Universiteit Leiden

Master Computer Science

Consistency Regularization for Test-time Prompt Tuning

Name:	Joël During
Student ID:	s3558339
Date:	05/08/2024
Specialisation:	Artificial Intelligence
1st supervisor:	Prof.dr. M.S.K. Lew
2nd supervisor:	Dr. E.M. Bakker

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Universiteit Leiden

TNO innovation for life

Consistency Regularization for Test-time Prompt Tuning

Joël During (s3558339)

A thesis presented for the degree of
Master Computer Science
Specialisation in Artificial Intelligence
at Leiden University under supervision of
Prof.dr. M.S.K. Lew
and
Dr. E.M. Bakker

Research performed during an internship at
Netherlands Organisation for Applied Scientific Research (TNO)
Under supervision of
Sabina van Rooij, MSc

August 2024

ABSTRACT

Vision-Language Models (VLMs) have been intensely investigated recently. These multimodal models learn vision-language relationships from large amounts of noisy image-text pairs. The large scale at which these models are trained and their understanding of relationships between concepts in both the visual and textual domains gives them promising zero-shot generalization capabilities on various visual downstream tasks. For image classification, a VLM such as Contrastive Language-Image Pre-training (CLIP) can reach competitive zero-shot performance on standard benchmarks when using suitable textual prompts. These prompts can be found through manual prompt engineering or by prompt tuning through optimization methods. In the latter case, labeled data of the target domain is required. To enable on-the-fly optimization of prompts at inference time, Test-time Prompt Tuning (TPT) utilizes Test-Time Adaptation (TTA) techniques to optimize prompts using only unlabeled data. We investigate the unsupervised learning methods behind TPT and find that they can often be omitted without affecting performance. We hypothesize that these unsupervised learning methods can produce unwanted results when the model produces different predictions for multiple augmentations of the same image. To test our hypothesis, we propose two methods for aligning the predictions for all augmentations of a single image. We show that our method, which uses consistency regularization to encourage consistent predictions, outperforms standard TPT on seven out of ten fine-grained classification tasks. Furthermore, we show that our method can learn from using more augmentations or more severe augmentation, while standard TPT cannot.

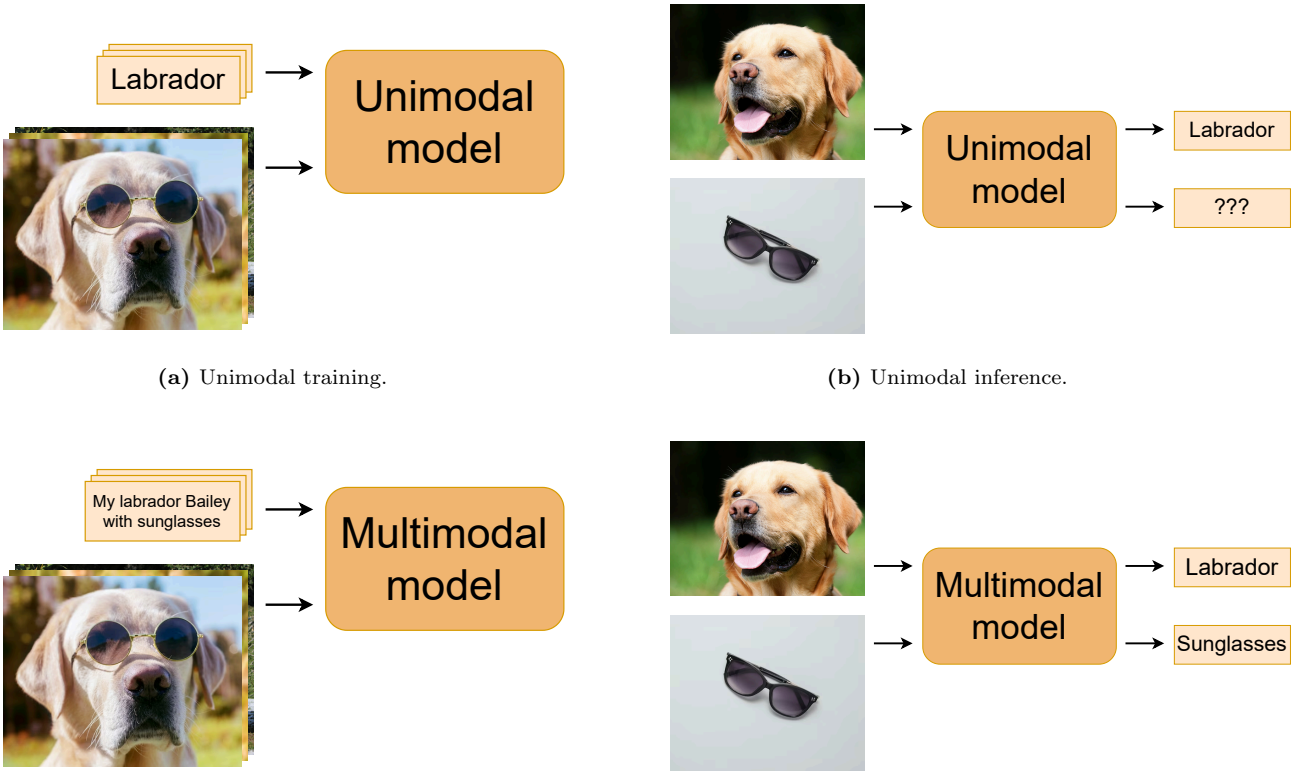
CONTENTS

LIST OF FIGURES

CHAPTER

ONE

INTRODUCTION



(a) Unimodal training.

(b) Unimodal inference.

(c) Multimodal training.

(d) Multimodal inference.

Figure 1.1: A simplified visualization of the training and inference of unimodal and multimodal models for a vision task. Where unimodal models can not generalize the learned visual concepts to new classes, multimodal models can.

Ever since the popularization of Convolutional Neural Networks (CNNs) for image classification by AlexNet [3], they have consistently outperformed other machine vision architectures on benchmarks like the ImageNet image classification task [4]. Over the years, the complexity of these architectures has increased, with each new model adding innovations such as inception modules [5], residual connections [6], and depthwise separable convolutions [7]. Only recently, CNNs were outperformed by transformer architectures such as Vision Transformers (ViTs) on the ImageNet benchmark [8]. Their parallel nature allows these transformer models to train at a larger scale than CNNs. However, the high computational complexity of their attention mechanisms makes inference more time-consuming than for CNNs. Despite their differences, all these architectures have one thing in common: They train in a single modality.

Unimodal vision models only use a single modality, vision, for training. While the labels for each image are textual, they are encoded using a one-hot embedding, resulting in a purely numerical label. This prevents the model from exploiting textual information. For example, the ImageNet classes "`white shark`", "`tiger shark`", and "`hammerhead shark`" are seen as unrelated by a unimodal model, even though their textual description indicates they are very similar.

Another downside of unimodal models is that they can not generalize to classes they have not seen before. Since they lack textual understanding, the learned concepts can not be generalized to a new class. For example,

a model that has learned the classes of "`hammer`" and "`shark`" will not be any better at categorizing the unseen class of "`hammerhead shark`" than a model that is only trained to recognize flowers.

Multimodal models add one or more modalities to the training data. For example, a model trained in the modalities of vision and text will train on an image and a textual description. These descriptions can be much more varied than the single label used in unimodal models. For a visual comparison of the training of unimodal and multimodal models, Figures 1.1a, and 1.1c show how a multimodal model is able to learn the concept of sunglasses from the textual descriptions of the image whereas a unimodal model only associates the image with the class label.

By learning concepts in two modalities, multimodal models can exploit textual relationships between classes and generalize their learned visual concepts to previously unseen classes. For an example of this generalization capability, which is not present in unimodal models, see Figures 1.1b and 1.1d.

This generalization capability of multimodal models is best seen when trained on large amounts of data. Meanwhile, a unimodal model might only be trained on a single dataset, such as ImageNet, while a multimodal model will train on many image-text pairs that are freely available on the Internet and do not need to be annotated. The result is a visual model with high generalization capability. For example, a pre-trained multimodal model can achieve competitive performance on the ImageNet benchmark without training on ImageNet specifically [1].

Multimodal vision models that incorporate textual understanding are known as Vision Language Models (VLMs). Recent VLMs such as CLIP [1] and ALIGN [9] can be trained at a very large scale using millions of noisy image-text pairs, which are widely available on the Internet. These pairs can be easily scraped from websites containing images and captions, such as forums, social media, Wikipedia, and many others. This wide data availability allows VLMs to train at a very large scale. ALIGN even shows that VLMs do not need expensive data preprocessing when trained at a large enough scale [9].

These VLMs use a contrastive loss to learn a feature space that aligns text descriptions with their corresponding images. This allows them to learn a broad range of visual and linguistic concepts and their relationship to one another. These learned concepts across multiple modalities make VLMs promising foundation models that can be applied to various downstream vision tasks, such as image classification, image captioning, and visual question answering [1].

VLMs utilize a pre-trained text encoder, which allows them to generalize learned visual-textual relations to related textual information, giving them a high capability for adapting to various downstream tasks in a zero-shot manner. It also adds a new dimension to vision problems: prompting. For example, in the task of image classification, the embedding of an image is compared to the embeddings of several text descriptions. These descriptions consist of a textual prompt followed by the class label. A prompt might be "`a photo of a [CLASS]`", which is a general description of the image. For inference, the similarity of the image embedding to the embeddings of all prompts is computed, giving a probability distribution over all the classes.

It has been shown that the zero-shot performance of CLIP highly varies depending on the used prompt [1]. It is, therefore, common practice to engineer a prompt for a specific task, either manually or through optimization [10, 11]. For example, a prompt engineered for a specific task might be "`a photo of a [CLASS], a kind of flower`". Prompt tuning allows VLMs to improve their generalization to new domains. However, it requires either engineering by a human domain expert in the case of manual tuning or labeled data from the target domain in the case of prompt optimization. Both are typically unavailable at inference time, which restricts prompt tuning from making on-the-fly optimizations.

Test-Time Adaptation (TTA) can enable prompt tuning as a method of adapting to new domains at inference time. TTA utilizes unsupervised learning techniques to learn from unlabeled data at test time [12]. Therefore, TTA techniques can adapt to new domains at inference time without additional labeling efforts. One method that uses TTA for prompt tuning of VLMs is Test-time Prompt Tuning (TPT) [2]. TPT uses augmentation, confidence selection, and entropy minimization to learn from a single unlabeled input image. Whereas most TTA methods optimize the parameters of a learned network [13, 14, 15], TPT only learns the parameters of a textual prompt.

In this thesis, we take a closer look at the unsupervised learning mechanisms behind TPT. Our contribution will be threefold:

1. We show through ablation that the unsupervised learning methods of TPT sometimes decrease the performance of the method on a fine-grained classification task.

2. We propose two prediction alignment methods to alleviate this problem.

3. We show how our prediction alignment methods compare to standard TPT and how they respond to additional augmentation.

We show that our method, which aligns predictions in TPT using consistency regularization, outperforms standard TPT on most of the evaluated fine-grained classification datasets. It also outperforms the baseline on datasets where standard TPT could not. We further show that our method can learn from additional augmentation while the original implementation cannot.

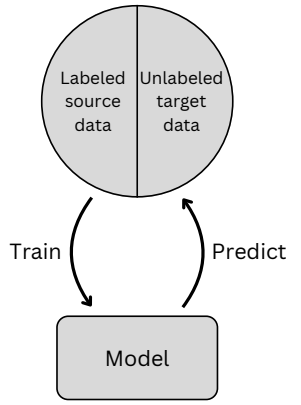
This thesis is organized as follows. First, chapter 2 will discuss the literature on test-time adaptation, vision-language models, and prompt tuning. In chapter 3, we will perform an in-depth analysis of the unsupervised learning mechanisms behind TPT and show through an ablation study how each component of TPT affects the performance on fine-grained classification tasks. Then, chapter 4 introduces our two methods for aligning predictions in TPT and shows how they compare to standard TPT. Chapter 5 investigates the impact of more severe augmentation on standard TPT and our methods. Finally, chapter 6 discusses our results and proposes directions for future work.

CHAPTER
TWO

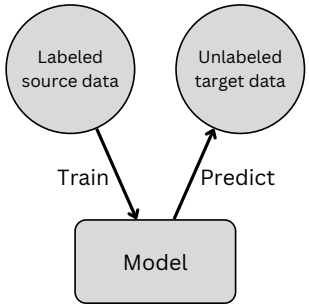
RELATED WORK

This section explores previous research and developments relevant to the focus of this thesis. We identify key findings, methodologies, and gaps that inform and contextualize our study by examining existing literature. This section is organized into three main areas. First, we will introduce the concept of test-time adaptation and give an overview of relevant methods and techniques. Second, we will explore the concept of foundational models. Specifically, we will look at vision-language models which are used in this thesis. Finally, we will link the first two sections by discussing how test-time adaptation can tune the prompts of vision-language models at inference time.

