Efficient Transfer Learning driven by Layer-wise Features Aggregation

Chanwoo Kim
University of Seoul
kcw6621@uos.ac.kr

Jeyoon Yeom
Yonsei University
jeyoon.yeom@yonsei.ac.kr

Joowang Kim
Yonsei University
zuwang@yonsei.ac.kr

Suho Kang
Yonsei University
suhokang@yonsei.ac.kr

Kyungwoo Song
Yonsei University
kyungwoo.song@yonsei.ac.kr

Abstract

Large-scale pre-trained models capable of extracting generalized features from huge data have become a key component on various downstream tasks. However, it is challenging to achieve substantial performance improvement with efficiency. There have been many works, such as prompt tuning and adapters, but both the efficiency and the performance improvement are limited. We propose a novel approach called Layer-wise Feature Aggregation (LFA), utilizing features from all layers of a pre-trained model with attention mechanism. Our focus is on utilizing existing low level features rather than generating new ones. First, LFA captures hierarchical features from low-level to high-level, enabling the extraction of richer and more general features; therefore, it significantly improves the performance in domain shift and few-shot learning. Second, LFA requires optimization only on top of large pre-trained models. Therefore, LFA optimization is efficient because it does not require back-propagation through the model. LFA is a new efficient transfer learning approach with improved performance and efficiency. Our methods are implemented at: https://github.com/MLAI-Yonsei/LFA

1 Introduction

Transfer learning applies pre-trained models to new tasks, leveraging learned patterns to enhance performance with less data and training time [Weiss et al., 2016, Neyshabur et al., 2020]. Vision language models (VLMs), such as CLIP [Radford et al., 2021], combine vision and natural language processing to perform tasks like image interpretation and captioning. However, fine-tuning large models is costly and performance varies based on the method used. To address this, strategies like prompt learning [Zhou et al., 2022a,c, Khattak et al., 2023] and adapters (e.g., CLIP-Adapter [Gao et al., 2021b, Zhang et al., 2022] and LoRA [Hu et al., 2021]) optimize models using smaller datasets and parameters. Although pre-trained parameters are frozen, these methods still demand considerable computational resources as gradients pass through the entire model. Previous work focused mainly on the final layer, overlooking useful features from earlier layers. We propose that low-level features are robust and invariant to domain shifts, making them valuable for fine-tuning.

Building on this foundation, we introduce a novel approach termed **Layer-wise Feature Aggregation (LFA)**. **First**, this approach eliminates the need for the gradient to flow through the model during training, thereby expediting both training time and memory usage. **Second**, We employ an attention mechanism that dynamically weights and aggregates features across all layers, allowing the model to focus on the most relevant features for a given task while leveraging the abundant information available. This approach indicates that lower layers can effectively compensate for the feature

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

a lack of representation and diminished performance under distribution shifts, we have demonstrated that our approach can theoretically preserve feature representation more effectively, enhancing robustness to out-of-distribution (OOD) scenarios. **Third**, our method offers notable flexibility, as it can be easily applicable with existing CLIP-based SOTA models. This user-friendly approach enables promoting widespread adoption among major users and supporting sustainable development. **The overall model structure and data flow** are illustrated in Figure 2. We first generate a query vector from the hidden vector of the visual encoder's last layer and compute key and value vectors from all layers to calculate the self-attention value, $\mathbf{z}^{(v)}$. In the textual encoder, we calculate self-attention ($\mathbf{z}self^{(t)}$) and cross-attention ($\mathbf{z}cross^{(t)}$) values by reusing the query vector from the visual encoder. The final output vector, $\mathbf{z}^{(t)}$, is derived using a hyperparameter β. Finally, we compute the logits from the final output vectors of both encoders, $\mathbf{z}^{(v)}$ and $\mathbf{z}^{(t)}$.

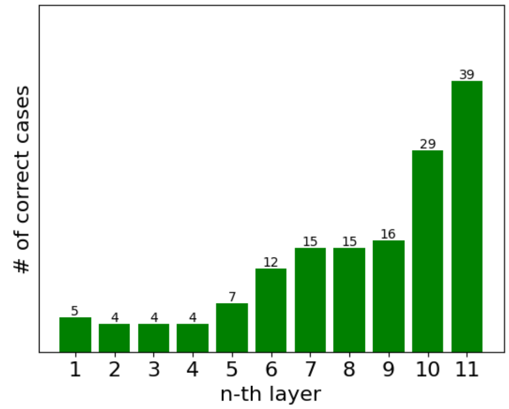

Figure 1: In the DomainNet real benchmark, we analyzed the correction of errors in the 12th layer by other layers. In the Linear-probing CLIP study, **2∼22%** of errors in the final layer were accurately corrected by earlier layers, demonstrating that lower layers effectively compensate for gaps in feature extraction.

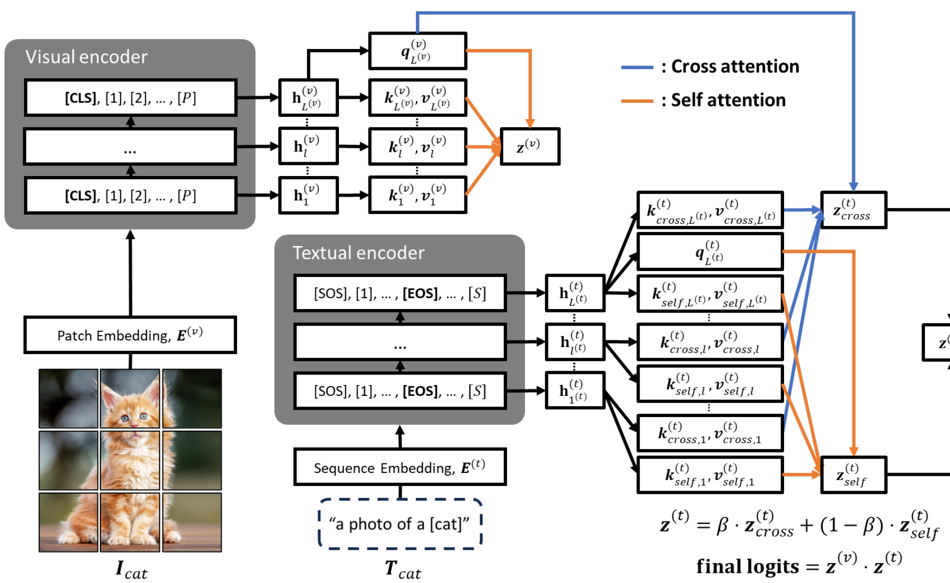

Figure 2: **The Pipeline of CLIP + LFA.** Our model employs the ViT [Dosovitskiy et al., 2020] model as its visual encoder and the transformer [Vaswani et al., 2023] model as its text encoder. Our approach leverages feature vectors from all layers, producing a richer and more stable representation. This strategy improves generalization and limits backpropagation to feature aggregation, boosting memory and computational efficiency.

