MULTIMODAL PROMPT TUNING: REAL-WORLD USAGE ON NTU TREE CLASSIFICATION.

R11922147 林鴻儒[1], R11922A15 張仲喆[1], R11944004 李勝維[2] and R11944021 廖金億[2]

[1]Department of Computer Science and Information Engineering, National Taiwan University
[2]Graduate Institute of Networking and Multimedia, National Taiwan University

ABSTRACT

In this research, we designed and collected a tree dataset, named "Trees in NTU Campus," and used it to evaluate the performance of soft prompts in a multimodal model, CLIP. We designed three different soft prompts for the CLIP model, and also compared the performance with traditional finetuning method on the same task. The goal of our research was to investigate the potential of multimodal prompt tuning to converge on solutions with limited time and training resources. Our results showed that the model with the vision prompt performed the best, and the performance of the vision-language prompt was second only to the pure vision prompt, while the performance of the pure language prompt was far worse. Traditional finetuning method took longer to train in order to achieve similar performance. Our research demonstrates the effectiveness of soft prompts in training models with limited resources and highlights the importance of considering different prompt design choices.

***K*eywords** Prompt tuning · Multimodal deep learning · "Trees in NTU Campus" · Texture classification

1 Introduction

Adapting large pre-trained models to recognition tasks is an effective method for high accuracy. However, this approach can present challenges, such as the need to store and deploy a separate copy of the model's parameters for each task during fine-tuning, particularly in the case of Transformer-based architectures.

To address these challenges, prompt tuning has gained significant attention in various fields of deep learning these years. This technique involves introducing a prompt or additional input to a pre-trained model in order to guide it towards a specific task or output.

In this project, we designed and collected our own tree dataset from the NTU campus, named "Trees in NTU Campus." Since the dataset uses uncommon feature, the stem of a tree, it tests the generalizability of the model and also challenges the model's learning capability on a limited size of dataset.

We designed three different soft prompts for the multimodal CLIP model, also compared the performance with SOTAs on the same task. The goal of our research was to investigate the potential of multi-modal prompt tuning to converge on solutions with limited time and training resource.

2 Related Work

2.1 Vision Transformer

Vision transformer[1] is a neural network architecture that uses self-attention mechanisms[2] to process visual data instead of convolutional layers. It divides images into patches, encodes the patches with self-attention mechanisms, and classifies them with a final MLP layer. Vision transformer has several advantages compared to convolution neural network, including flexibility, robustness, efficiency, and good performance on image recognition tasks.

2.2 CLIP

CLIP [3] a visual model trained under natural language supervision. CLIP consists of two components: a vision encoder and a text encoder. The vision encoder is a ResNet-based[4] or a Vision Transformer-based[1] architecture; the text encoder is a BERT-based[5] architecture. CLIP is trained with InfoNCE[6] loss on a large corpus of web-crawled image-text pair.

This approach brings two major benefits to CLIP compared to traditional supervised visual learning:

1. The learnt visual features are transferable from task to task: Since natural language is a high-level understanding of the world, the natural language supervision makes the visual features very robust and generalized.
2. CLIP can do zero-shot tasks with the aid of its language branch: For instance, if the task of a dataset is classifying photos of dogs vs cats, we check for each image whether a CLIP model predicts the text description "a photo of a dog" or "a photo of a cat" is more likely to be paired with it.

2.3 Soft-Prompt in NLP

Since GPT-3[7] demonstrated its good zero-shot performance on NLP problems using prompts, the research on prompts has become increasingly popular. However, it is difficult to find effective prompts through hand-crafting, and even slight differences in the content of the prompts can greatly affect the results. As a result, there have been many subsequent studies on how to find suitable prompts, including studies that try to train a language generation model to learn to generate prompts[8] [9]. Subsequent research has also found that using discrete prompts, which correspond to natural language, may not be the best approach, and that using continuous vector soft prompts may achieve better results[10] [11] [12].

P-tuning[11] was one of the first approaches to introduce the use of soft prompts for a variety of NLP tasks. Prefix tuning[10] also applied the soft prompt approach to NLG tasks and proposed adding soft prompts to each layer rather than just the initial embedded layer. P-tuning v2[12] improved upon the original P-tuning approach and introduced the multi-layer soft prompt method from Prefix tuning, which showed very good results on NLP tasks.