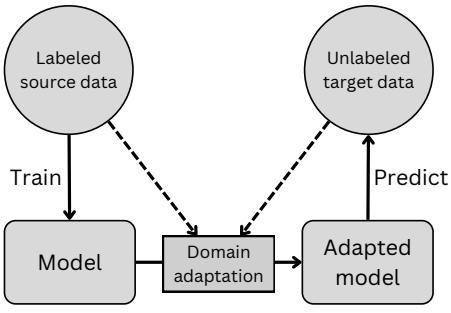
2.1 Test-time adaptation



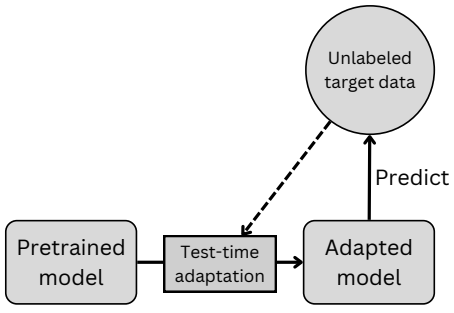
(a) Classical machine learning.



(b) Domain shift.



(c) Domain adaptation.



(d) Test-time adaptation.

Figure 2.1: Visualization of the problem of domain shift and possible solutions. **(a) Classical machine learning**: the source and target data are assumed to be drawn from the same distribution. **(b) Domain shift**: the source and target data come from different distributions. **(c) Domain adaptation**: the labeled source data and unlabeled target data are used to adapt a trained model to the target domain. **(d) Test-time adaptation**: only unlabeled target data is used to adapt a trained model to the target domain.

Classical machine learning assumes that the training and test data are drawn independently and identically from the same distribution (Figure 2.1a). In real-world applications, this assumption usually does not hold. When the training data (source) differs from the test data (target) (Figure 2.1b), the problem of *distribution shift* occurs [16]. Distribution shift is usually paired with severe drops in performance. For example, in segmenting

medical images of brain tumors, a CNN was shown to perform significantly worse on images taken at another institution than the one that produced its training data [17]. In the image classification task for autonomous driving, models might generalize very poorly to data containing objects in unusual poses [18]. These examples all show a performance degradation as a result of a distribution shift.

Domain adaptation (DA) aims to close the performance gap of a discriminative model between the source domain it has trained on, and a new target domain [19]. Typically, DA methods will align some statistics of the two different domains through sampling [20, 21] or feature transformations [22, 23]. Therefore, DA methods need access to both the source data used for training and the target data (Figure 2.1c). Since source data sets can be very large, applying DA during inference time in practical applications is often not feasible. In these cases, test-time adaptation (TTA) can be used [12]. TTA is a source-free variant of DA that does not utilize any data from the source domain (Figure 2.1d). In addition to enabling adaptation during inference time, TTA addresses privacy and data storage concerns by not requiring source data.

This section will explain the difference between TTA and other test-time learning methods. We will then describe variants of TTA techniques that can operate on large amounts of target data, smaller batches, or single instances. After, we will present several popular TTA methods and categorize them based on the unsupervised learning strategies that they use. For a more extensive survey covering this topic, see [12].

2.1.1 Related techniques

Test-time adaptation is not the only technique for improving performance at test time. In this subsection, we will explain the differences and similarities between test-time adaptation, -training, and -augmentation.

Test-time training

Test-time training (TTT) adapts a model at test-time on an individual test sample. In this way, TTT is similar to Test-Time Instance Adaptation (TTIA), which we will discuss in Section 2.1.2. However, the key difference lies in the optimization method. Whereas TTA optimizes over the original classification task using a method such as entropy minimization, TTT trains over an auxiliary self-supervised task. The auxiliary task needs to be explicitly defined. Examples of auxiliary tasks are rotation prediction, where the model predicts the rotation angle of an input image [24], inpainting, where the model tries to reconstruct missing or corrupted parts of an image [25], and contrastive learning, where the model learns to distinguish between similar and dissimilar image patches or augmented versions of the same image [26].

Test-time augmentation

Test-time augmentation, which is sometimes also referred to as TTA, is another method for improving model performance during inference [27]. The key difference with test-time adaptation is that test-time augmentation does not modify the learned model. Instead, it applies various data augmentations (such as flips, rotations, or crops) to each test sample, generates predictions for these augmented versions, and then aggregates them (often through averaging or voting) to produce a final output. Test-time augmentation essentially creates an ensemble effect for each test sample, potentially improving prediction robustness and accuracy. The ability of test-time augmentation to improve performance has been extensively shown by pioneering image classification works such as AlexNet [3], Inception [5], and ResNet [6].

As this thesis will show, data augmentations can also improve the performance of TTA methods. However, since the resulting methods modify the parameters of the learned model, they are not test-time augmentation methods.

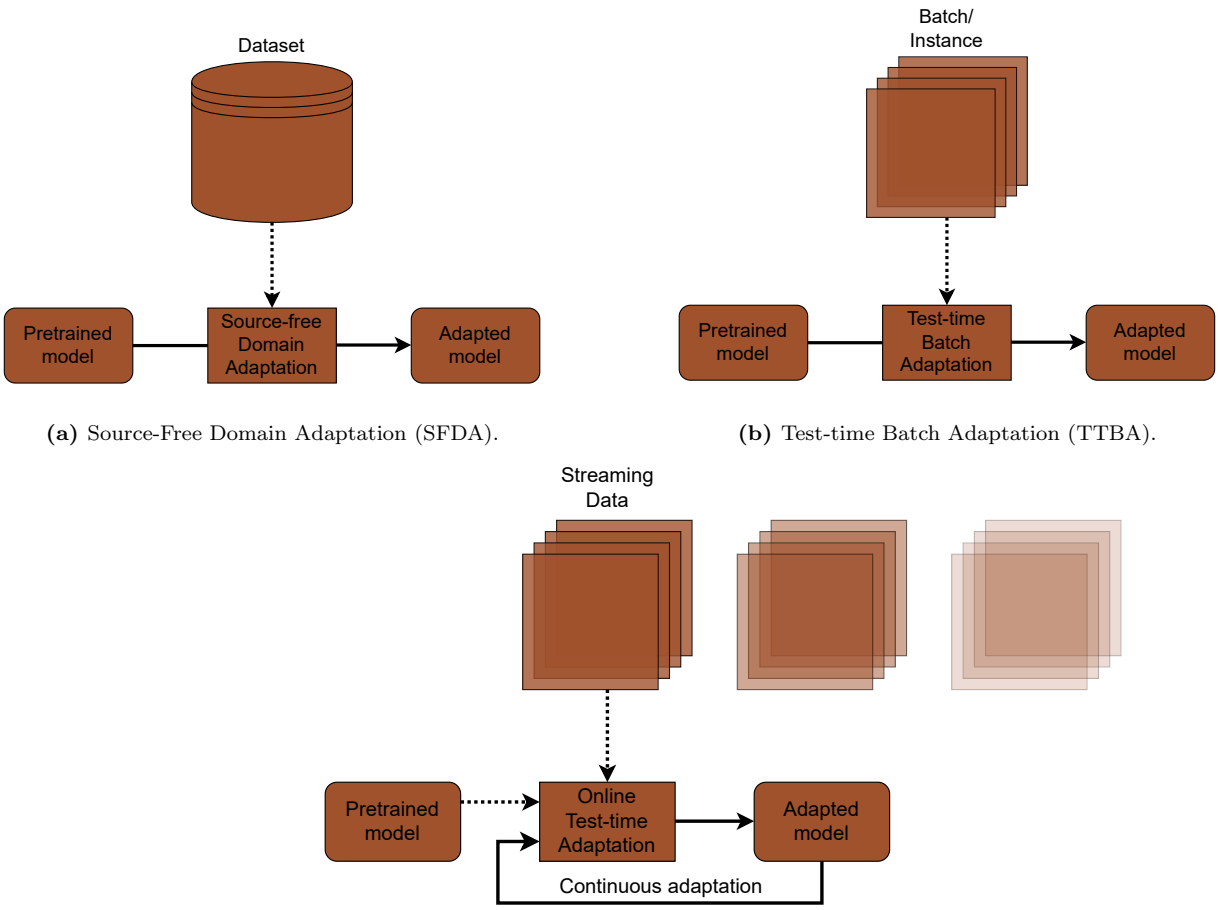
2.1.2 Variants

We can categorize various types of test-time adaptation based on the volume of target data they work with. This section will outline TTA methods that work with complete datasets, individual batches, single instances, and streaming data. To see a visual representation of the different TTA variants, refer to Figure 2.2.

Source-free domain adaptation

TTA methods operating on an entire target dataset can be seen as a source-free domain adaptation (SFDA) variant. For a visualization of the SFDA approach to TTA, see Figure 2.2a. Unlike standard DA methods, TTA methods can not use sampling or feature transformations to align the features of the target domain to those of the source domain since the source domain is unknown. Instead, TTA methods rely on various unsupervised learning approaches.

Since SFDA can adapt to a large amount of target data, it has the potential to perform significant adjustments to the pretrained model. However, training on a large amount of data has a significant computational cost. Since the entire target dataset will need to be collected before adaptation occurs, SFDA methods cannot be applied during inference time.



(a) Source-Free Domain Adaptation (SFDA).

(b) Test-time Batch Adaptation (TTBA).

(c) Online Test-time Adaptation (OTTA).

Figure 2.2: Visualization of different variants of test-time adaptation. **(a) Source-free domain adaptation** adapts a pretrained model using an unlabeled dataset of the target domain. **(b) Test-time batch adaptation** only uses a batch of images or a single instance from the target domain. **(c) Online test-time adaptation** continuously updates a model based on streaming data.

Test-time batch adaptation

In a practical setting, it is unlikely that the entire target domain will be available at test time. Instead, samples will likely arrive one by one or in batches. Test-Time Batch Adaptation (TTBA) focuses on the latter, adapting the model to one batch of samples at a time [12]. For a visualization of the TTBA approach to TTA, see Figure 2.2b. TTBA is well-suited for on-the-fly optimizations at test time and for situations where each batch of samples can come from a different distribution.

With a batch size of 1, TTBA is reduced to Test-Time Instance Adaptation (TTIA). TTIA can be used for streaming data such as a video feed. However, it is important to note that TTIA will treat every frame as coming from an independent distribution and, therefore, will not apply learned knowledge to new frames.

Online test-time adaptation

Whereas TTBA can be used for on-the-fly adaptations at test-time, it requires that data is grouped in batches and will treat each batch as coming from a different distribution. This is not ideal for streaming data received continuously and which may come from the same distribution. Online Test-Time Adaptation (OTTA) reuses past knowledge, which allows for online learning [28]. OTTA methods continuously adapt a model to new input and will use the adapted model as a starting point for the next input. While this allows the model to retain knowledge learned at test time, it also leads to new challenges. For example, a model might forget a learned pattern after having seen many inputs that lack this pattern. This is known as catastrophic forgetting [29, 30].

2.1.3 Optimization methods

Test-time adaptation can use various unsupervised optimization methods to improve performance during test time. This section will discuss methods based on pseudo-labeling, entropy minimization, and consistency regularization. We will name a few TTA techniques for each method that utilize it. For a more extensive overview of TTA techniques for each optimization method, see [12].

Pseudo-labeling

One approach at test time is to use the model trained on the source data to label all instances in the target domain. Under domain shift, these pseudo-labels are likely inaccurate. Therefore, various methods exist to improve the accuracy of the pseudo-labels or filter out inaccurate pseudo-labels. A self-supervised learning approach can then be used, updating the network using supervised training methods and pseudo-labels. This process can be repeated for an iterative self-supervised learning approach.

The pseudo-labeling strategy assumes that most pseudo-labels will be accurate if the domain shift is mild. Therefore, training requires a large number of target samples. This means that the pseudo-labeling optimization strategy is only feasible for SFDA, not TTBA, TTIA, or OTTA.

An example of a pseudo-labeling-based SFDA technique is Source HypOthesis Transfer (SHOT) [13], which generates a prototype representation for each class by taking the centroid of all instances from the target dataset labeled as belonging to that class and then obtains pseudo-labels using the nearest centroid classification. Privacy-Preserving Domain Adaptation (PPDA) [31] proposes improving the class centroids using only highly confident predictions. Similarly, Self-Supervised Noisy Label Learning (SSNLL) [32] performs K-means clustering in the target domain and obtains pseudo-labels by averaging the predictions of samples in the same cluster. Domain-Invariant Parameter Exploring (DIPE) [33] assigns pseudo-labels to each sample based on a majority vote over the label of its nearest neighbors in the target domain.

In contrast to these methods, which assign pseudo-labels based on the distribution of samples in the target domain, some methods assign a pseudo-label by aggregating predictions of an ensemble. For example, Augmented Self-Labeling (ASL) [34] uses an ensemble over different data augmentations and ST3D [35] over predictions of previous training steps.

Several methods combine the above-mentioned labeling strategies to obtain their pseudo-labels [36, 37, 38].

Entropy minimization

Entropy minimization is a popular strategy for all forms of TTA. This method optimizes the model to obtain low entropy and, therefore, high predictive confidence for samples in the target domain. For SFDA, Augmentation-based Source-Free Adaptation (ASFA) [39] minimizes the α-Tsallis entropy. Other works, such as ASL [34], minimize the more traditional Shannon entropy. SS-SFDA [40] chooses to prioritize minimizing the entropy in confident samples over less confident samples.

Similarly to SFDA, many test-time batch adaptation approaches rely on entropy minimization. Notably, Test Entropy Minimization (TENT) updates the model by minimizing the entropy of its predictions for each batch. However, the authors observe that updating all model parameters on a few test samples can lead to overfitting. Therefore, they borrow an idea from batch normalization calibration methods such as Representative BatchNorm (RBA) [41]. Instead of updating all the network parameters, TENT only learns the parameters of a linear transformation of the features at each batch normalization layer. The transformation parameters can be maintained for an online approach (OTTA) or forgotten for an offline (TTBA) approach. By adapting very few parameters of the original model (2 per batch normalization layer), TENT not only reduces overfitting but also mitigates the problem of catastrophic forgetting when used in an online learning setting.