extraction deficiencies of the final layer, as shown in Figure 1. Furthermore, while the sequential multi-layer computation of neural network models inevitably reduces the rank of features, leading to

2 Methodology

2.1 Feature Extraction Using the CLIP backbone

In contrast to conventional CLIP-based methods that primarily depend on the output of the final encoder layer, this paper enhances the representational capacity of the feature vector by incorporating features from multiple levels across all layers. We select the hidden vector $\mathbf{h}_{l_1}^{(v)} \in \mathbb{R}^{d_h^{(v)}}$ using the CLS token. For the textual part, We use the EOS token, which is commonly employed to detect sequence end and grasp context [Guo et al., 2023], allowing us to extract the textual hidden feature vector $\mathbf{h}_{l_2}^{(t)} \in \mathbb{R}^{d_h^{(t)}}$By integrating early layers' invariant information with later layers' semantic information, we obtain a rich set of feature vectors $\mathbf{h}^{(v)}1, \ldots, \mathbf{h}^{(v)}L^{(v)}, \mathbf{h}^{(t)}1, \ldots, \mathbf{h}^{(t)}L^{(t)}$ for effective downstream tasks. For more details, refer to the Appendix B.

2.2 Attention-Based Feature Aggregation

We adopted the attention method, which is task-specific by querying the last layer's feature vector and can activate the appropriate layers according to the training input data. Our model processes hidden feature vectors from visual and textual encoders using an attention mechanism. This involves generating query, key, and value matrices to focus on relevant information. Visual features are more diverse and discriminative compared to textual features, which remain consistent within the same class. This disparity creates a challenge in aligning the two modalities. To address this, we use self-attention for each modality and cross-attention to enhance textual features by incorporating visual information. The attention outputs from both modalities are merged using a hyperparameter β, balancing the contributions of self-attention and cross-attention for better model performance.

2.3 Merge Similarities and Prediction

The attention mechanism's value vectors for both visual and textual modalities share the same space, enabling a contrastive learning framework. The logits are computed using a dot-product between the output vectors of the image and the text for each class. The computed logits are used to calculate prediction probabilities with the softmax function, facilitating learning of similarities between positive and negative pairs. The temperature parameter τ controls the distribution scale. The LFA model fine-tunes the pre-trained CLIP backbone by aggregating features from different layers without backpropagation through the backbone. This makes the approach computationally efficient and enhances representational capacity for downstream tasks. The model uses query vectors from the last layer of the visual encoder and generates key and value vectors across all layers. Self-attention and cross-attention operations are performed in both visual and textual encoders. The final output vectors for both modalities are merged using a hyperparameter β and logits are calculated.

2.4 Theoretical Analysis

We theoretically show that LFA counters the reduction in rank in neural networks [Feng et al., 2022], which leads to overfitting due to an 'Independence Deficit'—a lack of independent representations—resulting in poorer performance, particularly in out-of-distribution scenarios. By proving two theorems, we demonstrate that LFA effectively preserves the rank within linear multi-layer networks, mitigating this issue. Detailed proofs of the theorems are provided in the Appendix F.

	Accuracy (%)					
Model	VLCS	PACS	OfficeHome	Terra	DomainNet	Avg.
ViT-B / 16 [11] with pre-trained weights from CLIP [44]						
ZS-CLIP(C) † [Radford et al., 2021]	76.4	95.7	79.9	33.9	57.8	68.7
ZS-CLIP(PC) † [Radford et al., 2021]	82.4	**96.1**	82.3	31.3	57.7	70.0
MIRO † [Cha et al., 2022]	82.2	95.6	82.5	**54.3**	54.0	73.7
ZS-CLIP(PC) + DPL [Zhang et al., 2021]	81.5	<u>95.8</u>	<u>82.6</u>	44.2	**59.3**	72.7
ZS-CLIP(PC) + QLoRA [Dettmers et al., 2024]	82.5	**96.1**	82.5	43.4	59.0	72.7
ZS-CLIP(PC) + LFA	81.0	**96.1**	**82.7**	50.3	<u>59.2</u>	<u>73.9</u>
ZS-CLIP(PC) + QLoRA + LFA	<u>81.9</u>	**96.1**	81.6	<u>52.0</u>	58.7	**74.1**

Table 1: Domain generalization accuracy with various comparison models on DomainBed benchmark. † indicates the numbers taken from Table 2 in [Cha et al., 2022, Zhang et al., 2021]. The best scores are bolded, and the second-best scores are underlined. CLIP + LFA demonstrates superior average performance compared to DPLCLIP, MIRO, and QLoRA models.

Method	Setting	ImageNet	Caltech101	OxfordPets	StanfordCars	Flowers102	Food101	FGVCAircraft	SUN397	DTD	EuroSAT	UCF101	Average
Zero-Shot CLIP Radford et al. [2021]	0-shot	66.73	93.31	89.13	65.64	70.13	85.86	24.72	62.56	44.03	48.44	67.67	65.29
LinearProbing CLIP Radford et al. [2021]	16-shot	72.24	**96.43**	**92.34**	81.46	97.85	**86.22**	41.19	**75.99**	**71.34**	84.10	**85.33**	80.41
CLIP + LFA	16-shot	**72.83**	95.74	91.28	**83.55**	**97.93**	84.84	**45.96**	74.66	70.63	**87.05**	84.22	**80.79**
CoOp Zhou et al. [2022c]	16-shot	71.80	**95.50**	**91.70**	83.10	96.70	**84.20**	43.20	**74.50**	69.60	84.40	82.40	79.74
CoOp + LFA	16-shot	**72.22**	95.38	90.30	**85.00**	**97.40**	83.18	**49.20**	72.96	**71.57**	**88.10**	**83.00**	**80.76**
CLIP-Adapter Gao et al. [2021a]	16-shot	70.20	94.50	**91.80**	69.20	77.90	**86.70**	29.10	70.40	49.20	62.30	74.80	70.55
CLIP-Adapter + LFA	16-shot	**72.87**	**95.42**	91.20	**83.25**	**98.21**	84.91	**46.32**	**74.54**	**71.28**	**86.63**	**83.88**	**80.77**
MaPLe Khattak et al. [2023]	16-shot	70.62	95.46	**93.68**	73.88	93.79	**87.37**	37.56	**74.66**	66.37	87.17	80.31	78.26
MaPLe + LFA	16-shot	**72.66**	**96.19**	91.58	**85.47**	**97.72**	85.05	**49.44**	74.38	**73.94**	**90.65**	**85.49**	**82.05**

Table 2: Comparative classification accuracy of pre-trained models and models applying our methodology across 11 datasets in 16-shot. The best scores are bolded, and the second best scores are underlined at original CLIP model. As a results, all models utilizing our methodology outperform origin pre-trained models. Except for LinearProbing CLIP and CoOp, which show high score about 80% of the original, the remaining models demonstrate significant improvement.

