancement, and ensembling strategies. My research demonstrates how effective prompt selection identifies the optimal prompts for each query image, how strategic prompt enhancement significantly boosts model accuracy, and how prompt ensembling leverages collective insights from multiple configurations to maximize the efficacy of large-scale visual models. Through extensive experimental validation, my approaches have proven to be highly effective in optimizing visual in-context learning. Additionally, these methods have shown remarkable robustness in handling domain shifts, maintaining high performance across diverse datasets and scenarios.

6.2 Limitations and Future Work

Prompt Location Optimization The location of visual prompts plays a crucial role in model performance. Overlapping between the visual prompts from the prompt enhancer and segmentation objects can adversely affect predictions, as demonstrated in Figure 6.1. Future improvements could involve strategies to prevent overlap, such as dynamically adjusting prompt locations based on the object's position or omitting borders when they would intersect with key image features.

Enhancing Prompt Selection The current mechanism for selecting prompts often results in choices that, although visually similar, do not significantly contribute to improving model efficacy. This problem comes from an over-reliance on superficial similarity metrics, which do not always account for the semantic relevance or contextual applicability of the prompts. For instance, as illustrated in Figure 6.2, a bicycle might be incorrectly paired with a duck due to superficial feature alignments such as texture

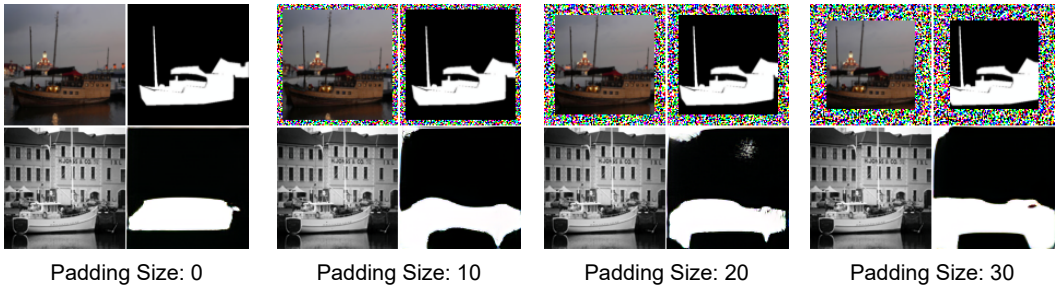

Figure 6.1: Impact of prompt location on prediction accuracy with varying border sizes.

or color, which is not beneficial for task-specific model training. To refine this process, the selection strategy should incorporate:

- Semantic Alignment: Enhancing the selection algorithm to evaluate the semantic similarity between the prompt and the target object. This involves assessing similarities in color, shapes, object size, and object class.
- Feature Localization: Shifting from global image features to localized feature analysis. This approach focuses on critical attributes of the object, improving the relevance of the prompt selection by aligning key visual features that are pivotal for accurate model training.
- Extra Layer: Introducing a extra assessment layer to the selection process. This involves quantitatively measuring the impact of a prompt on the model's performance through controlled neural network, thereby prioritizing prompts that offer optimal performance enhancements.

Improving Class-Specific Predictions The performance of the model on specific classes, such as bottles, highlights a significant challenge. Despite the incorporation of a visual prompt enhancer, the prediction results for the bottle class remain suboptimal and, in some cases, entirely non-sensical. This issue is illustrated in Figure 6.3, where the outputs for the bottle class fail to depict any meaningful segmentation, sometimes resulting in completely blank outputs.

To address this, exploring alternative models with enhanced generalization capabilities could be beneficial. One potential approach is to utilize CLIP (Contrastive Language–Image Pre-training) Radford et al. (2021), which has demonstrated superior performance in understanding textual descriptions with visual content. Implementing CLIP as the foundational model could leverage its robust feature extraction capabilities to better recognize objects like bottles.