Many TTA works build on the simple approach of TENT. Namely, Marginal Entropy Minimization with One test point (MEMO) [15] extends TENT to optimize the network over multiple random augmentations of the target batch. Test-Time Adaptation with Shape moments for image segmentation (TTAS) [42] adds a class-weighted entropy objective and Variational Model Perturbation (VMP) [43] adapts TENT to a probabilistic method by perturbating the model parameters with variational Bayesian inference.

Consistency regularization

Another strategy for TTA is consistency training, which aims to enforce consistent predictions under data or model variations. For SFDA, Robust Self Training (RuST) [44] optimizes over multiple permutations at the input, feature, and model levels. Feature Alignment by Uncertainty and Self-Training (FAUST) [45] takes a different approach, minimizing the epistemic model uncertainty estimated using Monte Carlo dropout [46].

Test-Time Adaptation through Perturbation Robustness (TTA-PR) [47] extends the entropy minimization approach of TENT by adding a consistency regularization term in the loss. For consistency regularization, they use the average KL divergence over all probability distributions with respect to the average probability distribution. Since the model trains over multiple augmentations of the original image, it utilizes the knowledge that all samples belong to the same class. The loss penalizes differences between probability distributions, therefore encouraging the model to give consistent predictions over augmentations of the same image. This consistency loss was first introduced for use with AugMix [48].

2.2 Foundational models

Foundation models are those trained on large-scale datasets (generally using self-supervision) covering many different domains [49]. The knowledge of these models can be transferred to downstream tasks using a task-specific dataset (transfer learning) [50], a few examples (few-shot learning) [51], or even without any examples (zero-shot learning) [52].

2.2.1 The introduction of scalable architectures

To achieve the generalization necessary for successful transfer learning across many domains, foundation models require training on very large datasets. However, most traditional deep learning architectures struggle when training on a large scale. For example, simple multi-layer perceptrons (MLPs) are very dense with many parameters. This makes it infeasible to train an MLP of sufficient width and depth to learn patterns in large datasets covering many domains.

Convolutional neural networks

Convolutional neural networks (CNNs) are less dense than their MLP counterparts because the convolutions share common parameters (filters) [53]. However, very large CNNs can suffer problems with vanishing or exploding gradients, leading to unstable training. These problems are mostly mitigated by modern CNN architectures featuring batch normalization [54], residual connections [6], and compound scaling [55]. Nevertheless, CNNs rely on their local receptive fields and pooling operations to capture spatial information hierarchically. This limits their generalization ability to local regions of the input, making them unfit for learning long-range dependencies in the data.

Recurrent neural networks

Recurrent neural networks (RNNs) are an architecture tailored to sequential data processing [56]. They maintain an internal memory to capture temporal dependencies in the data. While theoretically, their recurrent nature allows RNNs to capture long-range dependencies, in reality, the vanishing gradients problem may limit this ability. Training RNNs involves the iterative processing of sequential data over multiple time steps, involving matrix operations and backpropagation through time. These operations become increasingly demanding with longer sequences, larger batch sizes, and deeper architectures, leading to higher computational costs and longer training times. This computation complexity and the memory requirements for storing the hidden states over multiple timesteps make training RNNs at a very large scale infeasible.

The attention mechanism was introduced to mitigate the problem of long-range dependencies for RNNs [57]. Attention allows models to focus on specific parts of the input sequence when generating each part of the output sequence, which addresses the limitations of traditional sequence-to-sequence models. This mechanism dynamically assigns weights to different input tokens, enabling the model to prioritize more relevant information.

Transformers

Whereas RNNs with attention mechanisms are better at detecting long-range dependencies, they still suffer from the same computational complexity as RNNs, which makes it challenging to train them at a very large scale. The key innovation allowing for truly scalable training was the introduction of the transformer architecture [58]. This novel approach replaced recurrent neural networks with an architecture based entirely on attention mechanisms, allowing for more efficient parallel processing of input sequences. By eliminating the need for sequential processing, transformers can leverage modern GPU hardware for faster training and inference, making them highly scalable for large datasets and complex tasks. This scalability, combined with their ability to learn contextual relationships in data, has led to the development of increasingly powerful language models and their application across a wide range of domains, including machine translation [58], text generation [59, 60], and even computer vision [8].

2.2.2 Vision-language models

Vision-language models (VLMs) have recently been intensely investigated [61]. These new foundational models utilize both language and vision architectures to learn from the large number of noisy image-text pairs widely available on the internet. VLMs such as Contrastive Language-Image Pre-training (CLIP) [1] show that given large enough datasets, this weak supervision can provide zero-shot generalization capabilities. A Large-scale ImaGe and Noisy-text embedding (ALIGN) [9] shows that VLMs can even learn from very noisy data without expensive filtering or preprocessing as long as the datasets are sufficiently large.

CLIP is trained on a large dataset of image-text pairs using a contrastive learning approach [1]. CLIP processes batches of image-text pairs during training, encoding images and texts separately through pre-trained encoders. It then computes the cosine similarity between all possible image-text combinations, aiming to

maximize the similarity for matching pairs while minimizing it for non-matching pairs. This process creates a shared embedding space where semantically related images and texts are mapped close together. This training procedure can be seen in Figure 2.3 (1). For prediction, CLIP can perform zero-shot classification by encoding a given image and a set of candidate text descriptions (usually a prompt followed by the class label) into this shared embedding space (Figure 2.3 (2)). It then computes the similarity between the image embedding and each text embedding, selecting the class with the highest similarity as the predicted label (Figure 2.3 (3)).

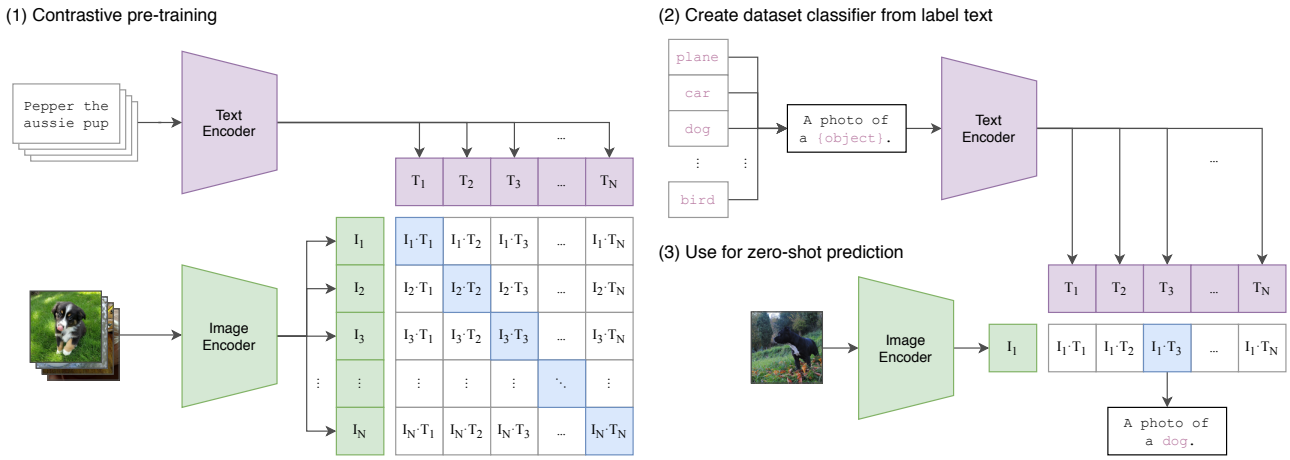


Figure 2.3: CLIP jointly trains an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples. At test time, the learned text encoder synthesizes a zero-shot linear classifier by embedding the names or descriptions of the target dataset's classes. Image and description taken from [1].

2.3 Prompt tuning for vision-language models

VLMs such as CLIP [1] and ALIGN [9] perform classification by comparing the embedding of the image to the embeddings of several textual prompts. An example of such a prompt is: "`a photo of a [CLASS]`", where `[CLASS]` is replaced by a class label. By contextualizing the class label into a description of the image, prompting allows VLMs to be easily transferred to downstream tasks such as classification and context-dependent visual reasoning [2]. The prompt used for any downstream task may significantly impact the model's zero-shot performance. For example, the authors of Context Optimization (CoOp) [10] show that small changes such as adding an article before a noun, changing the sentence structure, or adding a single word describing the target domain can significantly impact the model performance.

Naturally, it can be quite effective to manually change the prompt to find one which gives sufficient performance [62]. This manual tuning process optimizes the *hard prompts*, which is the textual prompt before embedding it with the text encoder. Optimizing the hard prompt is known as prompt engineering. Prompt engineering requires human involvement from a domain expert and is, therefore, extremely time-consuming. Since the hard prompts are discrete, they are difficult to optimize with automated techniques. One solution is to optimize the *soft prompts*, which are the embeddings of the prompts after passing through the text encoder. Since these are continuous, they can be optimized with standard optimization methods. This process is known as prompt tuning. This section will look at methods for few-shot prompt tuning and test-time prompt tuning.

2.3.1 Few-shot prompt tuning

CoOp [10] uses a few annotated samples from the target domain to tune the soft prompt. This method optimizes a context vector that serves as a task-specific prompt, allowing the model to adapt to various visual recognition tasks with minimal data. The authors show that with as few as one or two samples, CoOp can outperform manually engineered prompts. Conditional Context Optimization (CoCoOp) [11] extends this concept by making the prompt conditional on the input image, enabling more flexible and context-aware adaptations. Both methods have shown promising results in few-shot learning scenarios and out-of-domain generalization tasks, outperforming traditional fine-tuning approaches in many cases. These techniques leverage the strong pre-trained representations of CLIP while allowing for efficient adaptation to specific tasks, making them particularly useful in scenarios with limited labeled data.

2.3.2 Test-time prompt tuning

As we saw in Section 2.1, Test-Time Adaptation (TTA) techniques can optimize a model at test-time based on unlabeled data from a new target domain. TTA methods can also be used to optimize the soft prompts of a VLM at test time for a new domain without labeled data. This method is known as Test-time Prompt Tuning

(TPT) [2], a test-time instance adaptation method that uses an entropy minimization strategy to optimize the prompt over a single target image. TPT consists of a few essential components: augmentation, confidence selection, entropy minimization, and a learnable prompt. In this section, we will describe these components. See Figure 2.4 for a visualization of TPT.

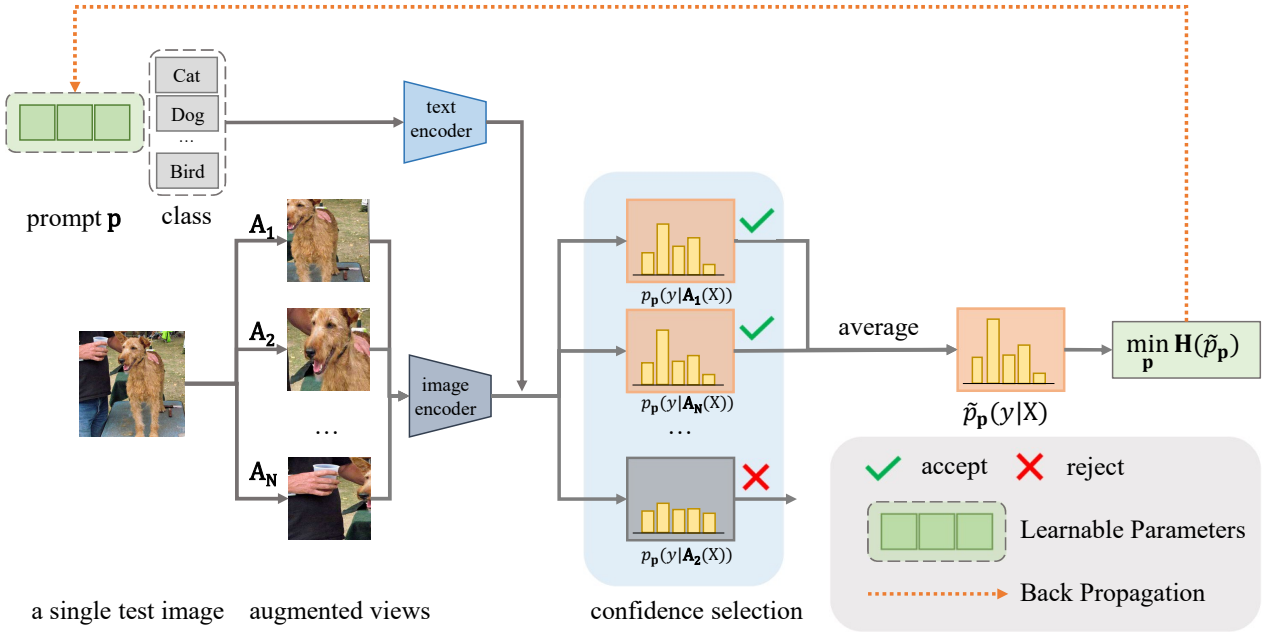


Figure 2.4: Test-time Prompt Tuning (TPT) for image classification. Prompts are tuned on the fly with a single test sample without additional training data or annotations. TPT optimizes the prompt to encourage consistent predictions across augmented views by minimizing the marginal entropy. We introduce *confidence selection* to filter out noisy augmentations. Image and description taken from [2].