For a neural network with L layers, we define the function f_l at each layer l as a product of weight matrices: $f_1 = W_1$, $f_2 = W_2 W_1$, and generally, $f_n = W_n W_{n-1} \dots W_1$ for $n = 1, \dots, L$. Therefore, basic default model is $\tilde{W} = \prod_{l=1}^{n} W_l$.

Theorem 1. ***[Rank Preservation of LFA]*** *Let W_l be a set of independent weight matrices for $l = 1, \dots, n$, and let C_l be arbitrary constant matrices. Define the simple LFA as: LFA$(W) = C_1 W_1 + C_2 W_2 W_1 + \cdots + C_n W_n \cdots W_1 = \sum_{l=1}^{n} C_l \left(\prod_{k=1}^{l} W_k \right)$ Then, the rank of the default model, denoted by $\tilde{W}$, is given by rank$(\tilde{W}) = \min($rank(W_1), rank$(W_2), \dots,$ rank$(W_n))$ Then, the rank of LFA(W) satisfies: rank(LFA$(W)) \geq$ rank$(\tilde{W})$*

3 Experiments and Results

3.1 Domain Generalization

For domain generalization evaluation, we designed an experiment where one domain is hidden during training and all domains are used during testing. We used 80% of the data for training and final evaluation, with 20% for hyperparameter tuning. In each test environment (cases 1 to 4), we conducted a random search with 5 trials over hyperparameter configurations, repeating the process 3 times. We selected the best hyperparameters from 60 cases using the training-domain validation method [Gulrajani and Lopez-Paz, 2020]. Table 1 shows the domain generalization (DG) performance across various backbone models. For a fair comparison, we limited the hyperparameter searches for DPLCLIP and our model to 5 trials due to time constraints. CLIP + LFA outperforms other models on the PACS and OfficeHome datasets and achieves the highest average performance.

3.2 Few-shot Image Classification

We evaluated LFA's impact on pre-trained CLIP-based models, including Zero-Shot CLIP, LinearProbing CLIP, CoOp, CLIP-Adapter, and MaPLe. In LinearProbing, only the last projection weight matrix is learned, with the rest of CLIP frozen. In a 16-shot environment, we compared the performance of these LFA-applied models with their original counterparts, as shown in Table 2. While comparing pre-trained models is challenging due to their diverse pre-training environments, it is evident that the average performance improves when our methodology is applied. Although it is not a dramatic performance improvement, it is noteworthy that applying a lightweight methodology that utilizes existing features can improve even for LinearProbing CLIP and CoOp models, which have basic performance close to 80%.

3.3 Learning Efficiency

To evaluate the model's computational and memory efficiency, we measured the peak memory usage during a single epoch and recorded the number of learnable parameters on the DomainBed benchmark. As shown in Table 3, both LFA and LinearProbing CLIP—methods where backpropagation does not extend through the pre-trained CLIP model—demonstrate superior learning efficiency, relying solely on the model's frozen features. Both models were implemented by extracting and reusing these frozen features. In contrast, the DPL method, which allows backpropagation through the pre-trained model, exhibits lower learning efficiency. Continuing our exploration, we designed a novel model termed "CLIP + LFA*" aimed at reducing the count of trainable parameters. The reduction is

Model		LinearProbing CLIP	CLIP + DPL	CLIP + QLoRA	CLIP + LFA
Accuracy Average (%)		72.0	72.7	72.7	73.9
Peak Memory (MB) (batch-size 32)	VLCS	1707	12763	8716	1757
	PACS	1707	13077	8717	1771
	OfficeHome	1707	19200	9840	1870
	TerraIncognita	1707	13301	8717	1770
	DomainNet	1745	46074*	21579	3089
Training Time (300 step)	VLCS	11m 40s	11m 46s	2m 4s	11m 32s
	PACS	2m 32s	10m 10s	1m 46s	2m 28s
	OfficeHome	5m 44s	10m 31s	1m 49s	5m 18s
	TerraIncognita	2m 50s	10m 31s	1m 51s	3m 1s
	DomainNet	4m 10s	10m 54s*	3m 47s	4m 7s
Inference Time	VLCS	49s	1m 9s	12s	49s
	PACS	21s	46s	7s	22s
	OfficeHome	1m 5s	1m 14s	16s	54s
	TerraIncognita	1m 55s	2m 9s	25s	2m 3s
	DomainNet	24m 22s	1h 4m 46s*	12m 4s	22m 33s

Table 3: Peak memory and learnable parameters for Domain Generalization training are shown. * indicates models with reduced parameters (Section 3.3). CLIP + LFA uses slightly more memory than LinearProbing CLIP but less than CLIP + DPL, while outperforming CLIP + DPL.

achieved by applying LoRA [Zhang et al., 2021] methodology to our attention mechanism. First, we initialize the weight matrices $\mathbf{W_Q}, \mathbf{W_K}, \mathbf{W_V}$ using the pretrained projection matrices from CLIP and freeze them. Next, we established a structure that incorporates the learnable update weight matrices $\triangle\mathbf{W_Q}, \triangle\mathbf{W_K}, \triangle\mathbf{W_V}$ by adding them to the initialized weight matrices. Each update weight matrix is composed of the matrix product of matrices $\mathbf{A}$ and $\mathbf{B}$, both of which have a low rank, with 32 being used as the rank. By adjusting the rank, the trade-off between learning performance and the number of learned parameters can be controlled.

