Finetune like you pretrain: Improved finetuning of zero-shot vision models

Goyal, S., Kumar, A., Garg, S., Kolter, Z., & Raghunathan, A. (2023). Finetune like you pretrain: Improved finetuning of zero-shot vision models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 19338-19347).

Introduction

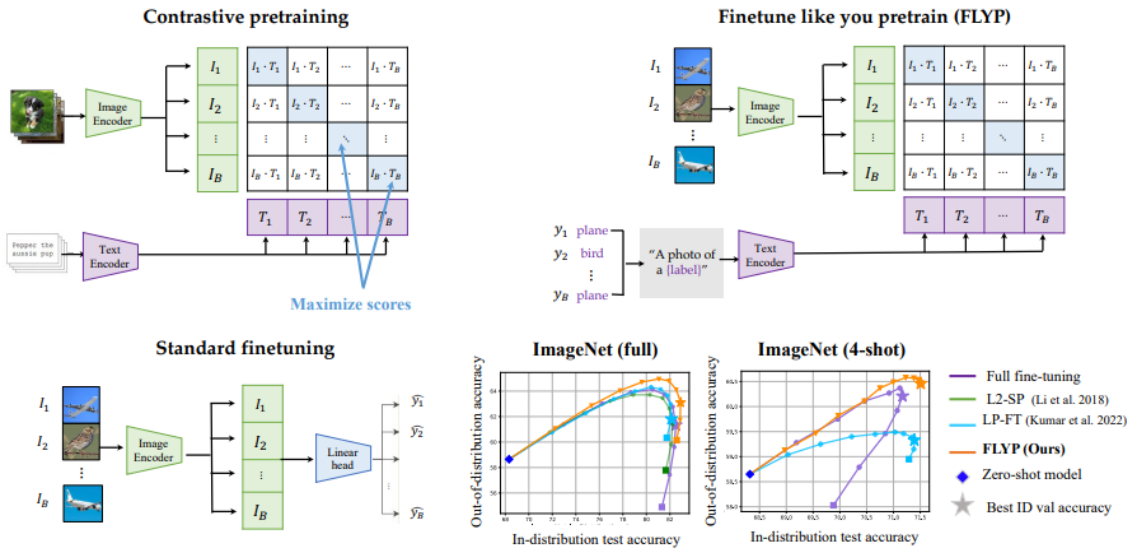
- 최근 CLIP[1]과 ALIGN[2]과 같은 image와 text를 동시에 사용해 분류를 진행하는 VLLM(Vision Language Large Model)이 개발됨
  - 해당 모델들은 image와 text 사이의 joint embedding을 찾는 방식으로 contrastive loss를 통해 학습
  - Test image에 대해, class 정보가 포함된 간단한 prompt들의 embedding에 대해 가장 similarity가 높은 것을 예측
- 이러한 VLLM들은 추가 학습이 필요하지 않는 Zero-shot Classification에 대해 좋은 성능을 보이며 distribution shift에 대해 강건한 모습을 보임
- 하지만 많은 상황에서 사용자의 TPO에 따라 모델을 fine-tunning 시켜 Zero-shot 보다 더 좋은 성능을 내는 것을 목적으로 하고 있음

Introduction

- 기존의 fine-tunning 방식들은 이미지 인코더의 파라미터들을 재학습시키지만, 이러한 방식은 distribution shift에 대해 취약하다는 단점을 가지고 있음
- 이러한 단점을 극복하기 위해 two-stage process of linear probing[3] 과 fine-tuned의 weight를 ensembling[4] 하는 방법들이 제안됨
  - 해당 방법들은 cross-entropy 방식에 추가적인 프로세스를 더하여 모델을 fine-tunning 진행
- 이런 방법들은 처음 학습시킨 방법과 다른 방식이기에 적합한 방법이라 할 수 없으므로 처음 학습 방법처럼 새로운 이미지들과 class 정보가 들어간 prompt로

  “same pretraining (contrastive) loss”

  fine-tunning 진행



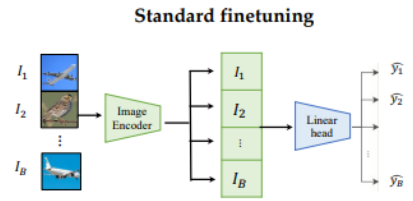
Finetuning pretrained VLLM models

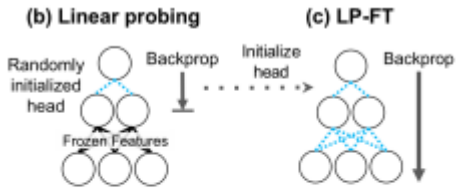
- 분류 관련 사용자의 downstream에 대해 VLLM 모델을 사용하는 방식은 다음과 같이 5가지 방법이 존재