Another strategy might involve refining the MAE-VQGAN model Bar et al. (2022) to enhance its sensitivity to class-specific features. This could include adjusting the training regime to focus on bottle classes or redesigning the architecture to incorporate

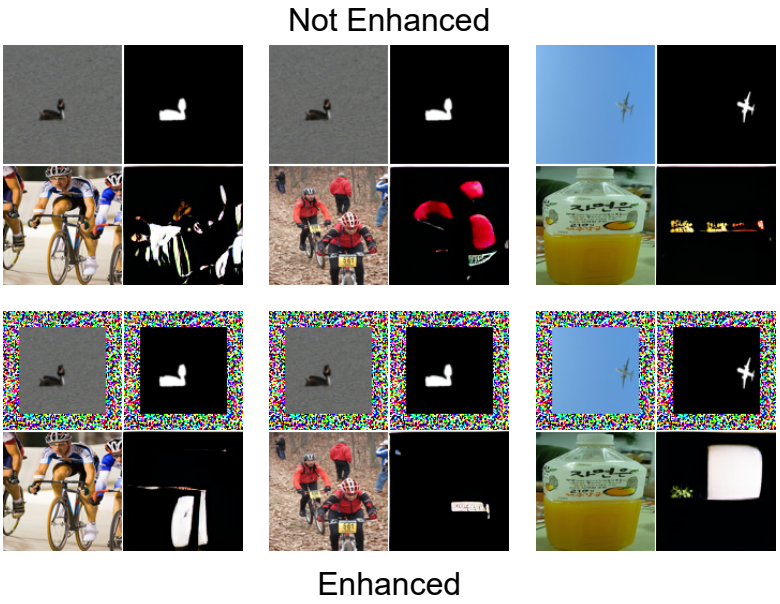

Figure 6.2: Challenges with current feature extractor leading to suboptimal prompt selection.

more class-specific attention mechanisms focusing on challenging object classes.

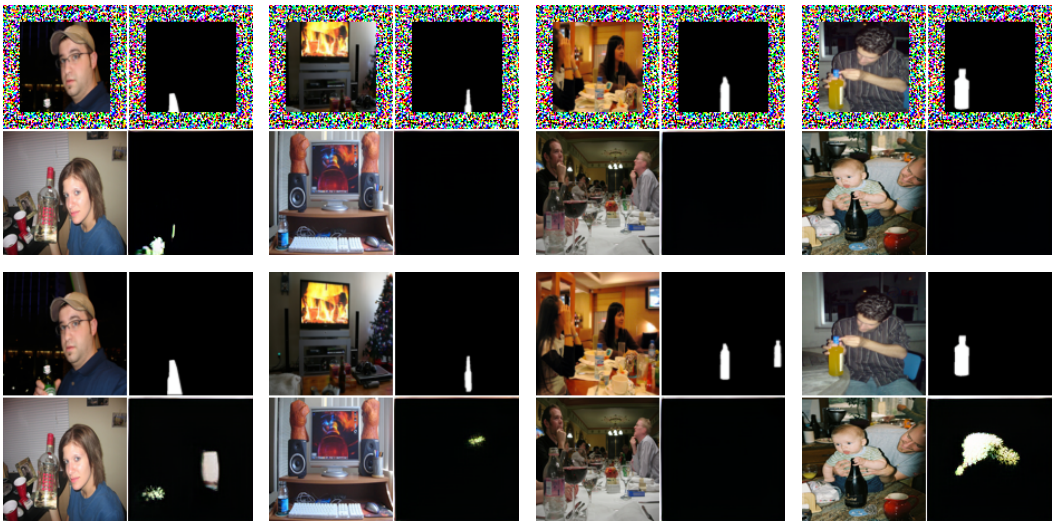

Figure 6.3: Inadequate prediction results for the bottle class using current models, showcasing the necessity for model refinement or replacement. Right bottom cell of each grid is the prediction result.

Cross-Fold Generalization As evidenced in Section 5.4.8, prompts that are tailored to specific folds lead to superior performance within those folds. However, this restricts their effectiveness across other folds, highlighting a critical gap in the model's generalization capabilities.

To bridge this gap, it is crucial to develop models that are not only robust within specific contexts but also adaptable across diverse scenarios. One effective strategy to achieve this balance is the implementation of regularization techniques. Incorporating methods such as dropout, L1/L2 regularization, or employing early stopping during training can mitigate the risk of overfitting to the unique characteristics of a single fold's data. These approaches help in promoting the extraction of features and learning of patterns that hold general applicability across various datasets, enhancing the model's overall generalization capabilities.

Combining Prompt Enhancer with Prompt Ensembling Figure 5.4 illustrates that while the application of a prompt enhancer significantly improves model accuracy on its trained configuration, it negatively impacts performance on untrained layouts. This phenomenon suggests a potential overfitting of the model to specific input arrangements, compromising its ability to generalize to alternative configurations.