3.4 Qualitative Result

Figure 3 shows a successful application example. The LinearProbing CLIP model incorrectly predicts the image of the elephant as a horse with a 39% probability, and utilizing only the last layer in the CLIP + LFA model leads to an incorrect prediction of a giraffe with a probability of approximately 78%. When aggregating features from all layers, the model correctly predicts an elephant with a 78% probability. Each model was trained without prior knowledge of the art painting domain in the PACS dataset. Above experimental results for art painting demonstrate that our method, which aggregates and utilizes features extracted from each layer, exhibits robust domain generalization performance.

		dog	elephant	giraffe	guitar	horse	house	person	predict
Linear Probing CLIP	last layer	0	0.12	0.38	0	0.39	0.03	0.08	horse
CLIP + LFA	last layer	0.02	0.1	0.78	0.08	0.01	0	0	giraffe
	aggregated	0.02	0.78	0.12	0.01	0.05	0.02	0.01	elephant

layer	1	2	3	4	5	6	7	8	9	10	11	12
text-cross	0.05	0.04	0.04	0.04	0.04	0.05	0.06	0.06	0.06	0.06	0.11	0.39
text-self	0.05	0.05	0.05	0.05	0.04	0.04	0.06	0.06	0.07	0.08	0.12	0.35
image-self	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.02	0.03	0.03	0.04	0.81

Figure 3: An example of applying LFA to the pre-trained CLIP model shows an image of an elephant from the PACS dataset misclassified as a horse by the LinearProbing CLIP model. When applying LFA, the highest activation is seen in the last layer. However, using only the last layer's features results in a misclassification as a giraffe. The correct prediction as an elephant is made only when all layers are utilized.

4 Conclusion and Limitation

Our study shows that leveraging logits from multiple layers, especially lower layers, enhances classification performance. The LFA method, using an attention mechanism, significantly improves Domain Generalization in the CLIP + LFA model. LFA's compatibility with CLIP-based models and its efficiency highlight its practical value, though its evaluation has so far been limited to CLIP-based models. Future work should explore its application to other models and tasks and optimize it for larger models and resource-constrained environments.

5 Acknowledgements

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT)(RS-2024-00457216, 20221A4A3033874).

References

Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In *Proceedings of the European conference on computer vision (ECCV)*, pages 456–473, 2018.

Junbum Cha, Kyungjae Lee, Sungrae Park, and Sanghyuk Chun. Domain generalization by mutual-information regularization with pre-trained models. In *European Conference on Computer Vision*, pages 440–457. Springer, 2022.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Chen Fang, Ye Xu, and Daniel N Rockmore. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1657–1664, 2013.

Ruili Feng, Kecheng Zheng, Yukun Huang, Deli Zhao, Michael Jordan, and Zheng-Jun Zha. Rank diminishing in deep neural networks. *Advances in Neural Information Processing Systems*, 35: 33054–33065, 2022.

Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. Clip-adapter: Better vision-language models with feature adapters. *arXiv preprint arXiv:2110.04544*, 2021a.

Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. Clip-adapter: Better vision-language models with feature adapters. *arXiv preprint arXiv:2110.04544*, 2021b.

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization, 2020.

Zixian Guo, Bowen Dong, Zhilong Ji, Jinfeng Bai, Yiwen Guo, and Wangmeng Zuo. Texts as images in prompt tuning for multi-label image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2808–2817, 2023.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 630–645. Springer, 2016.

Kenneth M. Hoffman and Ray Kunze. *Linear Algebra*. Prentice-Hall, Englewood Cliffs, NJ, second edition, 1971. ISBN 0135367972 9780135367971 0135368219 9780135368213.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021.

Muhammad Uzair Khattak, Hanoona Rasheed, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Maple: Multi-modal prompt learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19113–19122, 2023.

Muhammad Uzair khattak, Hanoona Rasheed, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Maple: Multi-modal prompt learning. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pages 5542–5550, 2017.

Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. What is being transferred in transfer learning? *Advances in neural information processing systems*, 33:512–523, 2020.

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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5018–5027, 2017.

Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pages 818–833. Springer, 2014.

Renrui Zhang, Wei Zhang, Rongyao Fang, Peng Gao, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. Tip-adapter: Training-free adaption of clip for few-shot classification. In *European Conference on Computer Vision*, pages 493–510. Springer, 2022.

Xin Zhang, Shixiang Shane Gu, Yutaka Matsuo, and Yusuke Iwasawa. Domain prompt learning for efficiently adapting clip to unseen domains. *arXiv preprint arXiv:2111.12853*, 2021.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16816–16825, 2022a.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision (IJCV)*, 2022b.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022c.

Supplementary Material for Efficient Transfer Learning driven by Layer-wise Features Aggregation

A Low-Level Feature Contribution

Previous research [LeCun et al., 2015, Zeiler and Fergus, 2014, He et al., 2016] primarily concentrated on the last layer and overlooked the utilization of features of previous layers, even though there was sufficient usable information remaining. We argue that low-level features are robust in various situations involving domain shifts. Because the features, which are local yet penetrate the essence, remain invariant on any changes.

Figure 4 shows the distribution of the number of correct answers on error cases, gathered by CLIP that utilizes only the last layer. This suggests lower level features contain components that can complement the features of the last layer.

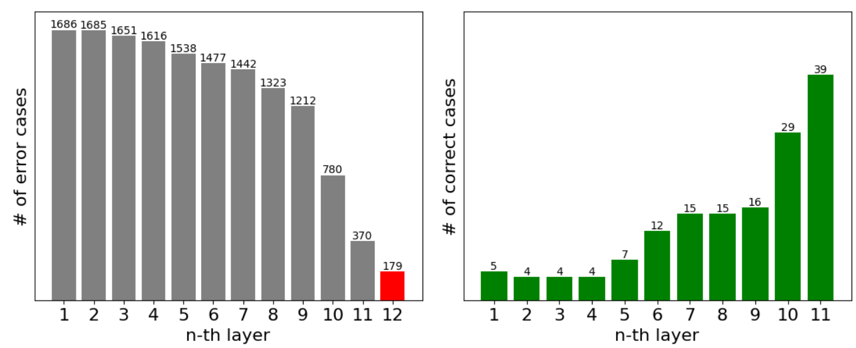

Figure 4: **DomainNet real bench, Left: Error Cases When Using Only Each Layer, Right: Correct Case Count in Other Layers for Error Cases in the 12th Layer.** In the Linear-probing CLIP study focusing on the last layer, **2∼22%** error cases were accurately predicted by the early layer. It suggests that lower layers effectively compensate for feature extraction deficiencies of the last layer.

