Classification with Foundation Models

Presented by Minho Park

KAIST

Table of Contents

Data-scares setting과 SOTA classification으로 나누어서

CLIP 위주로 CLIP의 몇 가지 좋은 특성을 알아보고, 해당 특징들을 잘 활용한 연구들을 살펴볼 것입니다.

Session 1:
Data-scares Classification Methods

Zero-shot and few-shot classification

Learning Transferable Visual Models From Natural Language Supervision

Alec Radford*, Jong Wook Kim* et al.

OpenAI

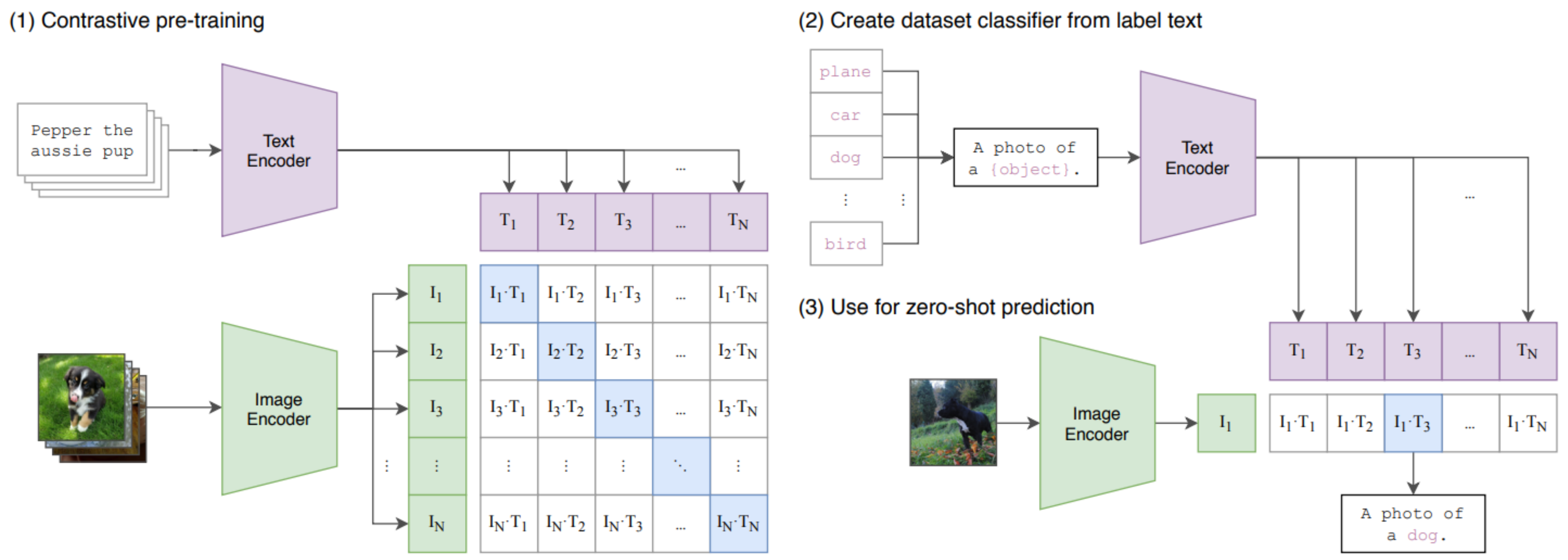
ICML 2021

Recap: Three Contributions of CLIP

- **Natural Language Supervision:** 자연어에 존재하는 supervision을 잘 활용해보자.
  - 직접 labeling 해야하는 classification과 다르게 image-text pair를 crawling하는 것은 large-scale dataset을 구축하기에 용이함.
  - 자연어의 특징을 사용하여 zero-shot으로 다른 task로 transfer하는 것이 가능함.
- **Creating a sufficiently large dataset:** OpenAI는 private WebImageText dataset (400M)을 구축함.
- **Efficient Pre-training Method:** Image-text contrastive learning 방법론 제안.