1. Zero-shot (ZS) : 학습된 모델을 그대로 이용하며, 분류하고자 하는 이미지와 여러 ”a photo of a class_name .” 프롬프트에 대해 유사도가 가장 높은 프롬프트에 해당된 class 로 분류 진행
2. Linear probing (LP) : freeze된 image embedding 부분 위에 linear classifier를 추가로 쌓아 CE loss를 이용해 추가된 linear classifier를 fine-tunning
3. Full fine-tuning (FFT)[4] : image embedding 부분 위에 추가로 쌓은 linear classifier를 fine-tunning
4. LP-FT[3] : 먼저 LP 방식으로 학습한 다음, FFT 방식으로 학습 진행
5. Weight-ensembling[4] : ensemble the weights by linearly interpolating 진행

   fine-tuned model의 weight와 pre-trained model의 weight를 알파만큼의 비율로 조율

$$\theta_{\text{we}} = \alpha\theta'_{\text{img}} + (1-\alpha)\theta_{\text{img}}, \text{ where } \alpha \in [0,1]$$





82.9%	85.7%
66.2%	68.9%

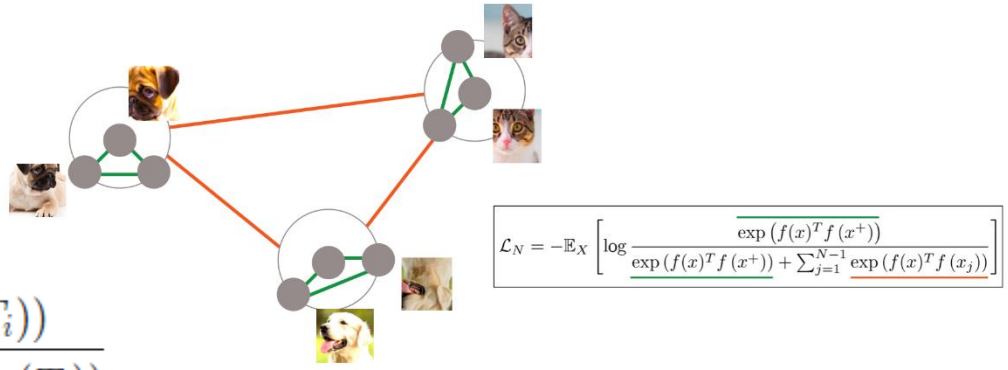
Contrastive pre-training with language supervision

- CLIP은 image embedding f(I_i)와 text embedding g(T_i)를 align하는 것을 목표
- 데이터셋(D)의 배치 사이즈(B) 만큼의 (이미지,텍스트) 쌍에 대하여 각 배치마다 아래의 식처럼 loss를 계산 후 이미지와 텍스트 인코더의 파라미터 θ를 업데이트 진행
  - 이러한 loss 식은 positive sample 끼리는 가깝게, negative sample 끼리는 멀게 representation을 정의하기 위한 InfoNCE Loss를 최소화 하는 방식
  - 즉, 서로 같은 (이미지, 텍스트) 쌍 끼리는 Positive sample로 거리가 가깝고, 다른 쌍 끼리는 negative sample로 거리가 먼 형태의 representation이 형성

$$\mathcal{L}_{\text{pre}}(D,\theta) := \sum_{i=1}^{B} -\log \frac{\exp(\bar{f}(I_i)\cdot\bar{g}(T_i))}{\sum_{j=1}^{B}\exp(\bar{f}(I_i)\cdot\bar{g}(T_j))} + \sum_{i=1}^{B} -\log \frac{\exp(\bar{f}(I_i)\cdot\bar{g}(T_i))}{\sum_{j=1}^{B}\exp(\bar{f}(I_j)\cdot\bar{g}(T_i))}$$

이미지 I_i에 대한 loss 합산

텍스트 T_i에 대한 loss 합산



$$\mathcal{L}_N = -\mathbb{E}_X\left[\log \frac{\exp(f(x)^T f(x^+))}{\exp(f(x)^T f(x^+)) + \sum_{j=1}^{N-1}\exp(f(x)^T f(x_j))}\right]$$

$$\underbrace{\text{score}(f(x), f(x^+))}_{\text{Positive samples}} >> \underbrace{\text{score}(f(x), f(x^-))}_{\text{Negative samples}}$$