Augmentation

TPT works on a single test sample, augmented using standard crop and flip operations into many views. It then uses a CLIP [1] model to encode these views and the text prompts describing the classes. The similarity between the augmented image and text embeddings is then computed, resulting in a probability distribution for each augmented view, containing the probabilities of the image belonging to each class, according to the CLIP model.

Confidence selection

The probability distributions with the highest entropy are filtered out, leaving only the more confident distributions. The rationale is that the augmentations introduce noise, possibly leading to uncertain predictions. For example, in Figure 2.4, the image's subject is a dog. However, if an augmented view crops the image so that the subject is no longer in it, this can lead to a prediction with low confidence and, therefore, high entropy.

Entropy minimization

The confident probability distributions are averaged to obtain a single distribution. Like TENT [14], TPT uses entropy minimization to optimize the model performance. It is important to note that this method encourages the network to be more confident in the predictions that we have selected, even if those predictions are wrong. Unlike TENT, TPT uses data augmentation to optimize multiple views of the same image. This approach is similar to that of MEMO [15].

Learnable prompt

Another difference between TPT and TENT/MEMO is the optimized parameters. TENT learns two parameters per batch normalization layer that serve as a linear transformation. On the other hand, TPT does not update any parameters of the CLIP model. Instead, it optimizes a soft prompt for use with the CLIP model, similar to the approach of CoOp [10]. The parameters of this learnable prompt are visualized in green in Figure 2.4. However, where CoOp uses few-shot learning to optimize the prompt using a few labeled samples from the target domain, TPT only uses a single unlabeled sample.

CHAPTER

THREE

TEST-TIME PROMPT TUNING FOR FINE-GRAINED CLASSIFICATION

As we saw in Section 2.3.2, Test-time Prompt Tuning (TPT) [2] can be used to adapt a model to a new domain at test time with only a single unlabeled sample from the new domain. In their paper, the authors demonstrate the method's ability to adapt to natural distribution shifts by evaluating four ImageNet [4] variation datasets, as is done for CLIP [1]. The variation datasets are:

- **ImageNet-V2** [63]: an independent test set of 1000 classes using data collected in the same way as the original ImageNet a decade later.
- **ImageNet-A** [64]: a set of 1000 classes containing samples that were misclassified by a standard ResNet-50 [6], and are therefore considered challenging.
- **ImageNet-R** [65]: a set of 1000 classes using artistic renditions of ImageNet classes, such as sketches, cartoons, tattoos, and graffiti.
- **ImageNet-Sketch** [66]: a set of 1000 classes using grayscale sketches of ImageNet classes.

The authors of TPT demonstrate that on these datasets, TPT can outperform a zero-shot CLIP baseline, a CLIP model with a hand-crafted ensemble of prompts, and even the few-shot methods CoOp [10] and CoCoOp [11]. They also show that they can combine TPT and CoOp by starting from a soft prompt learned by CoOp on a few samples from the original ImageNet dataset and then optimizing it using TPT. This approach leads to a higher top-1 accuracy on both ImageNet and the ImageNet variation datasets than TPT and CoOp individually.

This impressive result shows that TPT can efficiently generalize to a broad classification dataset like ImageNet. ImageNet is considered broad because it contains a wide variety of classes that are often distinct and easily distinguishable from one another. For instance, ImageNet contains thousands of categories ranging from animals to everyday objects. The primary challenge in such datasets is correctly identifying the broad category to which an image belongs.

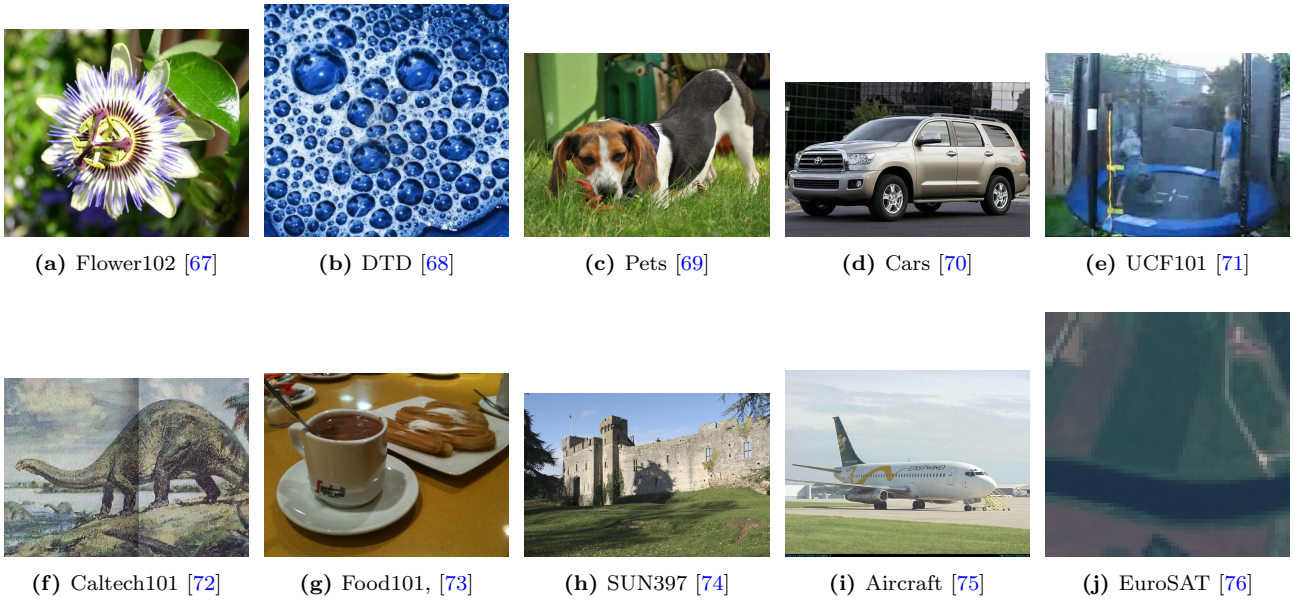
In this thesis, we investigate the ability of TPT to generalize to new domains. Therefore, we examine its generalization to fine-grained classification datasets. In contrast to broad datasets, these fine-grained datasets require distinguishing classes that are much more similar. For example, a fine-grained dataset might consist of many different breeds of dogs or species of flowers. The subtle visual differences between these classes make classification tasks more challenging than broad classifications, as they require the model to learn and identify small variations between classes, such as slight variations in fur patterns or flower petal shapes.

This chapter will first discuss the fine-grained classification datasets used to evaluate TPT. Then, we will discuss how the various components of TPT affect its performance on these fine-grained classification sets. This leads us to introduce an ablation study of the various components of TPT to explore what role each part plays in the final performance of the model. We will show the results of this experiment and discuss our findings.

3.1 Fine-grained classification datasets

In the paper, TPT is evaluated on ten fine-grained image classification datasets [2]. These 10 datasets are the same ones used to evaluate CoOp [10] and a subset of the 27 datasets used to evaluate CLIP [1]. Each of the datasets contains images from a large number of similar classes from a specific domain. The fine-grained datasets are:

- **Flower102** [67]: 102 classes of flowers common in the UK.
- **DTD** [68]: 47 classes of describable textures.
- **OxfordPets** [69]: 37 classes of breeds of cats and dogs.



(a) Flower102 [67] **(b)** DTD [68] **(c)** Pets [69] **(d)** Cars [70] **(e)** UCF101 [71]

(f) Caltech101 [72] **(g)** Food101, [73] **(h)** SUN397 [74] **(i)** Aircraft [75] **(j)** EuroSAT [76]

Figure 3.1: Sample images of the fine-grained classification datasets used in the evaluation of test-time prompt tuning.

- **OxfordCars** [70]: 197 classes of car models.
- **UCF101** [71]: 101 classes of human actions.
- **Caltech101** [72]: 101 classes of objects.
- **Food101** [73]: 101 classes of types of food.
- **SUN397** [74]: 397 classes of scene recognition.
- **Aircraft** [75]: 100 classes of aircraft models.
- **EuroSAT** [76]: 10 classes of types of land cover on Sentinel-2 satellite images in Europe.

In the TPT paper, they show that TPT with a ViT-B/16 backbone outperforms a baseline of CLIP with the standard prompt "`a photo of a [CLASS]`" on 9 out of 10 fine-grained datasets [2]. We reproduce this experiment in Table 3.1. We show a similar result, with TPT outperforming the baseline on 8 out of 10 fine-grained datasets. Our slightly different result can be due to the random noise in creating the data augmentations.

Table 3.1: Zero-shot generalization on fine-grained classification datasets. Comparison of zero-shot CLIP baseline with TPT, showing results from the TPT paper [2] in comparison to our reproduction of these results. We report the top-1 classification accuracy on each dataset.

Method	Flower102	DTD	Pets	Cars	UCF101	Caltech101	Food101	SUN397	Aircraft	EuroSAT	Average
Baseline from [2]	67.44	44.27	**88.25**	65.48	65.13	93.35	83.65	62.59	23.67	42.01	63.58
TPT from [2]	**68.98**	**47.75**	87.79	**66.87**	**68.04**	**94.16**	**84.67**	65.5	**24.78**	**42.44**	**65.10**
Baseline (ours)	67.36	44.39	**88.20**	65.61	65.13	93.35	83.65	62.61	**23.67**	42.16	63.61
TPT (ours)	**69.55**	**46.45**	87.30	**66.86**	**68.83**	**93.59**	**85.04**	**65.14**	23.34	**43.53**	**64.96**

3.2 Augmentation and confidence selection

In section 2.3.2, we described the four main components of TPT: augmentation, confidence selection, entropy minimization, and a learnable prompt. The main innovations of TPT are:

1. Applying entropy minimization to tune soft prompts of vision-language models.
2. Using augmentation and confidence selection to learn from a single image.

Whereas the first innovation is well motivated by literature such as TENT [14] and CoOp [10], the second is not explicitly motivated in the paper. While the authors do show that TPT performs best on ImageNet variation datasets when selecting the 10% most confident samples as opposed to more or less, they do not show this for the fine-grained datasets or perform an ablation study showing that the augmentation and confidence selection improves the performance of TPT.

In this chapter, we set up an experiment to evaluate whether the augmentation and confidence selection strategies are effective in TPT. We set out to test three hypotheses:

1. If augmentation and confidence selection are effective strategies to improve the performance of a zero-shot classifier, then we can apply these methods to the CLIP baseline for a performance boost.
2. If confidence selection helps filter out noisy augmentations, we expect TPT to perform worse without confidence selection.
3. If data augmentation helps learn from a single image, we expect TPT to perform worse without augmentation.

The following sections will describe the experiments used to evaluate these hypotheses, show their results, and discuss the outcome.

3.3 Experimental setup

For this experiment, we used the original code for TPT[1]. We use a pre-trained CLIP-ViT-B/16 with the standard prompt "`a photo of a [CLASS]`" for the baseline. For TPT, we use the same CLIP model, start with the standard prompt, and optimize the corresponding four tokens (one for each word) in the text input embedding space. For every image, we create 63 augmentations using flipping and cropping. We evaluate on the 63 augmentations and the original image, resulting in 64 different views of a single image. The top 10% most confident samples are selected by taking the 6 with the lowest self-entropy. We minimize the marginal entropy over these confident samples for 1 step by using the AdamW optimizer [77] and a learning rate of 0.005, as in the TPT paper.

For evaluation, we use the test set of each fine-grained classification dataset. For every image in the test set, TPT performs one optimization step and then performs inference on this image using standard CLIP with the optimized prompt. We report the average top-1 accuracy score over the test set of each dataset. Since there are many datasets, we also compute the average top-1 accuracy over all datasets. In this average, the performance on each dataset is counted equally, regardless of the dataset size.

3.3.1 Baseline with augmentation and confidence selection

To test our first hypothesis, we create an implementation of the baseline CLIP model which uses augmentation and confidence selection to make a prediction. Given a single image, this implementation will use augmentation to obtain 64 views like TPT. Then, it will use the same confidence selection to obtain the 6 most confident probability distributions. These 6 distributions are averaged to obtain a single distribution. Instead of using this distribution to minimize the entropy, as in TPT, it is used immediately to make a prediction for the original image.

3.3.2 TPT without confidence selection

To test our second hypothesis, we create an implementation of TPT without confidence selection. Whereas TPT selects the 10% samples with the lowest self-entropy, we will use all samples to obtain an average probability distribution. Therefore, this implementation will effectively train on 10 times more data than the original TPT implementation. Since an increase in data quantity is usually associated with an increase in performance, we compensate for this by creating an additional implementation of TPT without confidence selection, which only creates 6 augmentations. This implementation, therefore, trains on the same number of augmentations as the original TPT method.

3.3.3 TPT without augmentation

To test our third hypothesis, we create an implementation of TPT without augmentation. Given a single image, this implementation uses CLIP to obtain a probability distribution for this image. Then, entropy minimization is used directly on this distribution. Since there is only one view of the original image, confidence selection is not used. By eliminating augmentation and using only entropy minimization, we effectively reduce TPT to a form of TENT [14]. However, whereas TENT optimizes parameters in the model's batch normalization layers, this implementation optimizes a soft prompt used for CLIP.