Zero-shot Classification with CLIP

- **C**ontrastive **L**anguage-**I**mage **P**re-training (CLIP)

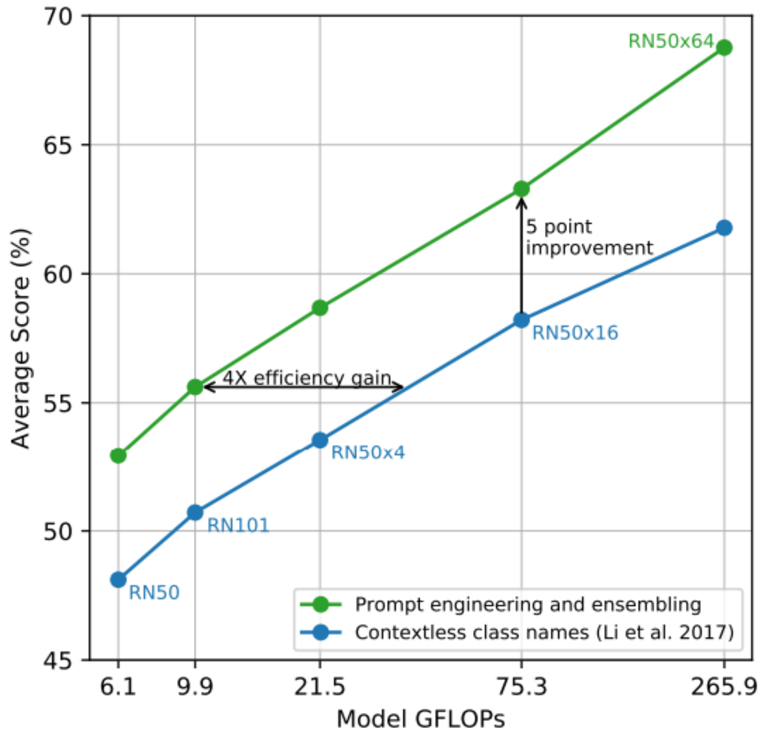


(1) Contrastive pre-training
Pepper the aussie pup
Text Encoder
Image Encoder
T_1 T_2 T_3 ... T_N
I_1 I_2 I_3 ... I_N
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
$I_2 \cdot T_1$ $I_2 \cdot T_2$ $I_2 \cdot T_3$ ... $I_2 \cdot T_N$
$I_3 \cdot T_1$ $I_3 \cdot T_2$ $I_3 \cdot T_3$ ... $I_3 \cdot T_N$
$I_N \cdot T_1$ $I_N \cdot T_2$ $I_N \cdot T_3$ ... $I_N \cdot T_N$
(2) Create dataset classifier from label text
plane
car
dog
bird
A photo of a {object}.
Text Encoder
(3) Use for zero-shot prediction
Image Encoder
I_1
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
A photo of a dog.

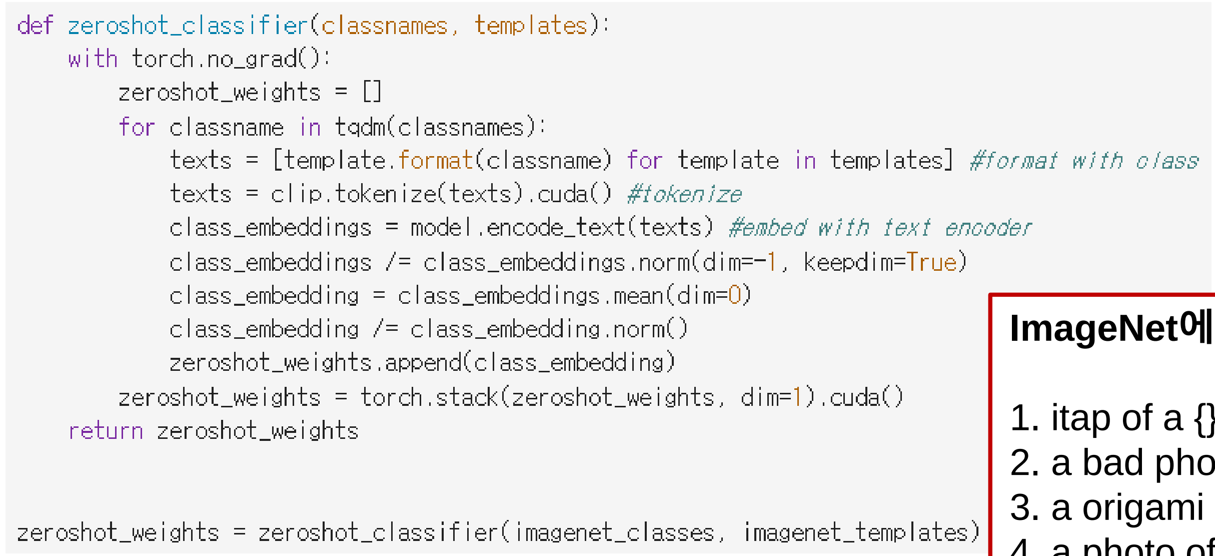
Figure 1. Summary of our approach. While standard image models jointly train an image feature extractor and a linear classifier to predict some label, CLIP jointly trains an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples. At test time the learned text encoder synthesizes a zero-shot linear classifier by embedding the names or descriptions of the target dataset's classes.