B Enhancing Feature Representational Power Through Multi-Layer Integration

Let's denote the visual and textual encoder as f and g, with their respective inner layers as f_{l_1} and g_{l_2}, where $l_1 \in \{1, \dots, L^{(v)}\}, l_2 \in \{1, \dots, L^{(t)}\}$. $L^{(v)}$ and $L^{(t)}$ are the numbers of layers in each encoder. In the visual domain, for given an input image $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$, we can transform it into patch embedding $\mathbf{E}^{(v)} \in \mathbb{R}^{P \times d_h^{(v)}}$ [Dosovitskiy et al., 2020]. Next, we obtain the hidden representation $\mathbf{H}_{l_1}^{(v)} \in \mathbb{R}^{P \times d_h^{(v)}}$ from each layer, where P denotes the image patch's length and $d_h^{(v)}$ means the dimension of image hidden vectors. Among these hidden representations, the class token at index 0 encapsulates information that represents the entire image with each dimension d_h [Dosovitskiy et al., 2020]. Consequently, we select the hidden vector $\mathbf{h}_{l_1}^{(v)} \in \mathbb{R}^{d_h^{(v)}}$ using the CLS token, as shown in Equation 1. For the textual part, the text input consists of a prompt template "A photo of a label.", as utilized in CLIP [Radford et al., 2021]. Similar to the visual processing, we obtain a sequence embedding $\mathbf{E}^{(t)} \in \mathbb{R}^{S \times d_h^{(v)}}$ by transforming the tokenized input text $\mathbf{T} \in \mathbb{Z}^S$ and extract the hidden representation $\mathbf{H}_{l_2}^{(t)} \in \mathbb{R}^{S \times d_h^{(t)}}$ from each layer, where S denotes the fixed sequence's length. We use the information from the EOS (End Of Sentence) token, as it is typically utilized to accurately detect the end of a sequence and comprehend the context [Guo et al., 2023]. This enables us to derive the textual hidden feature vector $\mathbf{h}_{l_2}^{(t)} \in \mathbb{R}^{d_h^{(t)}}$, as shown in Equation 1.

$$
\begin{aligned}
\mathbf{H}_1^{(v)} &= f_1(\mathbf{E}^{(v)}), & \mathbf{H}_{l_1+1}^{(v)} &= f_{l_1+1}(\mathbf{H}_{l_1}^{(v)}) \\
\mathbf{H}_1^{(t)} &= g_1(\mathbf{E}^{(t)}), & \mathbf{H}_{l_2+1}^{(t)} &= g_{l_2+1}(\mathbf{H}_{l_2}^{(t)}) \\
\mathbf{h}_{l_1}^{(v)} &= \mathbf{H}_{l_1,\text{[CLS]}}^{(v)}, & \mathbf{h}_{l_2}^{(t)} &= \mathbf{H}_{l_2,\text{[EOS]}}^{(t)}
\end{aligned}
\tag{1}
$$

By incorporating diverse information from early layers' invariant information to later layers' semantic information, we can obtain a set of abundant feature vectors $\{\mathbf{h}_1^{(v)}, \ldots, \mathbf{h}_{L^{(v)}}^{(v)}, \mathbf{h}_1^{(t)}, \ldots, \mathbf{h}_{L^{(t)}}^{(t)}\}$ for effective downstream task solving.

C Detailed Process of Feature Extraction Using an Attention Mechanism

In this paper, an attention mechanism has been applied to effectively leverage the complementary characteristics of these hidden feature vectors.

First, to perform attention operations on hidden feature vectors for each modality, we define query, key, and value components. The query identifies the attention focus, the key measures alignment with input elements, and the value contains the information to be integrated into the output.

Secondly, in the visual encoder, we derive the query matrix $\mathbf{Q}^{(v)} \in \mathbb{R}^{d_q \times L^{(v)}}$, key matrix $\mathbf{K}^{(v)} \in \mathbb{R}^{d_k \times L^{(v)}}$ and value matrix $\mathbf{V}^{(v)} \in \mathbb{R}^{d_v \times L^{(v)}}$ from feature vector matrix $\mathbf{X}^{(v)} = [\mathbf{h}_1^{(v)}, \ldots, \mathbf{h}_{L^{(v)}}^{(v)}] \in \mathbb{R}^{d_h \times L^{(v)}}$. Each transformation is achieved through a linear operation using learnable weight matrices $\mathbf{W}_{\mathbf{Q}}^{(v)} \in \mathbb{R}^{d_q \times d_h}$, $\mathbf{W}_{\mathbf{K}}^{(v)} \in \mathbb{R}^{d_k \times d_h}$ and $\mathbf{W}_{\mathbf{V}}^{(v)} \in \mathbb{R}^{d_v \times d_h}$ where d_q is the dimension of the query vector space, d_k is the dimension of the key vector space, and d_v is the dimension of the value vector space, as shown in Equation 2. This procedure is then repeated in the textual encoder.

$$
\begin{aligned}
\mathbf{Q}^{(v)} &= [\mathbf{q}_1^{(v)}, \ldots, \mathbf{q}_{L^{(v)}}^{(v)}] = \mathbf{W}_{\mathbf{Q}}^{(v)} \times \mathbf{X}^{(v)} \\
\mathbf{K}^{(v)} &= [\mathbf{k}_1^{(v)}, \ldots, \mathbf{k}_{L^{(v)}}^{(v)}] = \mathbf{W}_{\mathbf{K}}^{(v)} \times \mathbf{X}^{(v)} \\
\mathbf{V}^{(v)} &= [\mathbf{v}_1^{(v)}, \ldots, \mathbf{v}_{L^{(v)}}^{(v)}] = \mathbf{W}_{\mathbf{V}}^{(v)} \times \mathbf{X}^{(v)}
\end{aligned}
\tag{2}
$$

In the original CLIP model, a projection weight was used to map data from both modalities into a shared space, enabling interpretable similarity measurements. To achieve the same goal during the use of attention operations, we initialized $\mathbf{W}_\mathbf{Q}, \mathbf{W}_\mathbf{K}, \mathbf{W}_\mathbf{V}$ with well pre-trained visual and textual projection weights of CLIP model as a guideline to the shared space.