[1] Available from https://azshue.github.io/TPT/

3.4 Experiment results

The first experiment's results can be seen in Table 3.2. Note that the baseline without augmentation and confidence selection is the standard zero-shot CLIP baseline, as in Table 3.1. We can see that the baseline CLIP model with added augmentation and confidence selection performs better on some datasets than the standard baseline and worse on others. On average, this implementation performs slightly worse than the standard baseline.

Table 3.2: Zero-shot generalization on fine-grained classification datasets. Showing the impact of adding augmentation and confidence selection to the baseline. We report the top-1 classification accuracy on each dataset.

Method	Augmentation & confidence selection	Flower102	DTD	Pets	Cars	UCF101	Caltech101	Food101	SUN397	Aircraft	EuroSAT	Average
Baseline	×	**67.36**	44.39	**88.20**	65.61	65.13	**93.35**	83.65	62.61	23.67	**42.16**	**63.61**
Baseline	✓	67.19	**45.69**	86.75	**67.75**	**67.62**	93.23	**83.99**	**64.21**	**24.81**	34.48	63.57

The results of the second experiment can be seen in Table 3.3. Note that the TPT with confidence selection and batch size 64 is the standard TPT method. We can see that the TPT variant without confidence selection performs better than standard TPT on 7 out of 10 fine-grained datasets. It performs slightly better on average and achieves the best average top-1 accuracy in this experiment.

Table 3.3: Zero-shot generalization on fine-grained classification datasets. Showing the impact of removing confidence selection from TPT. We report the top-1 classification accuracy on each dataset.

Method	Confidence Selection	Batch size	Flower102	DTD	Pets	Cars	UCF101	Caltech101	Food101	SUN397	Aircraft	EuroSAT	Average
Baseline	×	1	67.36	44.39	88.20	65.61	65.13	93.35	83.65	62.61	23.67	42.16	63.61
TPT	✓	64	69.55	**46.45**	87.30	66.86	**68.83**	93.59	85.04	65.14	23.34	**43.53**	64.96
	×	64	**70.60**	45.80	**88.33**	**67.41**	68.28	**93.75**	**85.23**	**65.26**	23.70	41.53	**64.99**
	×	6	70.40	46.28	88.14	66.25	68.36	93.71	84.60	64.80	**23.85**	42.04	64.84

The use of more data could explain this increase in performance. We see that the implementation of TPT without confidence selection and a batch size of 6 performs worse on most datasets than both other implementations. Its average top-1 accuracy is also the lowest in this experiment. Interestingly, it does outperform the original TPT implementation on 4 out of 10 datasets.

Table 3.4: Zero-shot generalization on fine-grained classification datasets. Showing the impact of removing augmentation from TPT. We report the top-1 classification accuracy on each dataset.

Method	Augmentation	Flower102	DTD	Pets	Cars	UCF101	Caltech101	Food101	SUN397	Aircraft	EuroSAT	Average
Baseline	×	67.36	44.39	88.20	65.61	65.13	93.35	83.65	62.61	**23.67**	42.16	63.61
TPT	✓	**69.55**	**46.45**	87.30	**66.86**	**68.83**	**93.59**	**85.04**	**65.14**	23.34	43.53	**64.96**
	×	69.02	45.15	**88.23**	65.08	67.30	93.39	83.95	63.63	22.14	**47.10**	64.50

The results of the third experiment can be seen in Table 3.4. The implementation without any augmentation performs worse than standard TPT on 8 out of 10 fine-grained datasets. On average, it performs worse than both TPT and the baseline. Interestingly, this implementation achieves the best performance on EuroSAT, improving several percentage points over both the baseline and TPT.

3.5 Discussion

In this chapter, we examine the performance of TPT on fine-grained classification datasets. In Section 3.2, we presented three hypotheses to test whether the augmentation and confidence selection strategy is effective in TPT. For the first, we make a baseline CLIP classifier that uses augmentation, confidence selection, and the average probability distribution over confident samples to make a prediction. If augmentation and confidence selection are effective strategies to improve the performance of a zero-shot classifier, we expect this modified baseline model to perform better than the standard baseline. However, our experiment showed that it performs slightly worse on the classification datasets. While the difference is small, it is clear that adding augmentation and confidence selection to the baseline does not improve performance as we expected.

For the second hypothesis, we created a TPT implementation that does not use confidence selection but rather averages the probability distributions of all augmented views. Interestingly, this implementation performed slightly better than standard TPT on the fine-grained datasets. When we compensate for the extra data used by this implementation with a batch size of 6, the performance decreases to below that of both the baseline and standard TPT. TPT seems to perform worse without confidence selection, as we expected. However, the variant without confidence selection and with a compensated batch size does outperform standard TPT on 4

out of 10 datasets. Therefore, we can not draw clear conclusions on whether confidence selection in TPT is effective or not.

The effectiveness of the confidence selection could be dependent on the application. For example, one possibility is that for an application where the objects span a large part of the input image, we do not expect cropping to add a lot of noise to the classification task. In these applications, confidence selection should, therefore, be less effective. However, when we look at the results for DTD and EuroSAT (both tasks where the item to be classified spans the entire image), we see that this hypothesis does not hold. TPT with confidence selection performs better than the variant without confidence selection for both datasets.

Conversely, we would expect confidence selection to be effective for datasets where the classification depends on small details in the image since it can filter out augmentations where those small details are cropped out. For datasets with small details, we can look at Cars and Aircraft. In these datasets, a small detail, such as a vehicle logo, a differing number of windows, or a different wing shape, can be crucial to correctly classifying the image. However, when we look at TPT's performance, we see that on both datasets, the variants without confidence selection perform better.

Thus, while we expect confidence selection to be effective on tasks with small distinguishing features and less effective on tasks with large distinguishing features, we do not see this in practice. One explanation could be that the network makes overconfident predictions, even when incorrect. This is known as the problem of overconfidence, and it has been shown to impact modern image classifiers specifically [78]. Recent work has shown that overconfidence also occurs in multi-modal models such as CLIP [79]. In our case, the network's overconfidence could explain why confidence selection does not always perform as expected since incorrect predictions with high confidence can add a lot of noise to the training procedure. If this is the case, confidence calibration could help to alleviate this problem [78].

For the third hypothesis, we created a TPT implementation that does not use any augmentation or confidence selection. This implementation performs entropy minimization directly on the probability distribution obtained by passing the original image through the CLIP model. We expect that this implementation will perform worse than TPT with augmentation. Indeed, we have seen that this is the case. However, it performs better than the baseline and performs better than standard TPT on 2 out of 10 datasets. This result once again shows that the current method of augmentation, confidence selection, and entropy minimization is quite sensitive to the application, performing best on most datasets but not on all of them.

CHAPTER
FOUR

ENFORCING CONSISTENCY IN TEST-TIME PROMPT TUNING

In chapter 3, we saw that the current test-time prompt tuning method of augmentation, confidence selection, and entropy minimization does not consistently achieve the expected results. We hypothesized that over-confident model predictions could explain this. In this chapter, we will look at another possible explanation for the unstable performance of the TPT method: misaligned predictions. Specifically, we will explain what happens when the model tries to minimize the entropy over several differing predictions.

To alleviate this problem, we propose two methods for enforcing consistent predictions. The first will focus on a majority voting scheme to optimize the model using only predictions of the majority class. The second will introduce a consistency regularization term in the loss function to encourage the model to predict the same class for all augmentations of the same image.

This chapter will first describe the problem of misaligned predictions. Then, we will introduce our two methods for ensuring consistency in TPT. We will describe an experimental setup for comparing the prediction alignment methods to standard TPT, show the experiment's results, and discuss our findings.

4.1 Misaligned predictions

In section 2.3.2, we described the process of augmentation, confidence selection, and entropy minimization followed by TPT. A crucial step is averaging the selected probability distributions to obtain a single distribution. Entropy minimization is used to encourage the model to be more confident about the predictions in which it was confident and less confident about the predictions in which it was not. However, what happens when the selected distributions are not in agreement with the predicted class? In this case, averaging their probability distributions could lead to unwanted results.

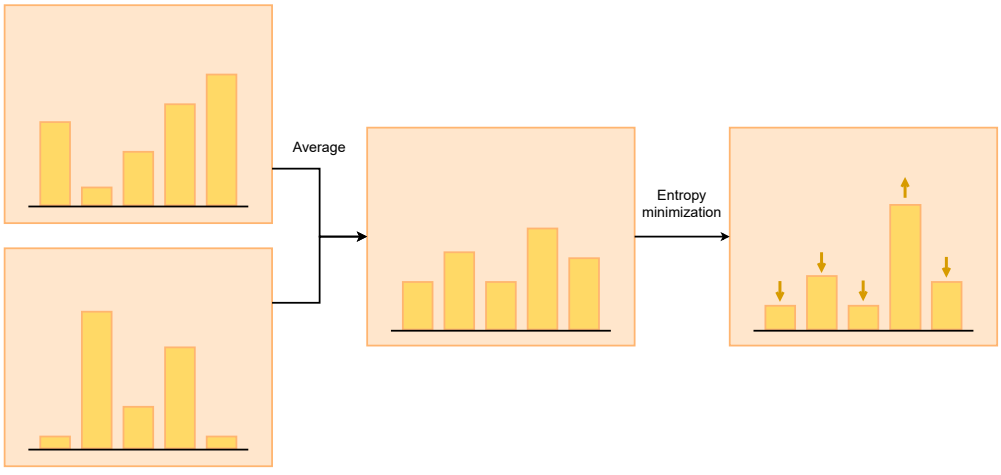


Figure 4.1: Two disagreeing probability distributions are averaged into a single distribution. The obtained average looks very different than both original distributions. Entropy minimization is applied to the average.

As a simple example, take the case where two probability distributions are in disagreement with each other. This example is illustrated in Figure 4.1. The first distribution is most confident in the fifth class, whereas the second distribution is most confident in the second class. If we average these distributions, we see that the resulting distribution looks very different. Not only does it have a higher entropy, but class four now has the highest confidence. If entropy minimization is applied, the model will be encouraged to become more confident in class 4 and less confident in the others, as this minimizes the entropy. This example illustrates how the

optimization performed on an average distribution might not reflect the original distributions' prediction if they disagree.

The important question is, therefore, does this phenomenon occur in test-time prompt tuning? We ran a simple test: TPT was run as normal, the six most confident augmentations were selected out of 64, and we counted all the times when these were not in agreement. As a result, we saw that over all the fine-grained classification datasets used in this thesis, the confident samples disagreed in 33.70% of cases. See Table 4.1 for the exact percentage of disagreements per dataset. Not surprisingly, the disagreement was generally higher for datasets in which TPT achieved a lower accuracy and vice versa. We do not see a clear link between the disagreement and the number of classes in the dataset.

Table 4.1: Percentage of times that the distributions selected by TPT were in disagreement with one another for each fine-grained dataset.

Dataset	Flower102	DTD	Pets	Cars	UCF101	Caltech101	Food101	SUN397	Aircraft	EuroSAT
Disagreement	30.17%	41.25%	14.61%	42.56%	34.42%	8.92%	20.13%	35.33%	80.20%	59.58%
TPT performance	69.55	46.45	87.30	66.86	68.83	93.59	85.04	65.14	23.34	43.53
Number of classes	102	47	37	197	101	101	101	397	100	10

We have seen that misaligned predictions could potentially be disruptive to the learning process of TPT. We have also seen that the predictions for all confident samples are often in disagreement when inferencing on fine-grained classification datasets. In the next section, we will introduce two techniques for aligning TPT predictions to test whether this impacts its performance.

4.2 Consistency techniques

This section will introduce two techniques for enforcing prediction consistency in TPT. The first will focus on ensuring the selected probability distributions are consistent with one another and, therefore, only alter the confidence selection phase. The second technique will instead alter the loss function to encourage consistent predictions across different augmentations of the same image.

4.2.1 Majority voting

For the first technique, we will align the selected predictions by altering the confidence selection to only select samples with identical predicted classes. We, therefore, need a method to decide which class is most likely to be the correct class based on the augmentations of the original image and their associated probability distributions. We choose to implement a majority voting scheme. We use hard voting. For each augmentation, we compute the predicted class as the arg max over its probability distribution. Then, we count which class is predicted most often.

Thus, given k views of the original image $x_1, \ldots, x_k$ and the associated probability distributions $p_{x_1}, \ldots, p_{x_k}$, the majority class can be determined as follows:

$$C_{\text{majority}} = \arg\max_{c} \sum_{i=1}^{k} \mathbf{1}\left(\arg\max_{c'} p_{x_i}(c') = c\right) \tag{4.1}$$

where $\mathbf{1}(\cdot)$ is the indicator function that equals 1 if the condition inside is true and 0 otherwise.

Confidence selection is then applied as in standard TPT, with the added restriction that only samples predicted to belong to the majority class can be selected. With this technique, it will not be possible for selected samples to be in disagreement with one another.

4.2.2 Kullback-Leibler divergence

For the second technique, we will not alter the confidence selection. Instead, we will encourage the model to produce similar probability distributions for different augmentations of the same image. We will do this using the concept of consistency regularization [80, 81]. The idea behind consistency regularization is to ensure that the model's predictions remain stable under small perturbations of the input data.

We obtain several augmented views of the original image in TPT and predict their classes. TPT uses unlabeled data, so we do not know the ground-truth class. However, we can use the underlying knowledge that the class for all augmented views is the same since they all come from the same augmented image. We can thus use consistency regularization over all augmented views to ensure the predicted probability distributions are similar.