Prompt Engineering

- Zero-shot transfer를 진행할 때, prompt engineering을 통해 큰 성능 향상을 얻을 수 있음.
    - 기본은 “a photo of a {label}”



Prompt engineering and ensembling improve zero-shot performance.



```
def zeroshot_classifier(classnames, templates):
    with torch.no_grad():
        zeroshot_weights = []
        for classname in tqdm(classnames):
            texts = [template.format(classname) for template in templates] #format with class
            texts = clip.tokenize(texts).cuda() #tokenize
            class_embeddings = model.encode_text(texts) #embed with text encoder
            class_embeddings /= class_embeddings.norm(dim=-1, keepdim=True)
            class_embedding = class_embeddings.mean(dim=0)
            class_embedding /= class_embedding.norm()
            zeroshot_weights.append(class_embedding)
        zeroshot_weights = torch.stack(zeroshot_weights, dim=1).cuda()
    return zeroshot_weights


zeroshot_weights = zeroshot_classifier(imagenet_classes, imagenet_templates)
```

ImageNet에서 사용된 templates

1. itap of a {}.
2. a bad photo of the {}.
3. a origami {}.
4. a photo of the large {}.
5. a {} in a video game.
6. art of the {}.
7. a photo of the small {}.

* ITAP: I Take A Picture

OpenAI CLIP official implementation: https://github.com/openai/CLIP
IN1k prompt engineering: https://github.com/openai/CLIP/blob/main/notebooks/Prompt_Engineering_for_ImageNet.ipynb

Linear Probing with CLIP

- 일반적인 classification model들과 다르게, zero-shot CLIP을 few-shot에 대해서 linear probing했을 때, 성능이 오히려 하락한다.

- **Linear probing을 진행해도 zero-shot 보다 성능이 오르는 연구.**
  - CLIP-Adapter
  - Tip-Adapter

- **Zero-shot의 성능을 더 올리는 방법론.**
  - Synthetic data를 활용함.
  - Is Synthetic Data
  - CaFo