A proposed solution to mitigate this issue involves training the model across all possible prompt arrangements and employing prompt ensembling techniques to synthesize the outputs. This approach would potentially increase the robustness and adaptability of the model but at the cost of requiring extensive computational resources and training time.

However, the additional computational expense required to train across multiple arrangements needs careful consideration, especially in environments where efficiency is a priority.

Bibliography

Alayrac, J.-B.; Donahue, J.; Luc, P.; Miech, A.; Barr, I.; Hasson, Y.; Lenc, K.; Mensch, A.; Millican, K.; Reynolds, M.; Ring, R.; Rutherford, E.; Cabi, S.; Han, T.; Gong, Z.; Samangooei, S.; Monteiro, M.; Menick, J.; Borgeaud, S.; Brock, A.; Nematzadeh, A.; Sharifzadeh, S.; Binkowski, M.; Barreira, R.; Vinyals, O.; Zisserman, A.; and Simonyan, K., 2022. Flamingo: a visual language model for few-shot learning. [Cited on pages 9 and 13.]

Bahng, H.; Jahanian, A.; Sankaranarayanan, S.; and Isola, P., 2022. Exploring visual prompts for adapting large-scale models. [Cited on pages 3, 8, 14, and 24.]

Bar, A.; Gandelsman, Y.; Darrell, T.; Globerson, A.; and Efros, A. A., 2022. Visual prompting via image inpainting. [Cited on pages 2, 8, 9, 13, 14, 15, 20, 21, 27, 31, 32, 34, and 44.]

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D. M.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D., 2020. Language models are few-shot learners. [Cited on pages 1, 7, 8, and 22.]

Bulat, A. and Tzimiropoulos, G., 2023. Lasp: Text-to-text optimization for language-aware soft prompting of vision language models. [Cited on page 17.]

Chen, A.; Yao, Y.; Chen, P.-Y.; Zhang, Y.; and Liu, S., 2023. Understanding and improving visual prompting: A label-mapping perspective. [Cited on pages 3 and 13.]

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K., 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. [Cited on page 8.]

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; Uszkoreit, J.; and Houlsby, N., 2021. An image is worth 16x16 words: Transformers for image recognition at scale. [Cited on pages 2, 3, 6, 8, 14, 20, 22, and 33.]

Esser, P.; Rombach, R.; and Ommer, B., 2021. Taming transformers for high-resolution image synthesis. [Cited on pages v, 14, and 19.]

Everingham, M.; Van Gool, L.; Williams, C. K. I.; Winn, J.; and Zisserman, A. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html. [Cited on page 27.]

Hariharan, B.; Arbeláez, P.; Girshick, R.; and Malik, J., 2014. Simultaneous detection and segmentation. [Cited on page 27.]

He, K.; Chen, X.; Xie, S.; Li, Y.; Dollár, P.; and Girshick, R., 2021. Masked autoencoders are scalable vision learners. [Cited on pages v, 9, 13, and 20.]

He, K.; Zhang, X.; Ren, S.; and Sun, J., 2015. Deep residual learning for image recognition. [Cited on page 33.]

Huang, Q.; Dong, X.; Chen, D.; Zhang, W.; Wang, F.; Hua, G.; and Yu, N., 2023. Diversity-aware meta visual prompting. [Cited on pages 14, 18, and 25.]

Jia, M.; Tang, L.; Chen, B.-C.; Cardie, C.; Belongie, S.; Hariharan, B.; and Lim, S.-N., 2022. Visual prompt tuning. [Cited on page 17.]

Khattak, M. U.; Rasheed, H.; Maaz, M.; Khan, S.; and Khan, F. S., 2023. Maple: Multi-modal prompt learning. [Cited on page 16.]

Kingma, D. P. and Ba, J., 2017. Adam: A method for stochastic optimization. [Cited on pages 10 and 31.]

Kirillov, A.; Mintun, E.; Ravi, N.; Mao, H.; Rolland, C.; Gustafson, L.; Xiao, T.; Whitehead, S.; Berg, A. C.; Lo, W.-Y.; Dollár, P.; and Girshick, R., 2023. Segment anything. [Cited on page 15.]

Lester, B.; Al-Rfou, R.; and Constant, N., 2021. The power of scale for parameter-efficient prompt tuning. [Cited on pages 4, 8, 17, and 25.]

Li, J.; Li, D.; Savarese, S.; and Hoi, S., 2023. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. [Cited on page 13.]