Following that, we choose task-specific query vectors $\mathbf{q}_{L^{(v)}}^{(v)}$, $\mathbf{q}_{L^{(t)}}^{(t)} \in \mathbb{R}^{d_q}$ that utilize the final layer hidden vectors $\mathbf{h}_{L^{(v)}}^{(v)}, \mathbf{h}_{L^{(t)}}^{(t)}$ and then we employ them in our attention operations. The point is, during the attention process, a disparity in representational power emerges between the final layer outputs obtained from the two modalities. In the case of images, even when inputs belong to the same class, they can exhibit diverse forms, resulting in final layer outputs with sufficient discriminative capability. Conversely, for text, both the template and the class name remain consistent for objects within the same class, leading to a lack of discriminative power among in-class data. Consequently, the representational ability of $\mathbf{h}_{L^{(t)}}^{(t)}$ becomes relatively weaker compared to $\mathbf{h}_{L^{(v)}}^{(v)}$. As a result, we conduct separate self-attention operations using $\mathbf{q}_{L^{(v)}}^{(v)}$ and $\mathbf{q}_{L^{(t)}}^{(t)}$ for each component and we conduct a cross-attention operation employing both $\mathbf{q}_{L^{(t)}}^{(t)}$ and $\mathbf{q}_{L^{(v)}}^{(v)}$ for the textual component, compensating for the weak representation ability of the textual component. Furthermore, in the textual encoder, we generate two types of independent key and value matrices, namely $\mathbf{K}_{self}^{(t)}, \mathbf{K}_{cross}^{(t)}$, and $\mathbf{V}_{self}^{(t)}, \mathbf{V}_{cross}^{(t)}$. By utilizing these matrices separately, we create one textual component, $\mathbf{z}_{cross}^{(t)}$, that considers meaningful layers related to the visual features and another one, $\mathbf{z}_{self}^{(t)}$, that considers only from textual features itself. These two types of attention operations, computed in this manner, are ultimately merged in Section 2.3.

In visual encoder, using the obtained key and query, we compute the attention score vector $\mathbf{e}^{(v)} = \{e_{l_1}^{(v)}\} \in \mathbb{R}^{L^{(v)}}$. By ensuring that both vector spaces d_k and d_q have the same dimension denoted as D, we utilize the scaled dot product as the score function. These scores, falling within the range of $[-\infty, +\infty]$, are appropriately scaled using the softmax function $\sigma(\cdot)$ to serve as coefficients for a weighted sum. Finally, by using $\sigma(\mathbf{e}^{(v)})$ as coefficients for $\mathbf{V}^{(v)}$, we acquire the self-attention output vector $\mathbf{z}^{(v)} \in \mathbb{R}^D$. Repeating this step twice in the textual encoder yields self-attention and cross-attention output vectors $\mathbf{z}^{(t)}_{self}, \mathbf{z}^{(t)}_{cross} \in \mathbb{R}^D$ as shown in Equation 3.

$$\begin{aligned}
\mathbf{e}^{(v)} &= \frac{(\mathbf{q}^{(v)}_{L^{(v)}})^T \mathbf{K}^{(v)}}{\sqrt{d_h}}, \quad \mathbf{z}^{(v)} = \mathbf{V}^{(v)} \sigma(\mathbf{e}^{(v)}) \\
\mathbf{e}^{(t)}_{self} &= \frac{(\mathbf{q}^{(t)}_{L^{(t)}})^T \mathbf{K}^{(t)}_{self}}{\sqrt{d_h}}, \quad \mathbf{z}^{(t)}_{self} = \mathbf{V}^{(t)}_{self} \sigma(\mathbf{e}^{(t)}_{self}) \\
\mathbf{e}^{(t)}_{cross} &= \frac{(\mathbf{q}^{(t)}_{L^{(t)}})^T \mathbf{K}^{(t)}_{cross}}{\sqrt{d_h}}, \quad \mathbf{z}^{(t)}_{cross} = \mathbf{V}^{(t)}_{cross} \sigma(\mathbf{e}^{(t)}_{cross})
\end{aligned} \tag{3}$$

Due to the presence of two types of textual output vectors, it is necessary to merge them using an appropriate hyperparameter $\beta \in [0, 1]$. As a result, we obtain the final textual output vectors, as illustrated in Equation 4.

$$\mathbf{z}^{(t)} = \beta * \mathbf{z}^{(t)}_{cross} + (1 - \beta) * \mathbf{z}^{(t)}_{self} \tag{4}$$

D Merging Similarities and Predictions: A Detailed Approach

In each modality, the value vectors of the attention mechanism share the same space. Therefore, the output vectors $\mathbf{z}^{(v)}$ and $\mathbf{z}^{(t)}$ for each modality also share the same dimensional space, as explained in Section 2.2. This allows the computation of the output logits $\mathbf{z}$ between the two output vectors using dot-product, enabling a contrastive learning framework. We supposed that the input data consists of one image $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$ and tokenized template prompts $\mathbf{T}_i \in \mathbb{Z}^S$ for C classes, as shown in Equation 5.

$$\mathbf{z} = \{z_i\}, \quad z_i = \mathbf{z}^{(v)}_{\mathbf{I}} \cdot \mathbf{z}^{(t)}_{\mathbf{T}_i}, \quad i = 1, \ldots, C \tag{5}$$

Using the logits obtained in this way, prediction probabilities can be computed through the softmax function, enabling the learning of similarities between positive and negative pairs, as shown in Equation 6.

$$P(y = j|x) = \frac{\exp(z_j/\tau)}{\sum_{i=1}^{C} \exp(z_i/\tau)} \tag{6}$$

where z_i represents the final similarity value between the given input image and the text input representing the i-th class, and τ is the temperature controlling the scale of the distribution.

The LFA model conducts its learning using features extracted from the pre-trained CLIP backbone, with all the required learnable parameters located exclusively at the rear end of CLIP. This fine-tuning approach is computationally efficient, as it bypasses the need for the backpropagation process to pass through the CLIP backbone pipeline. Additionally, by aggregating features at different levels, it enhances its representational capacity, making it effective for various downstream tasks.

E Layer-wise Features of Text Encoder

We conducted experiments on dog images from the PACS dataset of the DomainBed benchmark. Figure 5 shows that we can extract local relational features using the lower layer of the text encoder. As a result, text encoders also can extract features from various perspectives for each layer. Therefore, when utilizing the encoder, activating the appropriate layer based on the specific task can be beneficial.

F Theoretical Analysis Proofs

For a neural network with L layers, we define the function f_i at each layer i as a product of weight matrices: $f_1 = W_1$, $f_2 = W_2 W_1$, and generally, $f_n = W_n W_{n-1} \ldots W_1$ for $n = 1, \ldots, L$.