데이터를 더 줬는데 성능이 하락함.
⇒ 적절한 구조가 필요

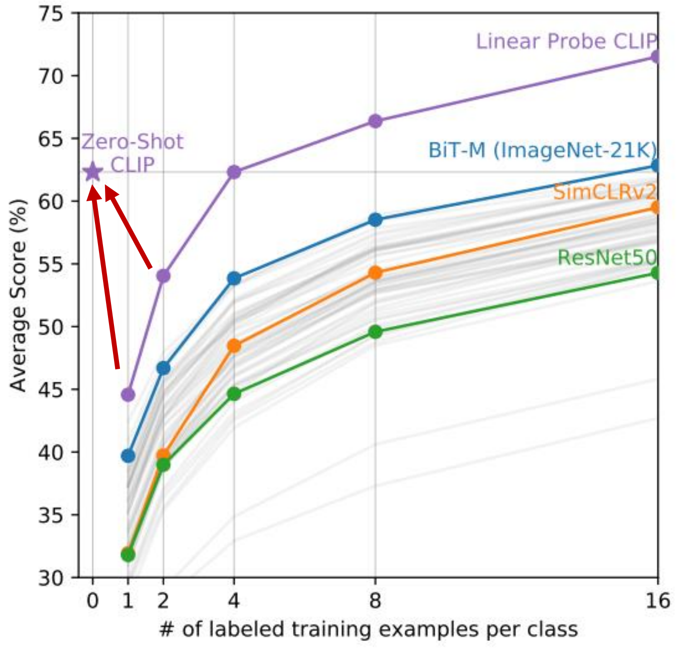
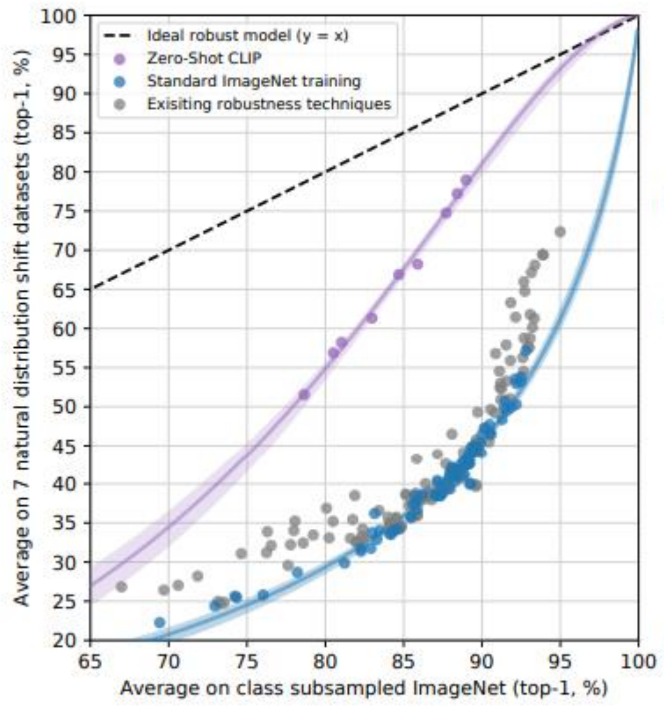
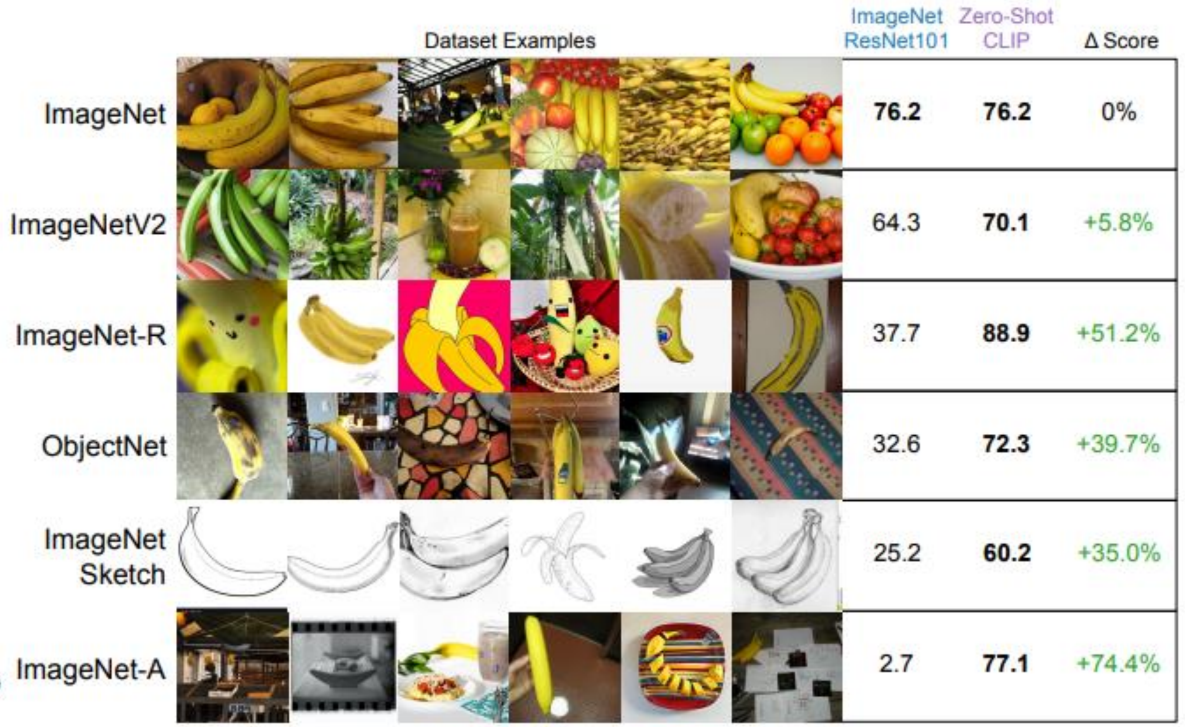