Li, X. L. and Liang, P., 2021. Prefix-tuning: Optimizing continuous prompts for generation. [Cited on pages 8 and 17.]

Liu, H.; Li, C.; Wu, Q.; and Lee, Y. J., 2023. Visual instruction tuning. [Cited on page 13.]

Liu, J.; Shen, D.; Zhang, Y.; Dolan, B.; Carin, L.; and Chen, W., 2021. What makes good in-context examples for gpt-3? [Cited on page 15.]

Loshchilov, I. and Hutter, F., 2017. Sgdr: Stochastic gradient descent with warm restarts. [Cited on pages 11 and 31.]

Oh, C.; Hwang, H.; young Lee, H.; Lim, Y.; Jung, G.; Jung, J.; Choi, H.; and Song, K., 2023. Blackvip: Black-box visual prompting for robust transfer learning. [Cited on page 3.]

Puri, R., 2023. Gpt-4v(ision) system card. *OpenAi*, (2023). [Cited on page 13.]

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; Krueger, G.; and Sutskever, I., 2021. Learning transferable visual models from natural language supervision. [Cited on pages 2, 3, 8, 16, 22, 33, and 44.]

Shaban, A.; Bansal, S.; Liu, Z.; Essa, I.; and Boots, B., 2017. One-shot learning for semantic segmentation. [Cited on page 27.]

Shtedritski, A.; Rupprecht, C.; and Vedaldi, A., 2023. What does clip know about a red circle? visual prompt engineering for vlms. [Cited on page 16.]

Subramanian, S.; Merrill, W.; Darrell, T.; Gardner, M.; Singh, S.; and Rohrbach, A., 2022. Reclip: A strong zero-shot baseline for referring expression comprehension. [Cited on page 16.]

Sun, X.; Hu, P.; and Saenko, K., 2022. Dualcoop: Fast adaptation to multi-label recognition with limited annotations. [Cited on page 18.]

Sun, Y.; Chen, Q.; Wang, J.; Wang, J.; and Li, Z., 2023. Exploring effective factors for improving visual in-context learning. [Cited on pages 3, 15, 31, and 32.]

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I., 2023. Attention is all you need. [Cited on page 6.]

Wang, X.; Wang, W.; Cao, Y.; Shen, C.; and Huang, T., 2023. Images speak in images: A generalist painter for in-context visual learning. [Cited on pages 2, 3, 9, and 14.]

Wu, J.; Li, X.; Wei, C.; Wang, H.; Yuille, A.; Zhou, Y.; and Xie, C., 2023. Unleashing the power of visual prompting at the pixel level. [Cited on page 14.]

Yao, Y.; Zhang, A.; Zhang, Z.; Liu, Z.; Chua, T.-S.; and Sun, M., 2022. Cpt: Colorful prompt tuning for pre-trained vision-language models. [Cited on page 16.]

Zang, Y.; Li, W.; Zhou, K.; Huang, C.; and Loy, C. C., 2022. Unified vision and language prompt learning. [Cited on page 16.]

Zhang, J.; Wang, B.; Li, L.; Nakashima, Y.; and Nagahara, H., 2023a. Instruct me more! random prompting for visual in-context learning. [Cited on pages 14, 24, 31, and 32.]

Zhang, Y.; Zhou, K.; and Liu, Z., 2023b. What makes good examples for visual in-context learning? [Cited on pages 3, 15, 22, and 40.]

Zhou, K.; Yang, J.; Loy, C. C.; and Liu, Z., 2022a. Conditional prompt learning for vision-language models. [Cited on pages 17 and 18.]

Zhou, K.; Yang, J.; Loy, C. C.; and Liu, Z., 2022b. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130, 9 (Jul. 2022), 2337–2348. doi:10.1007/s11263-022-01653-1. http://dx.doi.org/10.1007/s11263-022-01653-1. [Cited on page 17.]

Zhou, Y.; Li, X.; Wang, Q.; and Shen, J., 2024. Visual in-context learning for large vision-language models. [Cited on page 13.]

Zhu, D.; Chen, J.; Shen, X.; Li, X.; and Elhoseiny, M., 2023. Minigpt-4: Enhancing vision-language understanding with advanced large language models. [Cited on page 13.]

Zou, X.; Yang, J.; Zhang, H.; Li, F.; Li, L.; Wang, J.; Wang, L.; Gao, J.; and Lee, Y. J., 2023. Segment everything everywhere all at once. [Cited on page 15.]