FLYP: Finetune like you pretrain

- 논문에서 제안하는 FLYP도 똑같은 방식으로 fine-tunning 진행
- 새로운 (이미지,텍스트) 쌍에 대해 매 배치마다 Cross entropy loss가 아닌 contrastive loss를 이용해 fine-tunning 진행
- 제안하는 방법은 사전에 연구된 다른 방법[3-4]과 다르게
  1. Text 인코더 또한 업데이트 진행
  2. Linear Head를 사용하지 않아도 좋은 결과를 보임

$$\mathcal{L}_{\text{pre}}(D,\theta) := \sum_{i=1}^{B} -\log \frac{\exp(\bar{f}(I_i)\cdot\bar{g}(T_i))}{\sum_{j=1}^{B}\exp(\bar{f}(I_i)\cdot\bar{g}(T_j))} + \sum_{i=1}^{B} -\log \frac{\exp(\bar{f}(I_i)\cdot\bar{g}(T_i))}{\sum_{j=1}^{B}\exp(\bar{f}(I_j)\cdot\bar{g}(T_i))}$$

이미지 I_i에 대한 loss 합산

텍스트 T_i에 대한 loss 합산

Algorithm 1 FLYP : Contrastive Finetuning (One batch)

Given: Pretrained parameters θ_{img} and θ_{text},
Labeled batch $D = \{(x_1, y_1), \ldots (x_B, y_B)\}$,
Distribution over text descriptions $P_{\text{text}}(\cdot \mid y), y \in \mathcal{Y}$
Learning rate α

Training step:

1. Create text/image paired data via labels

$$D' = \{(I_1, T_1), \ldots (I_B, T_B)\}, \text{ where } T_i \sim P_{\text{text}}(\cdot \mid y_i)$$

2. Update parameters via contrastive loss

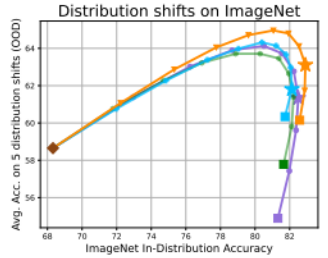
$$\theta := \theta - \alpha \nabla \mathcal{L}_{\text{pre}}(D', \theta) \text{ (Equation 1).}$$

Experiments / Evaluation Under Distribution Shifts

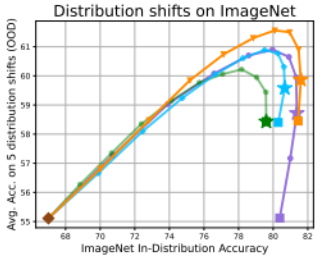
- 실험은 제로샷 및 각각의 파인튜닝 방법론으로 학습한 모델 단독 & Weight-ensembling[4] 을 사용해 ID 및 OOD 상황에 대해 검증 실험 진행
- 실험 결과, 표와 같이 ID 및 OOD 상황에서 모두 좋은 결과를 보이고 있음
- 꺾은선 그래프는 알파에 대한 ID 및 OOD에서의 성능 변화를 그린 그래프
  - 실험 결과, 최적의 ID 검증 정확도(별표가 있는 부분)와 비교했을 때, FLYP는 현재 최고의 성능을 내는 LP-FT보다 평균 1.3% OOD와 1.1% ID에서, WiseFT보다 평균 2% OOD와 1.6% ID에서 더 높은 성능을 보임

