Deep Learning 2024 - Project Assignment

1 Introduction

Deep neural networks have enabled impressive results in many computer vision applications, such as image classification, object detection and tracking, and anomaly detection. However, they suffer from severe performance degradation when being tested on images that are visually different from the ones encountered during training due to the *domain shift*.

To address this problem, recent research has focused on devising domain adaptation techniques for building deep models that can adapt from an annotated source dataset to a target one. However, such methods require access to either downstream training data, which is difficult to collect, especially in real-world applications where privacy concerns may hinder data acquisition, or multiple samples from the training or testing split. Another line of work is **Test-Time adaptation** (TTA), which involves improving the robustness of a pre-trained neural network to a test dataset, possibly by improving the network's predictions on one test sample at a time.

In this assignment, your goal is to **construct, fine-tune, and evaluate a TTA method for the task of image classification**. You will be given two methods to draw inspiration from: Marginal Entropy Minimization with One test point (MEMO) [1] and Test-time Prompt Tuning (TPT) [2]. To evaluate your approach for TTA, you will be provided with a benchmark consisting of ImageNet variants [3, 4] with natural distribution shifts (e.g., images collected from different sources or hard to classify) that you will use as an evaluation set. The performance will be evaluated based on top-1 accuracy on the test datasets.

To complete the assignment, you are free to use any technique or algorithm in the literature, such as attention mechanisms or convolutional neural networks. You are also free to propose a novel method that is different from the approaches of MEMO [1] and TPT [2] as long as it is well-reasoned and sound. In such a case, we encourage you to discuss your idea with us first. This will allow us to provide feedback and ensure that your proposal aligns with the project's objectives.

2 Test-time adaptation

In this section, we first formalize Test-Time Adaptation (TTA) for the image classification task. We then describe MEMO [1] and TPT [2], two TTA methods for image classification using a single test sample at a time.

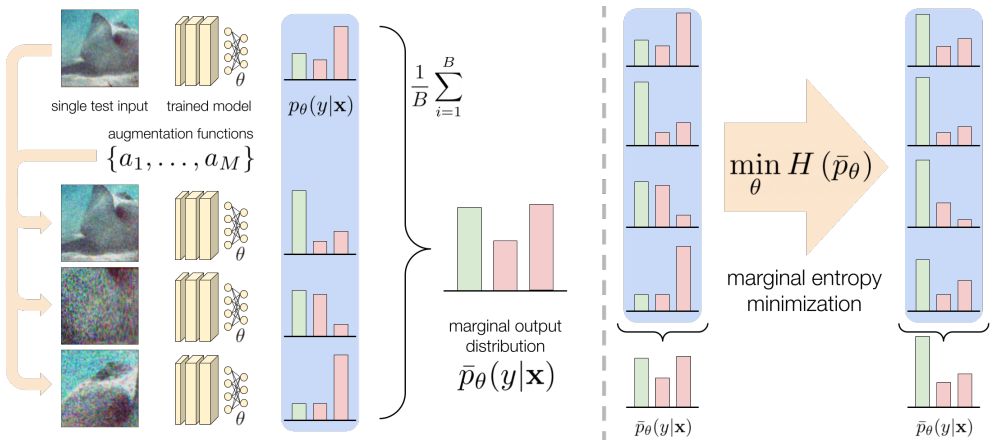

Fig. 1 Given an input test sample $\mathbf{x} \in \mathcal{X}$, MEMO [1] adapts a pre-trained model to a test dataset $\mathcal{X}$ by first augmenting $\mathbf{x}$ using a set of transformations $\{a_1, \ldots, a_M\}$, and then minimizing the entropy of the marginal distribution $\bar{p}(y|\mathbf{x}; \mathbf{w})$ obtained by averaging the conditional output distributions over the augmented views $p(y|a_i(\mathbf{x}); \mathbf{w})$.