Figure 6. **Zero-shot CLIP outperforms few-shot linear probes.** Zero-shot CLIP matches the average performance of a 4-shot linear classifier trained on the same feature space and nearly matches the best results of a 16-shot linear classifier across publicly available models. For both BiT-M and SimCLRv2, the best performing model is highlighted. Light gray lines are other models in the eval suite. The 20 datasets with at least 16 examples per class were used in this analysis.

Robustness of CLIP

- **Robustness: input distribution에 대한 robustness**로 같은 label space를 가지는 다른 domain의 이미지들을 분류하는 성능으로 robustness를 측정한다. E.g., ImageNetV2, ImageNet-Sketch, etc.
- CLIP은 대규모 데이터셋으로 학습했기 때문에 robustness가 다른 classification 모델들에 비해 확연히 좋은 것을 알 수 있다.
- **Session 2. WiSE-FT:** Robustness를 보존하면서 fine-tuning하는 방법론을 제안.





	Dataset Examples	ImageNet ResNet101	Zero-Shot CLIP	Δ Score
ImageNet		76.2	76.2	0%
ImageNetV2		64.3	70.1	+5.8%
ImageNet-R		37.7	88.9	+51.2%
ObjectNet		32.6	72.3	+39.7%
ImageNet Sketch		25.2	60.2	+35.0%
ImageNet-A		2.7	77.1	+74.4%

Few-shot Classification

Tip-Adapter: Training-free CLIP-Adapter for Better Vision-Language Modeling

Renrui Zhang*, Rongyao Fang*, Wei Zhang* et al.

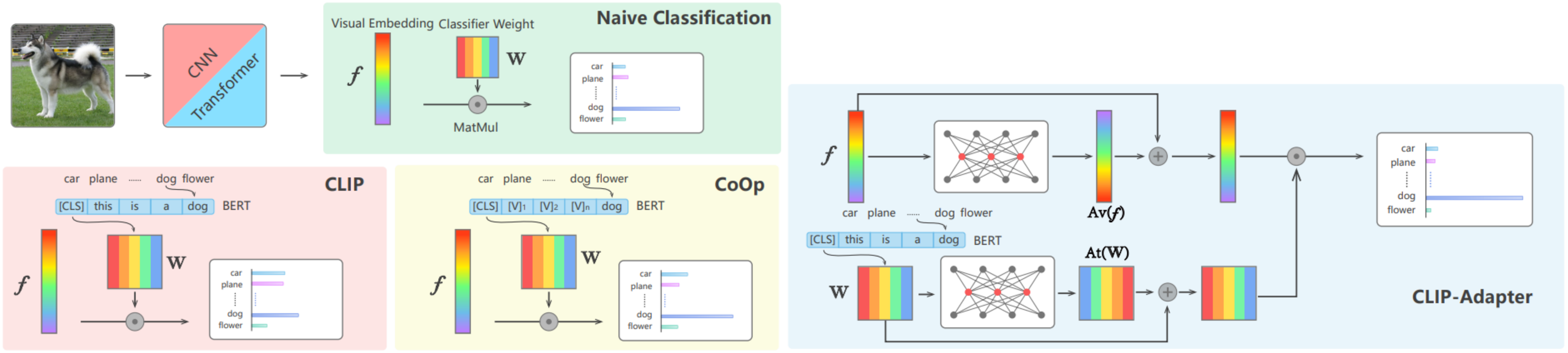
Shanghai AI Laboratory,

The Chinese University of Hong Kong, SenseTime Research

ECCV 2022

Related Work: CoOp, CLIP-Adapter

- CLIP의 zero-shot 성능을 보존하자.
  - CoOp: CLIP의 parameter는 모두 그대로 두는 상태에서 prompt에 learnable vector를 넣는 방식으로 원하는 데이터셋에 adaptation을 진행함.
  - CLIP-Adapter: CLIP의 zero-shot logit 값을 유지하면서, adapter라고 불리는 새로운 learnable parameter만 학습하도록 구조하였음.