We need a distance measure to enforce similar probability distributions from one predicted probability distribution to the next. We can use the Kullback-Leibler (KL) divergence [82]. For two distributions p and q

of size k, the KL-divergence can be defined as follows:

$$D_{KL}(p||q) = \sum_k p^k \log(\frac{p_k}{q_k}) \tag{4.2}$$

The divergence is higher for dissimilar distributions and lower for similar distributions.

For TPT, we can create a KL loss, which penalizes dissimilarity between the probability distributions of all augmented views of the original image. For k views of the original image $x_1, \ldots, x_k$ and the associated probability distributions $p_{x_1}, \ldots, p_{x_k}$, the KL loss can be computed as follows:

$$\mathcal{L}_{KL}(p_{x_1}, \ldots, p_{x_k}) = \frac{1}{k} \sum_k D_{KL}(p_{x_k}||\bar{p}) \tag{4.3}$$

Where $\bar{p}$ is the average distribution:

$$\bar{p} = \frac{1}{k} \sum_k p_{x_k} \tag{4.4}$$

This loss function was first proposed in TTA-PR [47]. They show that the addition of a KL loss can improve the performance of TENT [14] on a corrupted version of a broad classification dataset: CIFAR-100-C [83].

This KL loss can be used as the only loss function in TPT, but it can also be added to the existing entropy minimization loss $\mathcal{L}_{ent}$, possibly with a scaling parameter α to control the importance of the KL term:

$$\mathcal{L}(p_{x_1}, \ldots, p_{x_k}) = \mathcal{L}_{ent}(CS(p_{x_1}, \ldots, p_{x_k})) + \alpha \mathcal{L}_{KL}(p_{x_1}, \ldots, p_{x_k}) \tag{4.5}$$

Where $CS(p_{x_1}, \ldots, p_{x_k})$ represents the confidence selection. An important distinction is that the entropy loss is computed only over the selected augmented views, while the KL loss is computed over all augmented views.

Whereas the majority voting technique guarantees that selected samples in TPT will never be in disagreement with one another, the KL-divergence technique does not. While it does encourage the model to make consistent predictions, it is not guaranteed that the predictions will be consistent.

4.3 Experimental setup

We set up an experiment to compare the two proposed methods for aligning predictions with each other, standard TPT and the CLIP baseline. We use a pre-trained CLIP-ViT-B/16 with the standard prompt "`a photo of a [CLASS]`" for the baseline. For TPT, we use the same CLIP model, start with the standard prompt, and optimize the corresponding four tokens (one for each word) in the text input embedding space. For every image, we create 63 augmentations using flipping and cropping. We evaluate on the 63 augmentations and the original image, resulting in 64 different views of a single image. The top 10% most confident samples are selected by taking the 6 with the lowest self-entropy. We minimize the marginal entropy over these confident samples for 1 step by using the AdamW optimizer [77] and a learning rate of 0.005, as in the TPT paper.

For evaluation, we use the test set of each fine-grained classification dataset. For every image in the test set, TPT performs one optimization step and then performs inference on this image using standard CLIP with the optimized prompt. We report the average top-1 accuracy score over the test set of each dataset. Since there are many datasets, we also compute the average top-1 accuracy over all datasets. In this average, the performance on each dataset is counted equally, regardless of the dataset size.

4.3.1 Majority voting

Our majority voting implementation adds an additional step to the confidence selection. After obtaining the probability distribution from every augmented view, it computes the predicted class for each view with an argmax over the associated distribution. It then computes the majority class as described in equation 4.1. In the confidence selection, the distributions are filtered to only contain those with their predicted class equal to the majority class. From these, the most confident samples are selected.

The same number of samples are selected as without majority voting. Thus, when selecting the top 10% of confident samples, this means 10% of the augmented views, not 10% of the samples predicted to belong to the majority class. In the rare case where the samples predicted as the majority class are less than 10% of the augmented views, only those will be selected. Thus, in this case, less samples will be used.

4.3.2 Kullback-Leibler divergence

For our KL divergence method, we implement the KL loss as given in equation 4.3. To evaluate the impact of a KL divergence loss on TPT, we test two implementations. The first implementation will only use the KL loss to update the prompt in TPT. Thus, no entropy minimization is used. The second implementation will combine the two losses as in equation 4.5. Based on preliminary testing, we use $\alpha = 0.1$, giving more importance to the entropy minimization term than the KL term. This value could be optimized with more extensive parameter tuning.

4.4 Experiment results

The result of the experiment can be seen in Table 4.2. Firstly, we can see that the majority voting technique performs worse overall than standard TPT without any alignment of the predictions. Interestingly, it does outperform standard TPT on four out of ten fine-grained datasets. On the DTD and Caltech101 datasets, it also outperforms the other alignment methods.

Table 4.2: Zero-shot generalization on fine-grained classification datasets. Comparison of TPT with different methods for enforcing consistency between predictions. We report the top-1 classification accuracy on each dataset.

Method	Alignment method	Flower102	DTD	Pets	Cars	UCF101	Caltech101	Food101	SUN397	Aircraft	EuroSAT	Average
Baseline	None	67.36	44.39	**88.20**	65.61	65.13	93.35	83.65	62.61	23.67	42.16	63.61
TPT	None	69.55	46.45	87.30	66.86	68.83	93.59	85.04	65.14	23.34	43.53	64.96
	Majority voting	69.31	**46.51**	87.90	66.91	68.17	**93.83**	84.84	65.07	23.31	39.46	64.53
	KL only	66.63	44.50	85.66	66.16	66.30	92.25	83.34	63.68	22.53	37.53	62.86
	Entropy + KL	**69.68**	46.28	87.16	**67.02**	**68.90**	93.45	**85.06**	**65.22**	**23.72**	**43.57**	**65.01**

Secondly, we can see that when we only use the KL loss to train TPT, its performance degrades significantly. In fact, with only the KL loss, TPT performs worse overall than the zero-shot CLIP baseline. The performance of this method is worse on all datasets than standard TPT.

Thirdly, we see that our method, which combines an entropy minimization and KL divergence loss, achieves the best overall performance in this experiment. This method achieves the best performance of all the above-mentioned methods on seven out of ten fine-grained image classification datasets. The average performance over all datasets is slightly higher than that of standard TPT.

4.5 Discussion

In this chapter, we showed that the TPT performance could be potentially unstable due to misaligned predictions. We introduced two methods for aligning the predictions: majority voting and KL divergence. We compared these methods to each other with an experiment, standard TPT, and a zero-shot CLIP baseline.

Our results showed that the majority voting technique neither provides more stable performance across datasets nor achieves better performance on average. This can mean one of two things. Firstly, it could mean that the misalignment of predicted classes does not hinder the learning ability of TPT. In that case, filtering out samples with misaligned predictions means that samples with lower confidence will be selected. This could explain why majority filtering degrades the performance of TPT.

Secondly, our result could mean that merely filtering out any non-majority-class samples is not enough to stabilize the performance of TPT. While it does guarantee that all selected samples predict the same class, that class could be wrong. Thus, while majority voting does align the predictions, it eliminates prediction diversity, which could degrade performance when a wrong class is selected as the majority class. In that case, a method that encourages similar predictions without restricting the selection of confident samples (such as KL divergence) could perform better.

When we use TPT with only KL divergence as its loss, we see that the performance is severely degraded. In fact, the performance of this method is worse than standard TPT on every fine-grained dataset. This shows that the KL loss alone is not sufficient to learn a better prompt from a single image. This could be because the model abuses the prompt parameters to achieve consistent predictions. For example, when optimizing a prompt using consistency regularization on multiple augmentations of a single image, the prompt could be manipulated to hint at a single class without accounting for the visual information in the image. This would mean the model would make consistent predictions which are likely to be wrong.

When we combine KL divergence with TPT's standard entropy minimization, we can see an increase in performance. This method outperforms standard TPT on seven out of ten fine-grained datasets. It is important to note that the differences in performance are small, and this method does not increase performance on all datasets. However, from our results, we can say that this method does show promise in improving upon TPT. We hypothesize that the increase in performance is because the KL divergence loss reduces the problem of misaligned predictions. However, we can not say this for certain.

Since TPT does not utilize any ground-truth labels, it is a form of unsupervised learning. Other unsupervised learning methods have been shown to benefit from the use of KL divergence. For example, it is commonly used in Variational Autoencoders (VAEs) [84] to regularize the latent space. KL divergence can also be used to train Generative Adversarial Networks (GANs) [85, 86].

The combination of entropy minimization and KL divergence is also common. For example, in reinforcement learning, Soft-Actor Critic (SAC) [87] uses both entropy minimization and KL divergence to balance exploration and exploitation. Entropy minimization and KL divergence can also be used in the training of Bayesian Networks (BNs) [88].

CHAPTER
FIVE

AUGMENTATION METHODS FOR TEST-TIME PROMPT TUNING

So far in this thesis, we have examined the behavior of Test-time Prompt Tuning (TPT) in chapter 3 and proposed two solutions to the problem of misaligned predictions in chapter 4. One crucial aspect of TPT, data augmentation, has remained constant. In this chapter, we will look at the augmentation in more detail. Our first goal will be to examine standard TPT behavior under different degrees of augmentation. Secondly, we will do the same for TPT with our proposed alignment methods. On the one hand, we will want to see how little augmentation these methods need to learn and, on the other hand, whether more severe augmentations can increase their performance.

We will start by describing the augmentation method used in TPT. Then, we will introduce a more severe form of augmentation. We will describe an experiment to test the impact of more severe augmentation on standard TPT and TPT with the alignment methods. Lastly, we will describe the results of this experiment and discuss our findings.

5.1 Augmentation methods

Standard TPT uses cropping and flipping as augmentation methods. However, for more severe augmentation, we can use AugMix [48]. This section will describe both methods and their differences.

5.1.1 Standard augmentation in TPT

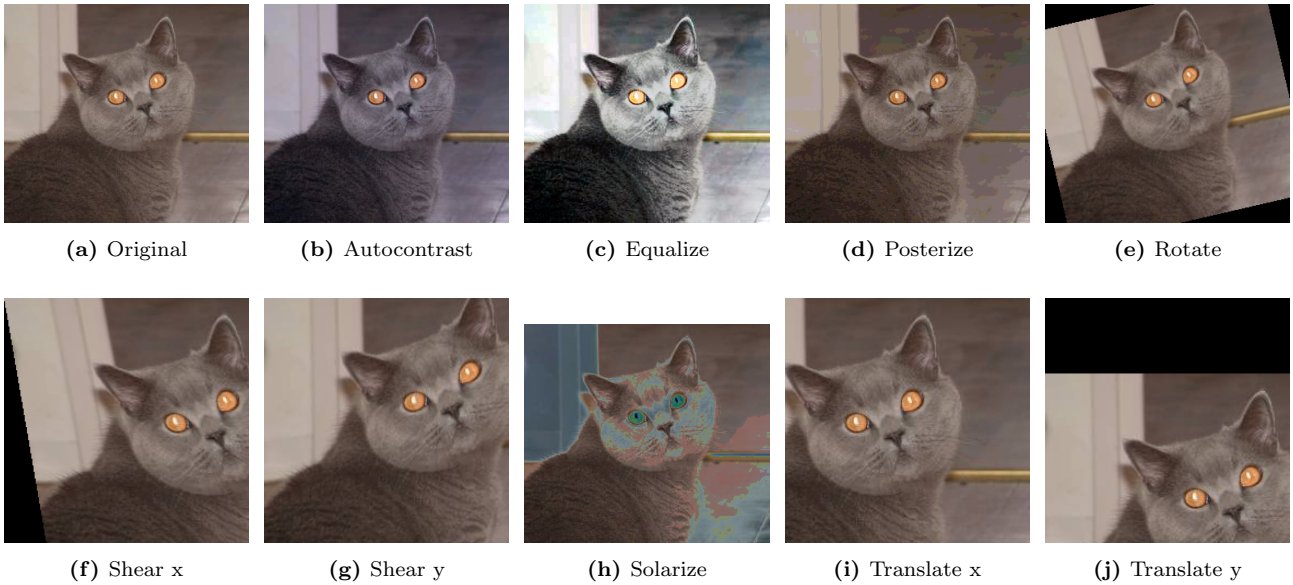
(a) Original **(b)** Random crop **(c)** Flip

Figure 5.1: Example of the standard augmentations used in TPT.

For the standard augmentation used in TPT, the original image is augmented 63 times using a random crop and a random flip. For the crop, a random area of the image is chosen, which must be at least 8% of the area of the entire image. The aspect ratio may differ up to 25% from the original image. The random crop is then resized to the resolution of the original image using bilinear interpolation. Lastly, the image is randomly flipped horizontally with a probability of 50%.

5.1.2 More severe augmentation with AugMix

For a more severe augmentation of the original image, we can use AugMix [48]. AugMix was introduced as an augmentation technique to improve image classifiers' robustness and uncertainty estimates. It consists of nine



(a) Original **(b)** Autocontrast **(c)** Equalize **(d)** Posterize **(e)** Rotate

(f) Shear x **(g)** Shear y **(h)** Solarize **(i)** Translate x **(j)** Translate y

Figure 5.2: Example of the augmentations that can be applied with AugMix.

different augmentations: autocontrast, equalize, posterize, rotate, shear x, shear y, solarize, translate x, and translate y, which are combined and mixed. The individual augmentations can be seen in Figure 5.2.