Given a test sample $\mathbf{x} \in \mathcal{X}$, a deep neural network $\mathcal{M}$, parameterized by $\mathbf{w} \in \mathcal{W}$, for image classification produces a conditional output distribution $p(y|\mathbf{x}; \mathbf{w})$ over a set of classes $\mathcal{Y}$ and predicts a label $\hat{y}$ as:

$$\hat{y} = \mathcal{M}(\mathbf{x}|\mathbf{w}) = \underset{y \in \mathcal{Y}}{\operatorname{argmax}}\, p(y|\mathbf{x}; \mathbf{w}) \tag{1}$$

TTA aims to improve the performance of a pre-trained model $\mathcal{M}$ when applied to a test dataset $\mathcal{X}$.

Let $\mathcal{A} = \{a_1, \ldots, a_M\}$ be a set of augmentation transformations, such as resizing, cropping, and color jittering. Each augmentation $a_i \in \mathcal{A}$ can be applied to an input sample $\mathbf{x}$, resulting in a transformed sample denoted as $a_i(\mathbf{x})$. During training, for each pair $(\mathbf{x}^{train}, y^{train})$, we apply augmentation functions sampled from $\mathcal{A}$ to make the model's prediction $\hat{y}^{train} = \mathcal{M}(\mathbf{x}^{train}|\mathbf{w})$ invariant to those specific transformations. Both MEMO [1] and TPT [2] employ a strategy similar to image augmentations during training, but during testing when the sample label y is unknown.

As shown in Fig. 1, MEMO starts by applying a set of B augmentation functions sampled from $\mathcal{A}$ to $\mathbf{x}$. Then, it calculates the average, or marginal, output distribution $\bar{p}(y|\mathbf{x}; \mathbf{w})$ by averaging the conditional output distributions over this set of augmentations, represented as:

$$\bar{p}(y|\mathbf{x}; \mathbf{w}) = \frac{1}{B}\sum_{i=1}^{B} p(y|a_i(\mathbf{x}); \mathbf{w}) \tag{2}$$

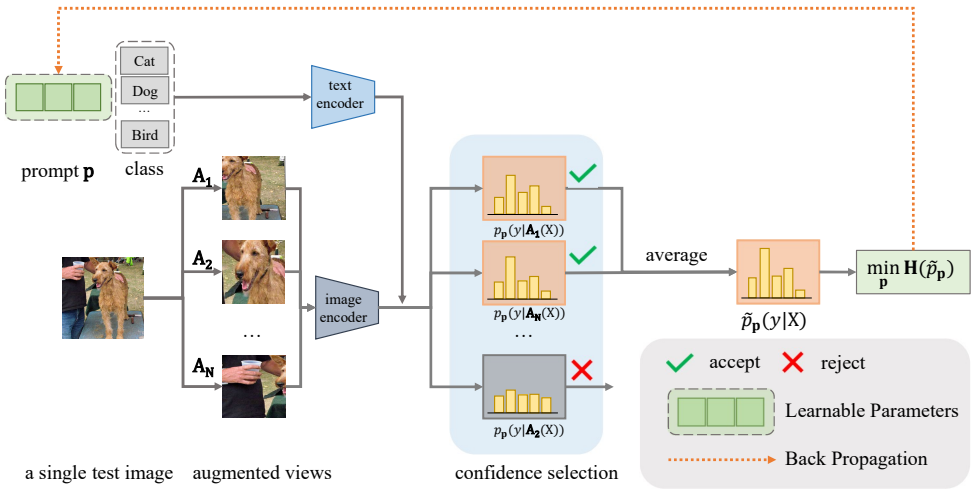

Fig. 2 Given an input test sample $\mathbf{x} \in \mathcal{X}$, TPT [2] adapts a pre-trained vision-language model to a test dataset $\mathcal{X}$ by first augmenting $\mathbf{x}$ using a set of transformations $\{a_1, \ldots, a_N\}$, then selecting the subset of the most confident augmented views, and finally minimizing the entropy of the marginal distribution $\overline{p}(y|\mathbf{x};\mathbf{w})$ obtained by averaging the conditional output distributions over the selected views $p(y|a_i(\mathbf{x});\mathbf{w})$.