Comparison of different visual classification architectures.

Zhou, Kaiyang, et al. "Learning to prompt for vision-language models." *IJCV.* 2022.

Gao, Peng, et al. "Clip-adapter: Better vision-language models with feature adapters." *arXiv.* 2021.

Training-free CLIP-Adapter

- Adapter를 few-shot data를 가지고 학습시키는 것은:
  - CLIP 전체를 fine-tuning 하는 것보다 안정적이고,
  - Linear probing보다 성능이 높지만,
  - 여전히 training cost를 필요로 한다.
- **Tip-Adapter는 training-free로 CLIP-Adapter를 구성한다.**
- 또한 training cost를 사용하면 CLIP-Adapter보다 성능이 향상되는 구조를 가진다.

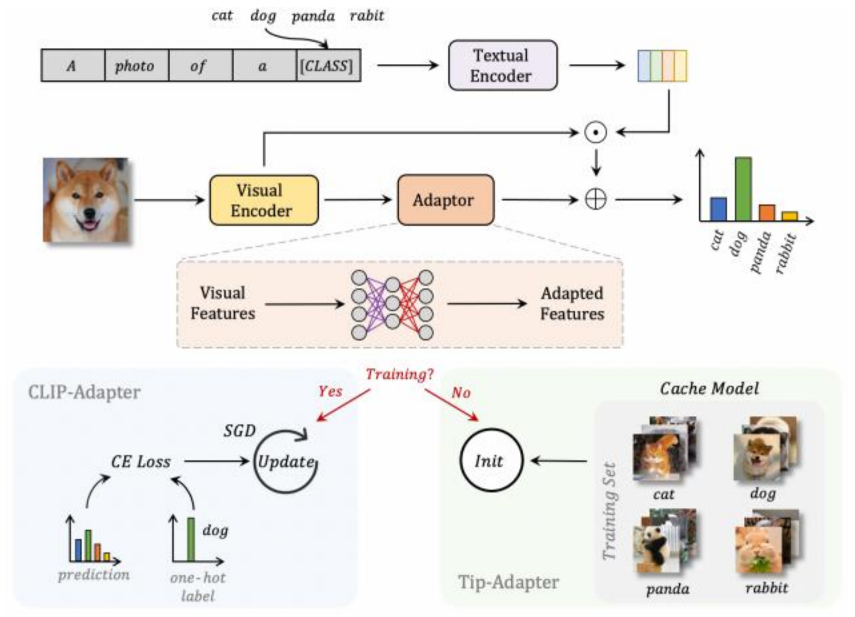


Figure 1. **A Comparison of CLIP-Adapter vs. the proposed Training-free CLIP-Adapter (Tip-Adapter).** CLIP-Adapter and Tip-Adapter share similar architectures, namely, two linear layers and a residual connection blending updated visual features with pre-trained CLIP features. CLIP-Adapter is trained with Stochastic Gradient Descent (SGD), while Tip-Adapter is training-free, whose weights of linear layers are initialized from Cache Model.

The Pipeline of Tip-Adapter

- Image들을 visual encoder에 통과시켜 cache 한 후, 이들 과의 similarity를 additional logit으로 사용.
- Fine-tuning 시, $\mathbf{F}_{train}$만을 learnable parameter로 생각하여 학습함. $\mathbf{L}_{train}$이 바뀌면 정답이 바뀌는 개념이라 freeze