3 Approach

3.1 Custom Dataset: Trees in NTU campus

The National Taiwan University (NTU) campus is home to a diverse array of tree species, however, many of which may be difficult to distinguish for the average person. In an effort to increase familiarity with the campus's natural beauty, a dataset of tree pictures was collected using our own personal cellphones.

The tree dataset we have compiled includes 15 different species, as shown in Figure 1, such as the Royal Palm, Indigenous Cinnamon Tree, and White Barkfig, among others. Within each class, there are 8 to 16 training images, and the remaining 8 to 10 images serve as test data.

The stem of the tree is used as the feature for our dataset. To minimize inter-class variance, the distance between the cellphone and the tree was standardized to one span unit. In contrast, to maximize intra-class variance, the trees were captured at different times, angles, and under varying shades and shadows.

After collecting the tree dataset, We conducted experiments using our tree dataset to evaluate the performance of prompt tuning, and also compared it to traditional methods such as vision transformers and convolution-based neural networks.

3.2 Multimodal Prompt Tuning

3.2.1 Backbone: CLIP

In our dataset, we need to identify the tree class of image $I \in \mathbb{R}^{H\times W\times 3}$ from *15* class. We use CLIP[3] model as backbone and compare the three kinds of prompting method. As shown in Figure 2, CLIP encode image *I* and corresponding text description as explained below.

Encoding Image: First, we split the image *I* into *M* fixed-size patches. Secondly, we project the patches into patch embeddings $E_0 = \{e_{1,0}, e_{2,0}, ..., e_{M,0}\} \in \mathbb{R}^{(M\times d_v)}$. Third, we fed patch embeddings E_{j-1} with CLS token c_{j-1} to V_j transformer layer and sequentially processed through *J* transformer blocks.

Figure 1: Our own datasets: "Trees in NTU Campus".

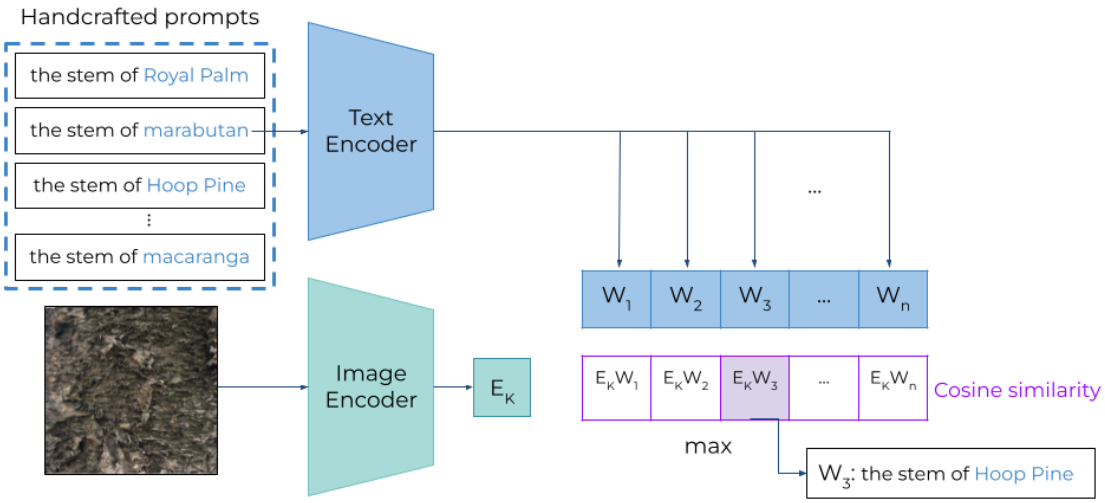


Figure 2: Zero-shot CLIP classification

$$[c_j, E_j] = V_j([c_{j-1}, E_{j-1}]), \quad j = 1, 2, 3, ..., J \tag{1}$$

Finally, we project the CLS token output of the last transformer layer via *ImageProj* as the encoding of image *I*,

$$x = ImageProj(c_J), \quad x \in \mathbb{R}^{d_v} \tag{2}$$

Encoding Text: Similar with the image encoding method. First, project the description of n-th class into embedding $W_0^n = [w_{1,0}^n, w_{2,0}^n, ..., w_{T,0}^n]$. Secondly, we fed embeddings W_{j-1}^n to the j-th layer of Language Encoder (L_j) to get the next embedding W_j^n and sequentially processed through *J* transformer blocks.