As the sample label y is not available during testing, TTA's objective is twofold: (i) to ensure that the model's predictions have the same label y across various augmented versions of the test sample, (ii) to increase the confidence in the model's predictions, given that the augmented versions have the same label. To this end, the model is trained to minimize the entropy of the marginal output distribution across augmentations, defined as:

$$\mathcal{L}(\mathbf{w};\mathbf{x}) = \mathbf{H}(\overline{p}(\cdot|\mathbf{x};\mathbf{w})) = -\sum_{y\in\mathcal{Y}} \overline{p}(y|\mathbf{x};\mathbf{w}) \log \overline{p}(y|\mathbf{x};\mathbf{w}) \tag{3}$$

TPT [2] is based on MEMO [1], extending its application to vision-language models (VLMs) [5]. VLMs are deep neural networks trained on immense amounts of image-text data collected from the web, and they are known for their remarkable zero-shot transfer capabilities, i.e. their features can be used for various downstream tasks without retraining, by manually designing prompts. Nevertheless, the performance of VLMs in downstream tasks can be further improved by directly learning the prompts instead of handcrafting them [6, 7].

To adapt VLMs to downstream tasks while avoiding access to training data, TPT [2] fine-tunes the prompt on the fly on a test sample $\mathbf{x} \in \mathcal{X}$, as shown in Fig. 2. Similarly to MEMO [1], TPT generates N augmented views of a single test image $\mathbf{x}$ through augmentation transformations from $\mathcal{A}$ and minimizes the entropy of the marginal output distribution, i.e., the average of the conditional output distribution of

augmented views. In addition to MEMO, TPT devises a confidence selection strategy based on selecting the augmented views with the lowest self-entropy (i.e., the most confident augmented views) to account for augmentations (e.g., cropping) that may result in augmented views with incorrect semantics.

3 Assignment

Project Objective

We request that you develop a method for TTA that enhances the predictive capabilities of a network on test samples $\mathbf{x} \in \mathcal{X}$, without having access to the labels of the test samples or the remaining data in $\mathcal{X}$. Before analyzing each new sample, the state of the network must be reset to the initial configuration (i.e., reloading the pre-trained weights).

Suggestions

You can achieve this goal by re-implementing MEMO [1] or TPT [2] and showing that by applying TTA you obtain better performance metrics on the test datasets (see Sec. 3.1) compared to a baseline that does not use TTA. For MEMO, the baseline can be the same deep neural network before applying TTA, while for TPT a fair baseline could consider zero-shot CLIP [5] (with hand-crafted prompts or few-shot prompt tuning methods such as CoOp [6] and CoCoOp [7]). Please refer to [1, 2, 5] for more details about baselines.

We also encourage you to propose, implement, and evaluate modifications to the original methods and show improvements in accuracy or efficiency. Remembering that you do not have access to the label information of the test sample at any stage, except for the evaluation of the model, here are some potential directions for improving the accuracy or efficiency of MEMO [1] or TPT [2]:

- Explore alternative image augmentation techniques, possibly leveraging text-to-image generative models, while ensuring the generated images remain within the test distribution.
- Replace the confidence selection strategy proposed in TPT. For example, you may use an approach that evaluates the text-image alignment between the class name and the augmented view and selects the highest-scoring views.
- Experiment with alternative unsupervised learning losses beyond entropy minimization.

3.1 Datasets and evaluation metrics

For the TTA task in this assignment, you will evaluate your proposed method on ImageNet-A [3] and ImageNet-V2 [4]. For the duration of the course, both datasets will be available on Google Drive[1].

ImageNet-A is a dataset of real-world images easily recognizable by humans but misclassified by methods trained on the original ImageNet. Specifically, it contains 7,500 images from 200 ImageNet-1K classes, whose classification error is considered

[1] https://drive.google.com/drive/folders/1a1-NOxfpX-IC0qMzWD9TIiBGGi6rznOb

Method	Backbone	# Params. ↓	Top1 Acc. ↑	
			ImageNet-A	ImageNet-V2
MEMO	CLIP (ViT-B/16)	86M	51.59	62.64
TPT + CoOp	CLIP (ResNet50)	77M	30.32	57.83
TPT + CoOp	CLIP (ViT-B/16)	125M	**57.95**	**66.83**

Table 1 Results of MEMO and TPT on ImageNet-A and ImageNet-V2.

semantically significant (e.g., mistaking a Persian cat for a candle is semantically more wrong than misclassifying Norwich terriers as Norfolk terriers).
ImageNet-V2 is a collection of three test sets of natural images, with 10,000 new images each sharing the same ImageNet categories, collected from multiple sources. Among the three test sets proposed by the authors, for this assignment, we are interested in the *matched frequency* one.