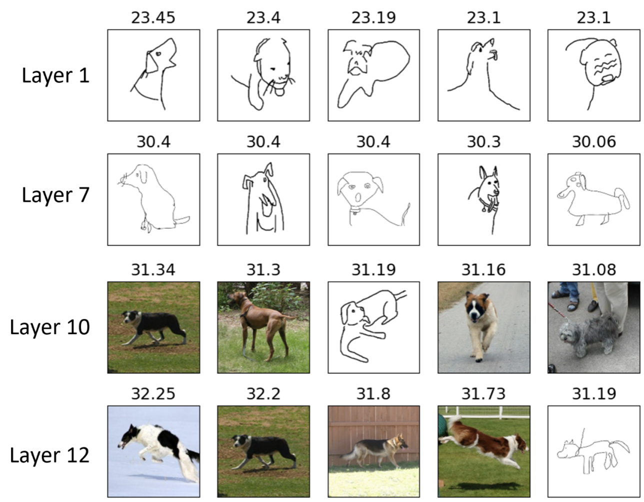

Figure 5: **Top-5 most similar images with the text 'a photo of a dog playing on a ground.', shown in the n-th layer of the text encoder.** The number above the image represents the similarity score between the last layer's image features and the n-th layer's text features. From the initial layer to 7th layer extracts basic features of 'a dog', whereas a more relational image, such as 'a dog on a ground', emerges at the 10th layer. At the last layers, top similar images include more complex relationships, exemplified by 'a photo of a dog playing on a ground'.

Theorem 2. ***[Principle of Rank Diminishing]*** *Let F be a neural network with L layers defined by $f_l = W_l W_{l-1} \dots W_1$ where $l = 1, \dots, L$. If each layer function is continuous and almost everywhere smooth, the rank of the sub-networks and the intrinsic dimension of feature manifolds monotonically decrease with depth:*

$$\mathit{rank}(W_1) \geq \mathit{rank}(W_2 W_1) \geq \dots \geq \mathit{rank}(f_n).$$
$$\mathit{rank}(W_1 x) \geq \mathit{rank}(W_2 W_1 x) \geq \dots \geq \mathit{rank}(f_n x).$$

Proof. Let F be a neural network with L layers defined by $f_i = W_i W_{i-1} \dots W_1$ where $i = 1, \dots, L$. If each layer function is continuous and almost everywhere smooth, the rank of the sub-networks and the intrinsic dimension of feature manifolds monotonically decrease with depth:

$$\text{rank}(W_1) \geq \text{rank}(W_2 W_1) \geq \dots \geq \text{rank}(f_n).$$
$$\text{rank}(W_1 x) \geq \text{rank}(W_2 W_1 x) \geq \dots \geq \text{rank}(f_n x).$$

Proof. The proof relies on the rank theorem [Hoffman and Kunze, 1971] for matrices:

$$\text{rank}(AB) = \text{rank}(B) - \dim(\ker(A) \cap \text{Im}(B)).$$

Given $\text{rank}(AB) = \text{rank}(B^T A^T)$ and $\text{rank}(A^T) = \text{rank}(A)$:

$$\begin{aligned} \text{rank}(AB) &= \text{rank}(B^T A^T) \\ &= \text{rank}(A^T) - \dim(\ker(B^T) \cap \text{Im}(A^T)). \end{aligned}$$

Thus, $\text{rank}(AB) \leq \text{rank}(B)$ and $\text{rank}(AB) \leq \text{rank}(A)$. □

□

Theorem 3. F.1 Theorem 1. [Rank Preservation of LFA]

Let W_l be a set of independent weight matrices for $l = 1, \dots, n$, and let C_l be arbitrary constant matrices. Define the simple LFA as:

$$\begin{aligned} \mathit{LFA}(W) &= C_1 W_1 + C_2 W_2 W_1 + \dots + C_n W_n \cdots W_1 \\ &= \sum_{l=1}^{n} C_l \left(\prod_{k=1}^{l} W_k \right). \end{aligned}$$

Then, the rank of the default model, denoted by $\tilde{W}$, is given by

$$\textit{rank}(\tilde{W}) = \min(\textit{rank}(W_1), \textit{rank}(W_2), \ldots, \textit{rank}(W_n))$$

Then, the rank of LFA(W) satisfies:

$$\textit{rank}(\textit{LFA}(W)) \geq \textit{rank}(\tilde{W}).$$

Proof. Let $r_k = \text{rank}(W_k)$ for $k = 1, \ldots, n$ and $R_k = \text{rank}(W_k \cdots W_1)$ for $k = 1, \ldots, n$.

1. First, we establish a relationship between R_k and r_k:

   $$R_k = \text{rank}(W_k \cdots W_1) = \min(\text{rank}(W_k), \ldots, \text{rank}(W_1)) = \min(r_k, r_{k-1}, \ldots, r_1).$$

   This follows from the property that the rank of the product of matrices cannot exceed the smallest rank among the multiplied matrices.

2. Next, we consider the rank of the LFA model, LFA(W):

   $$\begin{aligned} \text{rank}(\text{LFA}(W)) &= \text{rank}\left(\sum_{l=1}^{n} C_l \prod_{k=1}^{l} W_k\right). \\ &= \text{rank}(C_1 \prod_{k=1}^{1} W_k + C_2 \prod_{k=1}^{2} W_k + ... + C_n \prod_{k=1}^{n} W_k) \end{aligned}$$

   Also, the summation of the matrices is the union of the column spaces of each matrix [Hoffman and Kunze, 1971]. Then,

   $$\begin{aligned} \text{rank}(\text{LFA}(W)) &= \text{rank}(C_1 \prod_{k=1}^{1} W_k + C_2 \prod_{k=1}^{2} W_k + \ldots + C_n \prod_{k=1}^{n} W_k) \\ &= \text{rank}(\prod_{k=1}^{1} W_k + \prod_{k=1}^{2} W_k + \ldots + \prod_{k=1}^{n} W_k) \\ &\geq max(R_1, R_2, \ldots, R_n) \end{aligned}$$

3. We now focus on $\tilde{W}$ and its rank:

   $$\text{rank}(\tilde{W}) = \min(R_1, R_2, \ldots, R_n) = R_n.$$

   This follows from the theorem's definition of the rank of $\tilde{W}$ as the minimum rank among $W_1, W_2, \ldots, W_n$.

4. We combine these observations to conclude that:

   $$\begin{aligned} \text{rank}(\text{LFA}(W)) &\geq max(R_1, R_2, ..., R_n) \\ &\geq R_n \\ &= \text{rank}(\tilde{W}). \end{aligned}$$

Therefore, we have shown that $\text{rank}(\text{LFA}(W)) \geq \text{rank}(\tilde{W})$, which concludes the proof. $\square$

G Experimental Settings

CLIP Backbone

We applied our methodology to the pre-trained CLIP model, utilizing the ViT-B/16 as the CLIP backbone.