Methods	Imagenet				iWILDCam				FMoW	
	Without Ensembling		With Ensembling		Without Ensembling		With Ensembling		Without Ensembling	
	ID	OOD	ID	OOD	ID	OOD	ID	OOD	ID	OOD
Zeroshot	68.3 (-)	58.7 (-)	68.3 (-)	58.7 (-)	8.7 (-)	11.02 (-)	8.7 (-)	11.02 (-)	20.4 (-)	18.66 (-)
LP	79.9 (0.0)	57.2 (0.0)	80.0 (0.0)	58.3 (0.0)	44.5 (0.6)	31.1 (0.4)	45.5 (0.6)	31.7 (0.4)	48.2 (0.1)	30.5 (0.3)
FT	81.4 (0.1)	54.8 (0.1)	82.5 (0.1)	61.3 (0.1)	48.1 (0.5)	35.0 (0.5)	48.1 (0.5)	35.0 (0.5)	68.5 (0.1)	39.2 (0.7)
L2-SP	81.6 (0.1)	57.9 (0.1)	82.2 (0.1)	58.9 (0.1)	48.6 (0.4)	35.3 (0.3)	48.6 (0.4)	35.3 (0.3)	68.6 (0.1)	39.4 (0.6)
LP-FT	81.8 (0.1)	**60.5 (0.1)**	82.1 (0.1)	61.8 (0.1)	49.7 (0.5)	34.7 (0.4)	50.2 (0.5)	35.7 (0.4)	68.4 (0.2)	40.4 (1.0)
FLYP	**82.6 (0.0)**	60.2 (0.1)	**82.9 (0.0)**	**63.2 (0.1)**	**52.2 (0.6)**	**35.6 (1.2)**	**52.5 (0.6)**	**37.1 (1.2)**	**68.6 (0.2)**	**41.3 (0.8)**

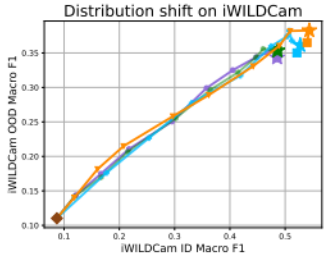
Table 1: FLYP outperforms the baselines both with and without ensembling. When ensembling, we choose the mixing coefficient (Equation 2) with the highest ID validation accuracy. For ImageNet, we report the mean OOD accuracy on 5 associated distribution shifts, and share individual numbers in appendix. FLYP outperforms all the baselines in 9 out of 10 various experiment settings. Without weight ensembling, averaged over all the datasets, FLYP outperforms full finetuning by 4.24% OOD and 1.8% ID. Similarly, FLYP outperforms LP-FT by 1.2% ID and gives similar OOD performance averaged over all the datasets.



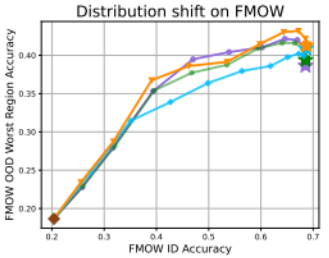
(a) OpenAI CLIP ViT-B/16 finetuned on ImageNet



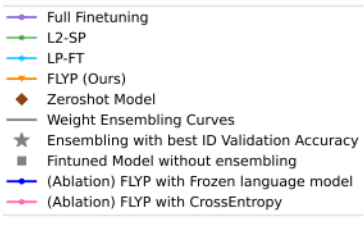
(b) LAION-400M CLIP ViT-B/16, finetuned on ImageNet



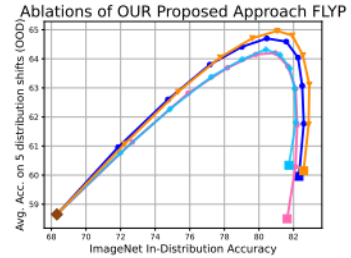
(c) CLIP ViT-B/16 finetuned on WILDS-iWILDCam



(d) CLIP ViT-B/16 finetuned on WILDS-FMoW



(e) Legend for all the subplots



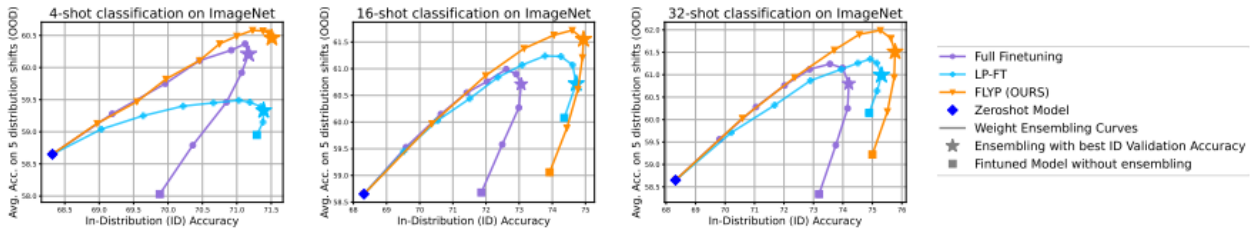
(f) Ablation of FLYP , details in Section 5