$$[W_j^n] = L_j(W_{j-1}^n), \quad j = 1, 2, \cdot \cdot \cdot J \tag{3}$$

Finally, we got the final text representation z^n from the text embeddings corresponding to the last token of the last transformer block L_J via *TextProj*.

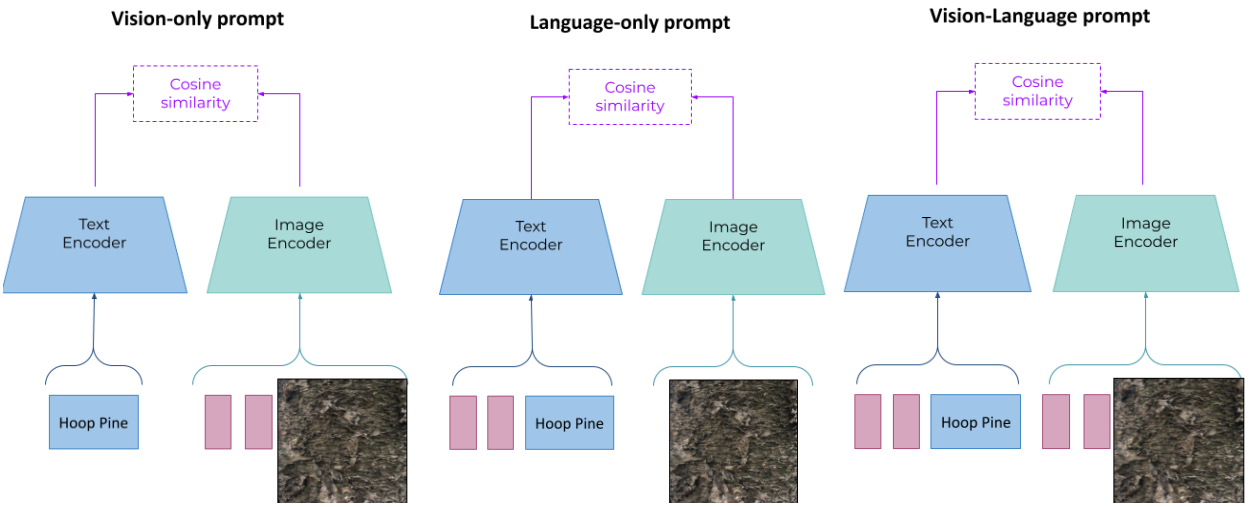
$$z^n = TextProj(w_{T,J}^n), \quad z^n \in \mathbb{R}^{d_v} \tag{4}$$

Classification: We choose the class with highest score calculate by similarity between z^n and x, which is formulated as:

$$p(y_n|x) = \frac{exp(sim(x, z^n)/\tau)}{\sum_{n=1}^{N} exp(sim(x, z^n))}, \quad n = 1, 2, ..., N \tag{5}$$

Here, τ is a temperature parameter and $sim(\cdot, \cdot)$ is the cosine similarity.

3.2.2 Vision-only soft prompt



(a) Vision-only prompt (b) Language-only prompt (c) Vision-Language prompt

Figure 3: Three different prompt strategies.