Augmix takes the original image and creates three augmented variants. A random number (between one and three inclusive) of augmentations from Figure 5.2 is applied to each augmented variant. These three variants are combined into a single augmentation using weights sampled from a Dirichlet distribution. Finally, the combined augmentation and the original image are mixed through a random convex combination sampled from a Beta distribution.

5.2 Experimental setup

We set up an experiment to evaluate the impact of standard augmentation and more severe augmentation with AugMix on the performance of standard TPT and TPT with the two alignment methods. For the method of KL divergence, we use the combination of entropy minimization and KL divergence because it performed much better than KL divergence alone. To better understand the impact of augmentations, we also test the impact of using less or more augmented views of the original image. We will use 8, 16, 32, 64, and 128 for the number of augmented views.

We use a pre-trained CLIP-ViT-B/16 with the standard prompt "`a photo of a [CLASS]`" for the baseline. We use the same CLIP model for TPT, starting with the standard prompt and optimizing the corresponding four tokens (one for each word) in the text input embedding space. The augmentations are created using either standard flipping and cropping or flipping and cropping in addition to AugMix. We evaluate on the augmentations and the original image. For example, we will train on the original image and seven augmentations when using eight augmented views. The top 10% most confident samples are selected by taking the ones with the lowest self-entropy. For example, for 64 augmented views, the six most confident samples are selected. We minimize the marginal entropy over these confident samples for 1 step using the AdamW optimizer [77] and a learning rate of 0.005, as in the TPT paper.

For evaluation, we use the test set of each fine-grained classification dataset. For every image in the test set, TPT performs one optimization step and then performs inference on this image using standard CLIP with the optimized prompt. We report the average top-1 accuracy score over the test set of each dataset. Since there are many datasets, we also compute the average top-1 accuracy over all datasets. In this average, the performance on each dataset is counted equally, regardless of the dataset size.

5.3 Experiment results

The result of the experiment can be seen in Table 5.1. Note that TPT with standard augmentation and 64 augmented views is the standard TPT variant. Since this table contains a considerable amount of data, we have visualized the average performance over all fine-grained datasets in Figure 5.3.

In the figure, we can see that for standard TPT, the performance increases as the number of augmentation increases. However, more augmented views above 64 do not seem to improve the performance. When we look at TPT with AugMix, we see that the same holds. When we compare it to TPT with standard augmentation

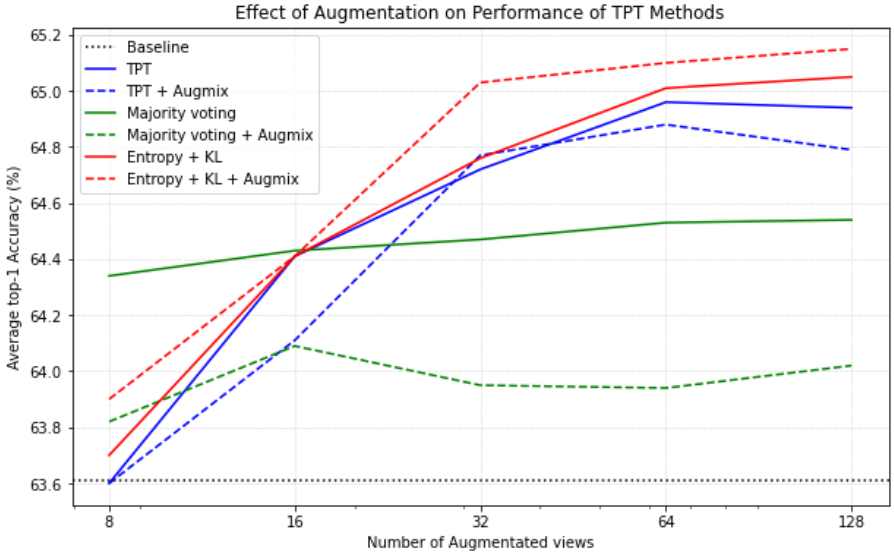


Figure 5.3: Comparison of standard TPT and TPT with majority voting or KL divergence for different numbers of augmented views and with or without AugMix. We report the average top-1 classification accuracy over all fine-grained datasets.

we see that the addition of AugMix does not notably increase performance. In fact, the performance seems to have slightly decreased for all augmented views except 8 and 32.

For TPT with majority voting, the results look quite different. With very few augmented views, it performs better than standard TPT. However, this technique does not gain much performance when adding more augmented views. For more than 16 views, this technique performs worse than standard TPT. Adding AugMix to the majority voting technique severely degrades performance, regardless of the number of augmented views used.

TPT with an added KL divergence term in the loss performs much more similarly to standard TPT compared to majority voting. For most numbers of augmented views, it achieves slightly higher performance. Notably, where standard TPT does not manage to improve its performance when using more than 64 augmented views, TPT with KL divergence achieves a slight improvement. We see that adding AugMix increases TPT's performance with KL divergence for almost all augmented views.

5.4 Discussion

In this chapter, we investigated the impact of data augmentation on TPT. Specifically, we set out to examine whether our proposed alignment methods from chapter 4 respond differently to more severe augmentation than standard TPT. We examined how the performance is affected by using a different number of augmented views and by adding more severe augmentations using AugMix.

We have seen that standard TPT benefits from more augmented views up to a certain point. However, it does not benefit from more views above 64 or more severe augmentation with AugMix. This result could mean that with severe augmentation, too much of the original information is lost, which prevents TPT from learning. However, it could also mean that standard TPT does not fully exploit the information supplied by the augmentations.

In chapter 4, we introduced our methods for aligning predictions using majority voting and KL divergence. Our method with majority voting performs well when using eight augmented views but is outperformed by the other method and standard TPT when using more augmented views or more severe augmentation.

We showed that the KL divergence method achieved slightly better results than standard TPT on the fine-grained datasets. This chapter showed that this method can also better utilize the augmentations than standard TPT. TPT with KL divergence sees increased performance for up to 128 augmented views. Most notably, this is the only method that shows a performance increase when using more severe augmentations with AugMix.

Our method with an added KL divergence term in the loss demonstrates that there is more to be learned from more severe augmentations. It seems that standard TPT is not able to utilize this extra information. Previous work on data augmentation for image classification has shown that more complex models can take advantage of more complex augmentations [89, 6]. However, for our task, both TPT methods rely on the same CLIP network. KL divergence is also used as a consistency regularization method to ensure consistent predictions over multiple versions of the same input [80, 81]. This could explain why our method takes better advantage of augmentation than standard TPT.

When we compare the various TPT methods to the zero-shot baseline, we see that they only need eight augmented views to perform on par with or outperform the baseline zero-shot CLIP method. With so few

Table 5.1: Zero-shot generalization on fine-grained classification datasets. Comparison of different augmentation methods for several variants of TPT. We report the top-1 classification accuracy on each dataset.

Method	Augmentation	no. of views	Flower102	DTD	Pets	Cars	UCF101	Caltech101	Food101	SUN397	Aircraft	EuroSAT	Average
Baseline	×	1	67.36	44.39	88.20	65.61	65.13	93.35	83.65	62.61	23.67	42.16	63.61
TPT	Standard	8	67.44	44.33	**88.23**	65.56	65.08	93.27	83.66	62.63	23.73	42.07	63.60
	Standard	16	68.13	45.57	86.37	66.56	68.12	93.43	84.61	64.51	22.38	**44.47**	64.41
	Standard	32	69.06	45.80	87.14	66.43	68.62	93.59	85.01	64.91	23.07	43.54	64.72
	Standard	64	**69.55**	46.45	87.30	**66.86**	68.83	93.59	**85.04**	65.14	23.34	43.53	**64.96**
	Standard	128	69.31	46.10	87.16	**66.86**	**68.97**	93.67	85.03	65.17	23.70	43.38	64.94
	AugMix	8	67.44	44.33	**88.23**	65.56	65.08	93.27	83.66	62.63	23.73	42.07	63.60
	Augmix	16	67.68	45.51	86.59	65.73	67.20	93.75	84.13	64.69	22.29	43.56	64.11
	AugMix	32	69.06	46.63	87.16	66.76	68.04	93.71	84.62	65.22	23.22	43.22	64.77
	AugMix	64	68.90	**47.40**	87.35	66.42	67.75	94.12	84.69	65.45	**23.76**	42.96	64.88
	AugMix	128	69.14	47.28	87.19	66.67	68.25	**94.16**	84.70	**65.53**	22.74	42.27	64.79
Majority voting	Standard	8	69.02	46.28	87.33	66.01	67.83	93.63	84.57	64.69	23.52	**40.54**	64.34
	Standard	16	69.14	45.98	87.68	66.63	68.15	93.67	84.70	64.95	23.16	40.22	64.43
	Standard	32	**69.43**	45.86	87.87	66.53	67.94	93.59	84.66	65.12	23.97	39.78	64.47
	Standard	64	69.31	**46.51**	87.90	**66.91**	**68.17**	93.83	**84.84**	65.07	23.31	39.46	64.53
	Standard	128	69.22	46.10	**87.95**	66.89	68.07	93.75	84.78	65.17	**23.94**	39.52	**64.54**
	Augmix	8	68.17	45.51	86.64	65.25	66.03	**94.04**	83.96	64.81	23.25	**40.54**	63.82
	Augmix	16	68.66	46.22	87.16	65.97	66.75	93.79	84.12	64.80	23.55	39.91	64.09
	Augmix	32	68.57	46.28	87.24	66.02	66.40	93.59	84.04	65.09	23.19	39.05	63.95
	Augmix	64	68.98	46.16	87.11	66.38	65.95	93.75	84.08	65.30	23.28	38.38	63.94
	Augmix	128	69.10	45.86	87.14	66.16	66.53	93.75	84.06	**65.38**	23.55	38.63	64.02
Entropy + KL	Standard	8	67.54	44.38	**88.30**	65.55	65.17	93.57	83.66	62.83	**23.80**	42.20	63.70
	Standard	16	68.13	45.57	86.37	66.56	68.12	93.43	84.61	64.51	22.38	44.47	64.41
	Standard	32	69.06	45.92	87.16	66.48	68.60	93.63	85.04	64.98	23.10	43.58	64.76
	Standard	64	69.68	46.28	87.16	67.02	68.90	93.45	85.06	65.22	23.72	43.57	65.01
	Standard	128	69.58	46.22	87.00	**67.39**	68.97	93.59	**85.07**	65.32	23.58	43.66	65.05
	Augmix	8	68.44	44.33	88.23	65.56	66.08	93.27	83.66	62.63	23.73	43.07	63.90
	Augmix	16	68.68	45.51	86.59	65.73	68.20	93.75	84.13	64.69	22.29	**44.56**	64.41
	Augmix	32	70.10	46.57	87.05	66.65	68.96	93.67	84.62	65.25	23.28	44.10	65.03
	Augmix	64	69.78	**47.58**	87.19	66.35	68.70	94.04	84.61	65.50	23.58	43.65	65.10
	Augmix	128	**70.18**	47.40	87.14	66.65	**69.20**	**94.16**	84.74	**65.56**	22.92	43.61	**65.15**

augmentations, only a single augmentation will be selected in the confidence selection. This result indicates, therefore, that all of our TPT methods can learn from a single image. With so few augmented views, the majority voting method performs better than the others. However, this method is outperformed by the other methods when using more views. Since the cost of additional augmentation is small, this method will consistently outperform the other two in a practical setting.

CHAPTER
SIX

CONCLUSIONS

This thesis took an in-depth look at Test-time Prompt Tuning (TPT), a method that uses Test-Time Adaptation (TTA) for prompt tuning of Vision-Language Models (VLMs) at inference time. We started by investigating TPT's augmentation and confidence selection mechanism, specifically focusing on its performance on ten fine-grained classification datasets. Our research revealed that the confidence selection used in TPT can be omitted without compromising its performance, and in some cases, TPT can even achieve better performance without the confidence selection. Depending on the application, the performance of TPT can even be higher without the confidence selection. These findings lead us to conclude that the current TPT method's performance is not consistent and is heavily dependent on the specific classification task at hand.

We offer two explanations for this. First, the confidence selection might be affected by overconfident predictions. TPT could benefit from confidence calibration on the underlying CLIP network if this is the case. We leave the investigation of this hypothesis for future work.

The second explanation for the behavior of confidence selection in TPT is that it might be affected by misaligned predictions. In short, when samples that predict different classes for their augmented images are selected, the resulting averaged probability distribution might not reflect the network's confidence in each of the predicted classes. When entropy minimization is performed on this average distribution, unwanted results could be achieved.

We proposed two methods to tackle the problem of misaligned predictions in TPT. Our first method eliminates the problem of misaligned predictions altogether by restricting the confidence selection to only select samples with the same predicted class. It uses a majority voting scheme to determine what class to choose. We found that this method does not improve the performance of TPT and might, in some cases, decrease it. We hypothesize that this is because, by omitting non-majority-class predictions from the confidence selection, samples with lower confidence are selected, which can degrade the performance. Future work could look at other methods of using majority voting which also account for the confidences of each sample.

Our second technique for aligning TPT's predictions uses consistency regularization to encourage consistent predictions across all augmented views of the same image. Specifically, a KL divergence term is added to the loss function of TPT. We show that this method slightly increases performance on most fine-grained datasets over standard TPT.