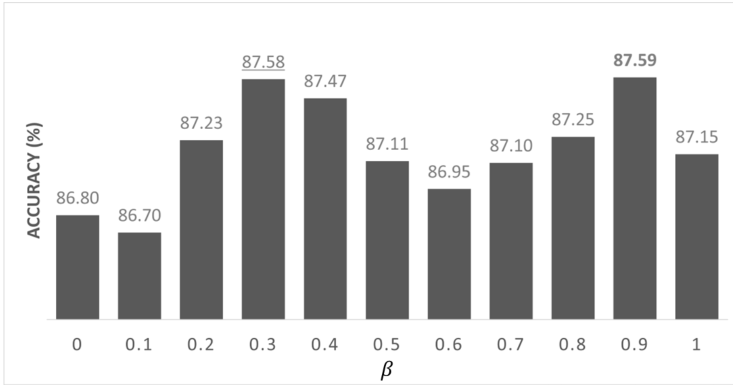

Figure 6: **Ablation study of hyperparameter β.** Table shows the performance of CLIP + LFA on Eurosat dataset, ranging from β values of 0.0 to 1.0 in increments of 0.1. The peak performance was observed at 0.9, followed closely by 0.3.

Domain Generalization

In the context of domain generalization experiments, we employed a set of benchmark datasets from DomainBed, which includes VLCS [Fang et al., 2013], PACS [Li et al., 2017], Office-Home [Venkateswara et al., 2017], and TerraIncognita [Beery et al., 2018]. VLCS comprises four photo datasets: Caltech101, LabelMe, SUN09, and VOC2007. PACS consists of four domains: Arts, Cartoons, Photos, and Sketches. OfficeHome encompasses four domains: Art, Clipart, Product, and Real. Finally, TerraIncognita, which records wild animals, is divided into four datasets based on filming locations: L100, L38, L43, and L46. As a prompt template for these datasets, we equally used 'a photo of a {class name}'.

We used the SGD optimizer, and the types of hyperparameters include batch size, learning rate, and momentum. The batch size is uniformly distributed between 2^2 and 2^5, the learning rate is uniformly distributed between $10^{-4.5}$ and $10^{-2.5}$, and the momentum is randomly sampled from 0.0, 0.1, and 0.2.

Few-show Image Classification

For few-shot experiments, we applied our methodology to CLIP [Radford et al., 2021], CoOp [Zhou et al., 2022b], CLIP-Adapter [Gao et al., 2021b], and MaPLe [khattak et al., 2023] models. Datasets employed were same to the ones used for existing models; ImageNet, Caltech101, OxfordPets, StanfordCars, Flowers102, Food101, FGVCAircraft, Sun397, DTD, Eurosat, and UCF101. When we apply LFA methodology to CLIP and CLIP-Adapter, we use the hand-crafted prompts [Radford et al., 2021] as the prompt templates (see H). Specifically, we utilized 'X X ... X {class name}.' for LFA + CoOp and 'a photo of a {class name}.' for LFA + MaPLe, which are identical to each prompt prefix used in original models. 'X' means the context word learning the appropriate prompt for a given data set.

We employed SGD optimizer with 0.1 momentum and 2e-3 learning rate. Additionally, we employed 40 epochs, 256 batch size for ImageNet, and employed 100 epochs, 32 batch size for other datasets, excluding Eurosat. The number of epochs for Eurosat was set to 200.

Ablation study on β

β is a hyperparameter that balances the weighting between the self-attention and cross-attention values of the text encoder. We set the value of beta to 0.9, where CLIP + LFA demonstrated the highest performance on the Eurosat dataset as depicted in Figure 6. We selected Eurosat dataset due to its lighter complexity compared to the other datasets in our study.

Learning Efficiency

In this experiment, we used a batch size of 32, selected the first domain to be masked during learning, and set the learning rate to 0.001.

H Hand-crafted Prompts

The pre-trained CLIP model was trained based on large-scale image-caption pairs obtained from the web. During this training, the accompanying text was presented in the form of natural sentences, including contextual information rather than isolated target words. This approach allowed the model to learn not only the target words but also the relationships between surrounding words. Similarly, when transfer learning on pre-trained models, utilizing natural sentences that align with the dataset context yields superior performance compared to using individual words from a single class. While the standard prompt template 'a photo of a class name.' is commonly employed, modifying it appropriately to match the dataset characteristics can enhance learning performance.

Following [Radford et al., 2021], We utilized appropriate hand-crafted prompts specific to each dataset as prompt templates for pre-trained CLIP, CLIP-Adapter, and MaPLe models; OxfordPets: 'a photo of a {class name}, a type of pet.', Flowers102: 'a photo of a {class name}, a type of flower.', FGVCAircraft: 'a photo of a {class name}, a type of aircraft.', DescribableTextures: '{class name} texture.', EuroSAT: 'a centered satellite photo of {class name}.', StanfordCars: 'a photo of a {class name}.', Food101: 'a photo of {class name}, a type of food.', SUN397: 'a photo of a {class name}.', Caltech101: 'a photo of a {class name}.', UCF101: 'a photo of a person doing {class name}.', and ImageNet: 'a photo of a {class name}.'.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS paper checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Yes, the main claims in the abstract and introduction accurately reflect the paper's contributions and scope in Section 1: Introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: Yes, the paper discusses the limitations of the authors' work in Section 5: Limitation.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Yes, the paper includes all assumptions and complete proofs for each theoretical result in Sectionn 3: Methodology.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Yes, Section 3: Experiments and Results fully discloses all information needed to reproduce the main experimental results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Yes, the paper provides open access to the data and code in the Supplementary Material section, with sufficient instructions for reproduction.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Yes, all training and test details are clarified in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [NA]

   Justification: N/A, this question is not applicable to this paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Yes, all necessary information on computer resources is clarified in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, this paper fully adheres to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: This question is N/A, as the paper does not address societal impacts within its scope.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

Justification: This question is N/A, as the paper does not involve the release of data or models with a high risk for misuse.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Yes, the creators of all assets used in the paper are properly credited, and the license and terms of use are respected in reference section.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Yes, all new assets are well documented, and the code access is provided in the abstract section.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: N/A, this paper does not involve crowdsourcing experiments or research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: N/A, this paper does not involve study participants or require IRB approval.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.