Experiments / Few-shot binary classification

- few-shot binary classification 또한 제안하는 방법이 가장 좋은 성능을 보임
- Image Net에 대해 Few shot 실험 결과, 꺾은선 그래프 또한 모든 알파에 대해 weight ensemble 결과가 가장 좋은 성능을 보임

	PatchCamelyon			SST2		
k (shots)	4	16	32	4	16	32
Zeroshot	56.5 (-)	56.5 (-)	56.5 (-)	60.5 (-)	60.5 (-)	60.5 (-)
LP	60.4 (4.0)	64.4 (3.7)	67.0 (4.4)	60.8 (1.8)	61.9 (1.4)	62.9 (1.3)
FT	63.1 (5.5)	71.6 (4.6)	75.2 (3.7)	61.1 (0.7)	62.4 (1.6)	63.4 (1.9)
LP-FT	62.7 (5.3)	69.8 (5.3)	73.9 (4.6)	60.9 (2.4)	62.9 (1.9)	63.6 (1.4)
FLYP	**66.9 (5.0)**	**74.5 (2.0)**	**76.4 (2.4)**	**61.3 (2.7)**	**65.6 (2.1)**	**68.0 (1.7)**

Table 3: In binary few-shot classification, FLYP performs remarkably well. For example, FLYP outperforms LP-FT by 4.4% and full finetuning by 4.6% in 32-shot classification on SST2. We observe similar gains when using a much larger CLIP architecture of ViT-L/14, as detailed in Appendix A.



(a) 4-shot classification (b) 16-shot classification (c) 32-shot classification (d) Legend

Figure 3: We evaluate FLYP on few-shot classification on ImageNet, where it outperforms all the baselines with weight ensembling, giving gains of 1.5% in 4-shot classification and 0.8% in 16-shot classification over LP-FT.

Experiments / Transfer Learning

- Transfer Learning 또한 제안하는 방법이 가장 잘됨

Methods	PCAM	CalTech	Cars	Flowers	ImageNet	iWILD
Zeroshot	56.49 (-)	87.7 (-)	64.4 (-)	71.2 (-)	68.3 (-)	8.7 (-)
LP	82.6 (0.1)	94.8 (0.0)	83.1 (0.0)	95.9 (0.0)	79.9 (0.0)	44.5 (0.6)
FT	89.1 (1.3)	97.2 (0.1)	84.4 (0.3)	90.4 (0.5)	81.4 (0.1)	48.1 (0.5)
LP-FT	89.0 (0.6)	96.9 (0.6)	89.4 (0.1)	**97.9 (0.1)**	81.8 (0.1)	49.7 (0.5)
FLYP	**90.3 (0.3)**	**97.6 (0.1)**	**89.6 (0.3)**	97.7 (0.1)	**82.6 (0.0)**	**52.2 (0.6)**

Table 4: We evaluate our proposed approach FLYP on 6 transfer learning datasets. FLYP gives *consistently* strong empirical performance, outperforming the baselines on 5/6 datasets considered.

Why does FLYP improve performance?

- 논문의 저자들은 FLYP fine-tuning 방법이 잘되는 이유를 exactly matches the pretraining이기 때문이라고 주장
  1. Class Collisions (batch 내에 같은 class가 존재하는 것)이 성능에 거의 영향을 주지 않음
  2. FLYP는 이미지 인코더 뿐만 아니라 텍스트 인코더의 파라미터를 학습
  3. Cross-Entropy loss 대신에 Contrastive-loss를 사용
  4. prompts를 sampling하여 추가적인 stochasticity를 fine-tuning process에 반영

결론 : 기존에 학습하듯이 contrastive pre-training 방법을 모방하여 fine-tunning 하는 것이 이미지 인코더를 fine-tunning하는 다른 방법들에 비해 일관적으로 좋은 성능을 보임

Reference

1. Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., ... & Sutskever, I. (2021, July). Learning transferable visual models from natural language supervision. In International conference on machine learning (pp. 8748-8763). PMLR.
2. Jia, C., Yang, Y., Xia, Y., Chen, Y. T., Parekh, Z., Pham, H., ... & Duerig, T. (2021, July). Scaling up visual and vision-language representation learning with noisy text supervision. In International conference on machine learning (pp. 4904-4916). PMLR.
3. Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. In International Conference on Learning Representations (ICLR), 2022c. URL https://openreview.net/forum?id=UYneFzXSJWh.
4. Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In International Conference on Machine Learning (ICML), 2022.