To learn the Vision prompts. Similar to P-Tuning v2[12], we initialized prompt for each layer of vision encoder layer V, each of the prompt with I learnable tokens $P_j^V = \{P_{i,j}^V \in \mathbb{R}^{d_v}\}_{i=1}^{I}$. Prompt $P_{*,j-1}^V$ is fed to the j-th layer of vision encoder V with the embedding of CLS token c_{j-1} and image $E_{*,j-1}$.

$$[c_j, E_{*,j}, \_] = V_j([c_{j-1}, E_{*,j-1}, P_{*,j-1}^V]), \quad j = 1, 2, 3, ..., J \tag{6}$$

3.2.3 Language-only soft prompt

Similar with the vision prompt, as shown in 3b, We create prompt for each layer of language encoder L with I learnable tokens $P_j^L = \{P_{i,j}^L \in \mathbb{R}^{d_v}\}_{i=1}^{I}, \quad j = 1, 2, ..., J$. Prompt $P_{*,j-1}^L$ is fed to the j-th layer of language encoder (L_j) with the embedding of class name $W_{*,j-1}$.

$$[\_, W_{*,j}] = L_j([P_{*,j-1}^L, W_{*,j-1}]), \quad j = 1, 2, \cdot \cdot \cdot J \tag{7}$$

3.2.4 Vision-Language soft prompt

The Vision-Language soft prompt combines the above two prompt methods shown in 3c, adding soft prompts to both the vision and language encoders. Here, we use the MaPLe [13] approach, where the vision prompt is formed through a multilayer perceptron from the language prompt.

The language prompt method is the same as the Language-only soft prompt. And the image prompt method is similar with the Vision-only soft prompt. The only difference is that the vision prompt $P_{*,j}^V$ is not a seperate set of vector, but instead we train a MLP F_j to project the language prompt $P_{*,j}^L$ to $P_{*,j}^V$.

$$P_{*,j}^L = F_j(P_{*,j}^V) \tag{8}$$

4 Experiment

In our experiment, we use the collected the “Trees in NTU Campus” dataset to compare the effects of different soft-prompt design on the CLIP [3] model. We also compared the results with fintuned Vision transformer[1] model and ResNet50[4] model, which are widely used in image classification.

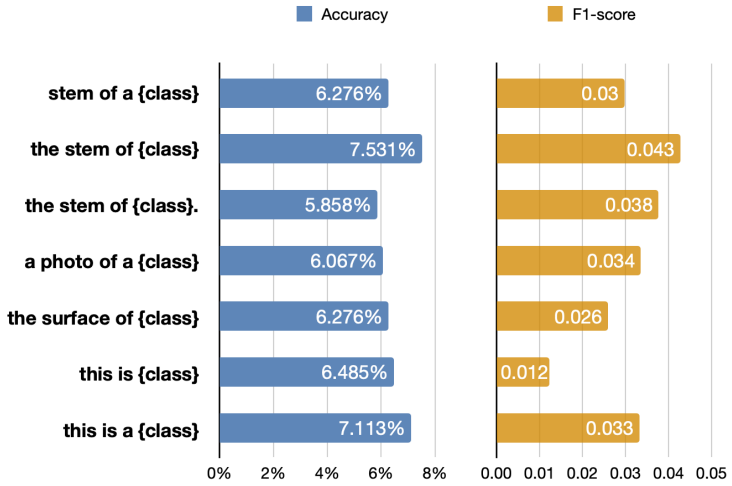


Figure 4: Performance of different discrete prompt designs. Different discrete prompts can result in large performance differences. Even small prompt differences that are not important to humans can have a significant impact on the model.

Discrete prompt We tried several types of discrete prompts to evaluate their influence on the model, as shown in Figure4. Different discrete prompt settings do affect the performance of the model, and just "whether a period is added" can greatly affect the accuracy of the prediction. However, it can still be found that the overall performance of the model using discrete prompts is still far from usable. Therefore, in our experiment, we adopted the method of initializing a language discrete prompt and initializing the vision prompt as random Gaussian distribution. Then we let the model learn to tune a suitable soft prompt on its own.

Soft prompt. We follow previous works [13, 1, 4] for experiment settings. Inputs tree images are resized to 224 x 224 using bicubic interpolation and normalized as CLIP [3] settings. We also apply two data augmentation methods, random_resized_crop and random_flip, to enhance model’s robustness and performance. In our experiments, we initialize the prompt as a hard prompt, "the stem of," and then let the model learn the soft prompt presentation on its own. We train models for a fixed 100 epochs using SGD optimizer with learning rate = 0.0035 and cosine annealing on the exact same server with a single Nvidia 3090 GPU. We mainly focus on the performance of different models with similar training time.