To further examine the behavior of TPT with our suggested alignment method, we investigate the impact of adding more or less augmented views of the original image and adding more severe augmentation with additional augmentation operations.. We show that our method of TPT with KL divergence can achieve a performance increase from using more augmented views and more severe augmentation with AuxMix, whereas standard TPT cannot. Thus, when using more augmentation, our method is able to achieve a larger performance increase over standard TPT than when using the standard augmentation.

We argue that our method alleviates some of the problems introduced by the confidence selection method in TPT. This leads our method to be less sensitive to the classification task. Where TPT outperforms the zero-shot CLIP baseline on all fine-grained datasets except Pets and Aircraft, our method can outperform the baseline for these datasets when sufficient augmentation is added.

Whether our method performs better is uncertain because it reduces the misalignment problem or because consistency regularization adds extra learning capability. We suggest that future work examines other consistency regularization methods in TPT. The current method trains under variations in the data through augmentations. Another approach would be to train for consistent predictions under model variations. For example, FAUST [45] is a TTA method that performs consistency regularization in the form of epistemic uncertainty estimated by MC dropout [46].

The model could also be jointly trained for consistency under data and model variations. One such example is the mean teacher framework [90], which mixes data augmentations with minimizing the difference between the student and teacher models.

Future work could also look at more augmentation methods. Where the current method applies augmentations to the image input, augmentations could also be made to the textual prompt. Alternatively, augmentation methods could be used that adapt to the input. For example, AugMax [91] learns an adversarial mixture of augmentation operations. Similarly, RandAugment [92] learns a distortion magnitude that controls the severity of the augmentation operations.

The main challenge for these adaptive methods would be to learn a good augmentation from a single input image. For that, these methods could take inspiration from previous work utilizing test-time augmentation [3, 5, 6].

BIBLIOGRAPHY

[1] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*, pp. 8748–8763, PMLR, 2021.

[2] M. Shu, W. Nie, D.-A. Huang, Z. Yu, T. Goldstein, A. Anandkumar, and C. Xiao, "Test-time prompt tuning for zero-shot generalization in vision-language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 14274–14289, 2022.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, 2012.

[4] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1–9, 2015.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

[7] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1251–1258, 2017.

[8] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2020.

[9] C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. Le, Y.-H. Sung, Z. Li, and T. Duerig, "Scaling up visual and vision-language representation learning with noisy text supervision," in *International Conference on Machine Learning*, pp. 4904–4916, PMLR, 2021.

[10] K. Zhou, J. Yang, C. C. Loy, and Z. Liu, "Learning to prompt for vision-language models," *International Journal of Computer Vision*, vol. 130, no. 9, pp. 2337–2348, 2022.

[11] K. Zhou, J. Yang, C. C. Loy, and Z. Liu, "Conditional prompt learning for vision-language models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16816–16825, 2022.

[12] J. Liang, R. He, and T. Tan, "A comprehensive survey on test-time adaptation under distribution shifts," *International Journal of Computer Vision*, pp. 1–34, 2024.

[13] J. Liang, D. Hu, Y. Wang, R. He, and J. Feng, "Source data-absent unsupervised domain adaptation through hypothesis transfer and labeling transfer," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 11, pp. 8602–8617, 2021.

[14] D. Wang, E. Shelhamer, S. Liu, B. Olshausen, and T. Darrell, "Tent: Fully test-time adaptation by entropy minimization," in *International Conference on Learning Representations*, 2020.

[15] M. Zhang, S. Levine, and C. Finn, "Memo: Test time robustness via adaptation and augmentation," *Advances in Neural Information Processing Systems*, vol. 35, pp. 38629–38642, 2022.

[16] O. Wiles, S. Gowal, F. Stimberg, S.-A. Rebuffi, I. Ktena, K. D. Dvijotham, and A. T. Cemgil, "A fine-grained analysis on distribution shift," in *International Conference on Learning Representations*, 2021.

[17] E. A. AlBadawy, A. Saha, and M. A. Mazurowski, "Deep learning for segmentation of brain tumors: Impact of cross-institutional training and testing," *Medical physics*, vol. 45, no. 3, pp. 1150–1158, 2018.

[18] M. A. Alcorn, Q. Li, Z. Gong, C. Wang, L. Mai, W.-S. Ku, and A. Nguyen, "Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4845–4854, 2019.

[19] W. M. Kouw and M. Loog, "A review of domain adaptation without target labels," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 3, pp. 766–785, 2019.

[20] Z. Lipton, Y.-X. Wang, and A. Smola, "Detecting and correcting for label shift with black box predictors," in *International Conference on Machine Learning*, pp. 3122–3130, PMLR, 2018.

[21] K. Azizzadenesheli, A. Liu, F. Yang, and A. Anandkumar, "Regularized learning for domain adaptation under label shifts," in *International Conference on Learning Representations*, 2018.

[22] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars, "Unsupervised visual domain adaptation using subspace alignment," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2960–2967, 2013.

[23] R. Flamary, N. Courty, D. Tuia, and A. Rakotomamonjy, "Optimal transport for domain adaptation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 1, no. 1-40, p. 2, 2016.

[24] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *International Conference on Learning Representations*, 2018.

[25] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2536–2544, 2016.

[26] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International Conference on Machine Learning*, pp. 1597–1607, PMLR, 2020.

[27] D. Shanmugam, D. Blalock, G. Balakrishnan, and J. Guttag, "Better aggregation in test-time augmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1214–1223, 2021.

[28] S. C. Hoi, D. Sahoo, J. Lu, and P. Zhao, "Online learning: A comprehensive survey," *Neurocomputing*, vol. 459, pp. 249–289, 2021.

[29] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," in *Psychology of learning and motivation*, vol. 24, pp. 109–165, Elsevier, 1989.

[30] Q. Wang, O. Fink, L. Van Gool, and D. Dai, "Continual test-time domain adaptation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7201–7211, 2022.

[31] Y. Kim, D. Cho, and S. Hong, "Towards privacy-preserving domain adaptation," *IEEE Signal Processing Letters*, vol. 27, pp. 1675–1679, 2020.

[32] W. Chen, L. Lin, S. Yang, D. Xie, S. Pu, and Y. Zhuang, "Self-supervised noisy label learning for source-free unsupervised domain adaptation," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10185–10192, IEEE, 2022.

[33] F. Wang, Z. Han, Y. Gong, and Y. Yin, "Exploring domain-invariant parameters for source free domain adaptation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7151–7160, 2022.

[34] H. Yan, Y. Guo, and C. Yang, "Augmented self-labeling for source-free unsupervised domain adaptation," in *NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2021.

[35] J. Yang, S. Shi, Z. Wang, H. Li, and X. Qi, "St3d: Self-training for unsupervised domain adaptation on 3d object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10368–10378, 2021.

[36] Z. Cao, Z. Li, X. Guo, and G. Wang, "Towards cross-environment human activity recognition based on radar without source data," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 11, pp. 11843–11854, 2021.

[37] L. Xiong, M. Ye, D. Zhang, Y. Gan, and Y. Liu, "Source data-free domain adaptation for a faster r-cnn," *Pattern Recognition*, vol. 124, p. 108436, 2022.

[38] H.-W. Yeh, T. Westfechtel, J.-B. Huang, and T. Harada, "Boosting source-free domain adaptation via confidence-based subsets feature alignment," in *2022 26th International Conference on Pattern Recognition (ICPR)*, pp. 2857–2863, IEEE, 2022.

[39] K. Xia, L. Deng, W. Duch, and D. Wu, "Privacy-preserving domain adaptation for motor imagery-based brain-computer interfaces," *IEEE Transactions on Biomedical Engineering*, vol. 69, no. 11, pp. 3365–3376, 2022.

[40] D. Kothandaraman, R. Chandra, and D. Manocha, "Ss-sfda: Self-supervised source-free domain adaptation for road segmentation in hazardous environments," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3049–3059, 2021.

[41] S.-H. Gao, Q. Han, D. Li, M.-M. Cheng, and P. Peng, "Representative batch normalization with feature calibration," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8669–8679, 2021.

[42] M. Bateson, H. Lombaert, and I. Ben Ayed, "Test-time adaptation with shape moments for image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 736–745, Springer, 2022.

[43] M. Jing, X. Zhen, J. Li, and C. Snoek, "Variational model perturbation for source-free domain adaptation," *Advances in Neural Information Processing Systems*, vol. 35, pp. 17173–17187, 2022.

[44] X. Luo, W. Chen, C. Li, B. Zhou, and Y. Tan, "Multi-level consistency learning for source-free model adaptation," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 12419–12426, 2022.

[45] J. Lee and G. Lee, "Feature alignment by uncertainty and self-training for source-free unsupervised domain adaptation," *Neural Networks*, vol. 161, pp. 682–692, 2023.

[46] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *International Conference on Machine Learning*, pp. 1050–1059, PMLR, 2016.
[47] F. Fleuret *et al.*, "Test time adaptation through perturbation robustness," in *NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2021.
[48] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan, "Augmix: A simple data processing method to improve robustness and uncertainty," in *International Conference on Learning Representations*, 2019.
[49] C. Zhou, Q. Li, C. Li, J. Yu, Y. Liu, G. Wang, K. Zhang, C. Ji, Q. Yan, L. He, *et al.*, "A comprehensive survey on pretrained foundation models: A history from bert to chatgpt," *arXiv preprint arXiv:2302.09419*, 2023.
[50] S. Thrun, "Lifelong learning algorithms," in *Learning to learn*, pp. 181–209, Springer, 1998.
[51] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 4, pp. 594–611, 2006.
[52] C. H. Lampert, H. Nickisch, and S. Harmeling, "Attribute-based classification for zero-shot visual object categorization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 3, pp. 453–465, 2013.
[53] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
[54] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, pp. 448–456, pmlr, 2015.
[55] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International Conference on Machine Learning*, pp. 6105–6114, PMLR, 2019.
[56] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," *Neural computation*, vol. 12, no. 10, pp. 2451–2471, 2000.
[57] D. Bahdanau, K. H. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *International Conference on Learning Representations*, 2015.
[58] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.
[59] J. D. M.-W. C. Kenton and L. K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of NAACL-HLT*, pp. 4171–4186, 2019.
[60] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901, 2020.
[61] J. Zhang, J. Huang, S. Jin, and S. Lu, "Vision-language models for vision tasks: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.
[62] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24824–24837, 2022.
[63] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do imagenet classifiers generalize to imagenet?," in *International Conference on Machine Learning*, pp. 5389–5400, PMLR, 2019.
[64] D. Hendrycks, K. Zhao, S. Basart, J. Steinhardt, and D. Song, "Natural adversarial examples," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15262–15271, 2021.
[65] D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo, *et al.*, "The many faces of robustness: A critical analysis of out-of-distribution generalization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8340–8349, 2021.
[66] H. Wang, S. Ge, Z. Lipton, and E. P. Xing, "Learning robust global representations by penalizing local predictive power," *Advances in Neural Information Processing Systems*, vol. 32, 2019.
[67] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *2008 Sixth Indian conference on computer vision, graphics & image processing*, pp. 722–729, IEEE, 2008.
[68] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi, "Describing textures in the wild," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3606–3613, 2014.
[69] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. Jawahar, "Cats and dogs," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3498–3505, IEEE, 2012.
[70] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3d object representations for fine-grained categorization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 554–561, 2013.
[71] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012.
[72] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 178–178, IEEE, 2004.

[73] L. Bossard, M. Guillaumin, and L. Van Gool, "Food-101–mining discriminative components with random forests," in *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part VI 13*, pp. 446–461, Springer, 2014.

[74] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3485–3492, IEEE, 2010.

[75] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi, "Fine-grained visual classification of aircraft," *arXiv preprint arXiv:1306.5151*, 2013.

[76] P. Helber, B. Bischke, A. Dengel, and D. Borth, "Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 12, no. 7, pp. 2217–2226, 2019.

[77] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2017.

[78] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *International Conference on Machine Learning*, pp. 1321–1330, PMLR, 2017.

[79] Y. Ge, J. Ren, A. Gallagher, Y. Wang, M.-H. Yang, H. Adam, L. Itti, B. Lakshminarayanan, and J. Zhao, "Improving zero-shot generalization and robustness of multi-modal models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11093–11101, 2023.

[80] P. Bachman, R. D. Hjelm, and W. Buchwalter, "Learning representations by maximizing mutual information across views," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[81] M. Sajjadi, M. Javanmardi, and T. Tasdizen, "Regularization with stochastic transformations and perturbations for deep semi-supervised learning," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[82] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[83] D. Hendrycks and T. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," in *International Conference on Learning Representations*, 2018.

[84] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *International Conference on Learning Representations*, 2014.

[85] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[86] S. Nowozin, B. Cseke, and R. Tomioka, "f-gan: Training generative neural samplers using variational divergence minimization," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[87] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International Conference on Machine Learning*, pp. 1861–1870, PMLR, 2018.

[88] M. Scutari, "Entropy and the kullback–leibler divergence for bayesian networks: Computational complexity and efficient implementation," *Algorithms*, vol. 17, no. 1, p. 24, 2024.

[89] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of big data*, vol. 6, no. 1, pp. 1–48, 2019.

[90] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[91] H. Wang, C. Xiao, J. Kossaifi, Z. Yu, A. Anandkumar, and Z. Wang, "Augmax: Adversarial composition of random augmentations for robust training," *Advances in Neural Information Processing Systems*, vol. 34, pp. 237–250, 2021.

[92] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le, "Randaugment: Practical automated data augmentation with a reduced search space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 702–703, 2020.