Your project's performance will be measured by top-1 accuracy (i.e., the percentage of samples for which the predicted label matches the target label) over the entire test sets of both datasets. For reference, we report the results of MEMO [1] and TPT [2] in Tab. 1. Specifically, for MEMO, we fine-tuned the image encoder using random resized crops as augmentations, and we trained it with AdamW and a learning rate of $5e-5$[2].

4 Project evaluation

We expect the project to be self-contained within a Jupyter Notebook hosted on Google Colab. The notebook must contain code and text cells, which are meant to integrate the code with comments, and they should form the final project report. The report must include:

- a thorough description of the solution you adopted. You must include a detailed overview of the architecture, the losses, and the overall pipeline. Make sure to highlight your original contributions and cite the literature if needed. It is important you carefully motivate each design decision you take, particularly when their motivation is nontrivial. You are also encouraged to enhance the description by incorporating images within the text cells;
- an extensive presentation and discussion of the results obtained with your solution. Organize the scores obtained in tables, including charts depicting learning curves, confusion matrices, and other useful visual representations. Structure the discussion in such a way that emphasizes the take-home message of your findings.
- **add comments to the code**. Comments must be clear, concise, and nontrivial, proving that you are aware of all the steps that you go through during the development and evaluation of the code;
- we expect to run the notebook from the first to the last code cell without any additional step required and without errors or crashes. Should your code require extra steps to run, please include them in the report.

Your project will be evaluated according to the following metrics:

[2]This is an example of what you can do with MEMO and a recent language and vision architecture.

- methodological thoroughness;
- clarity of the report;
- performance with respect to validation accuracy;
- originality of the solution;
- quality of the code.

In order to deliver the solution, send an email to francesco.tonini@unitn.it with e.ricci@unitn.it in cc. The email object must be `[DL2024] - Project delivery`, with your self-contained notebook in .ipynb format attached to it. The name of the notebook must follow the following format: `[student id 1] [student id 2] [student id 3].ipynb`. **IMPORTANT:** make sure that the file can be seamlessly loaded and executed into Google Colab. **The solution must be delivered strictly not later than 7 days before the date on which you decide to take the oral exam.**

5 Group registration

The project is intended for groups of 2 or 3 people. You can register your group by filling out the Google Form[3]. **The registration deadline is May, 6th**.

6 Policies

We require you **not to start from an existing GitHub repository** containing code developed by others. Of course, you can install common libraries such as matplotlib, scikit-learn, Pillow, and many more. You are allowed to choose any existing architecture as your backbone as long as pre-trained weights on ImageNet-1K are available. **It is strictly forbidden to share the source code with other groups**. Also, keep in mind that your code, along with the project report, will be checked against plagiarism tools.

Any violation of these policies will result in actions being taken towards all group participants: it is each group's responsibility to comply with the rules.

References

[1] Zhang, M., Levine, S., Finn, C.: Memo: Test time robustness via adaptation and augmentation. Advances in neural information processing systems **35**, 38629–38642 (2022)

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

[3] Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., Song, D.: Natural adversarial examples. CVPR (2021)

[3] https://forms.gle/DYBThDn8yXkaTzR97

[4] Recht, B., Roelofs, R., Schmidt, L., Shankar, V.: Do imagenet classifiers generalize to imagenet? In: International Conference on Machine Learning, pp. 5389–5400 (2019). PMLR

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

[6] Zhou, K., Yang, J., Loy, C.C., Liu, Z.: Learning to prompt for vision-language models. International Journal of Computer Vision (IJCV) (2022)

[7] Zhou, K., Yang, J., Loy, C.C., Liu, Z.: Conditional prompt learning for vision-language models. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022)