It can be found that, for a short time (two minutes), the model with the vision prompt performs the best, and the performance of the vision-language prompt is second only to the pure vision prompt, while the performance of the pure language prompt is far worse than the former. Traditional methods (Vision transformer and ResNet50) take longer to train in order to achieve similar performance.

5 Discussion

According to our findings in Figure5, the language prompt performed the worst in this particular instance. One reason for this is that the language prompt did not provide any useful information in this case. The names of tree species are mostly scientific names, and there is no way to give the model hints about Tree recognition. As a result, the Vision-Language prompt performed slightly worse than the vision prompt.

Further research has indicated that larger inter-class text variance makes the vision prompt more effective. This is likely due to the fact that the class names in question have very different word embeddings.

It is worth noting that these results may not be generalizable to all situations. Further studies should be conducted in order to determine the optimal approach for utilizing language prompts in various contexts. In the meantime, it may

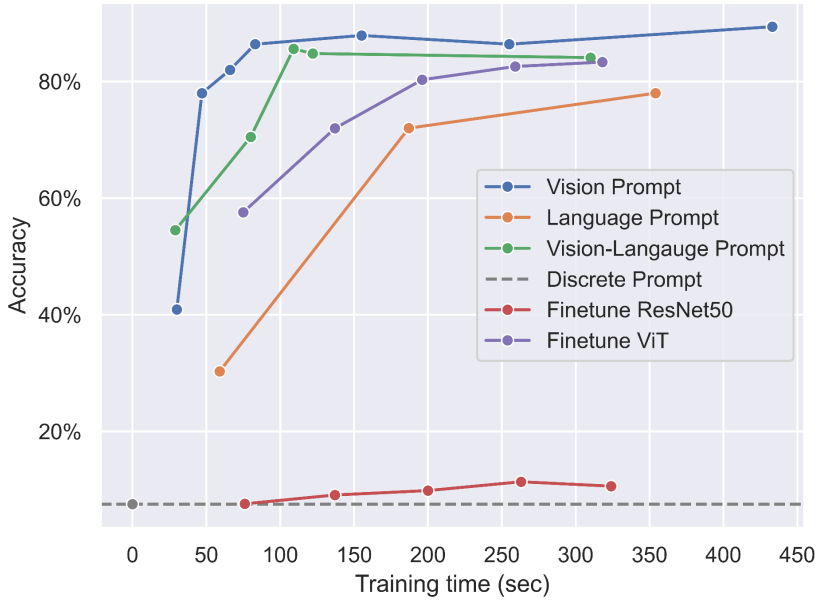


Figure 5: Performance vs. training time.

be advisable to consider the usage of vision prompts in cases where language prompts are not providing sufficient information.

6 Conclusion

In this research, we collected the "Tress in NTU Campus" dataset and demonstrated the ability of soft-prompt to train a model with very limited time and resource. We also compared the performance difference between traditional finetuning and three different prompting strategies including language-only soft-prompt, vision-only soft-prompt and vision-language soft-prompt. With different prompting strategies, we observed that each had very different performances in the training process, but all soft-prompt based methods can achieve high performance in less than two minutes. We believe this research helps the situation where the data and quantity are small and the training resources are limited.

7 Work Distribution

Work Distribution

李勝維	Topic proposal, Designing discrete prompt, Vision-Language prompt, Dataset collection, Presentation, and Report
廖金億	Vision prompt, Finetuning ResNet, Dataset collection, Presentation, and Report
林鴻儒	Language prompt, Dataset collection, Presentation, and Report
張仲喆	Finetuning Vision Transformer, Dataset collection, Presentation, and Report

References

[1] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2020.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[3] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[6] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding, 2018.

[7] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[8] Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438, 2020.

[9] Laria Reynolds and Kyle McDonell. Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–7, 2021.

[10] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[11] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too. *arXiv preprint arXiv:2103.10385*, 2021.

[12] Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*, 2021.

[13] Muhammad Uzair Khattak, Hanoona Rasheed, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Maple: Multi-modal prompt learning. *arXiv preprint arXiv:2210.03117*, 2022.