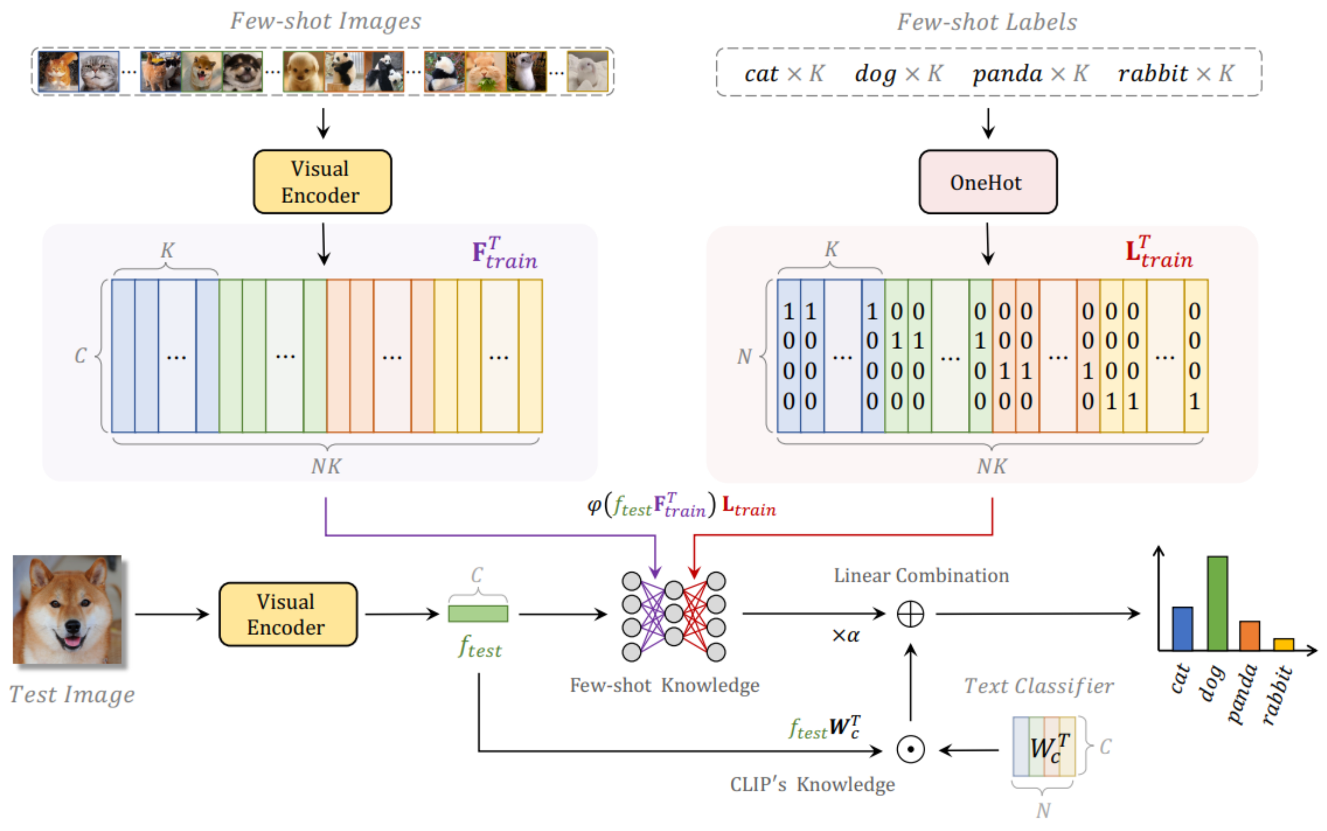


Figure 2. **The Pipeline of Tip-Adapter.** Given a K-shot N-class training set, we construct the weights of the two-layer adapter by creating a cache model from the few-shot training set. It contains few-shot visual features F_{train} encoded by CLIP's visual encoder and few-shot ground-truth labels L_{train}. F_{train} and L_{train} can be used as the weights for the first and second layers in the adapter.

Two hyperparameters:
β: sharpness of adapter's logit
α: residual ratio of adapter's logit

$$\varphi(x) = \exp\big(-\beta(1-x)\big)$$

Comparison with CLIP-Adapter

1. Initialization
2. Parameters (Tip-Adapter는 image encoder 이후 첫 번째 layer만 학습)
3. Hyperparameters (α,β)

- CLIP-Adapter의 hyperparameters:

$$\text{logit} = f^*\mathbf{W}^{*T}$$
$$f^* = \alpha A_v(f)^T + (1-\alpha)f$$
$$\mathbf{W}^* = \beta A_t(\mathbf{W})^T + (1-\beta)\mathbf{W}$$

- Tip-Adapter의 hyperparameters:

$$\text{logit} = \alpha\varphi\left(f\mathbf{F}_{\text{train}}^T\right)\mathbf{L}_{\text{train}} + f\mathbf{W}^T$$
$$\varphi(x) = \exp\left(-\beta(1-x)\right)$$

Results

- CoOp, CLIP-Adapter보다 few-shot에서 높은 성능을 기록함.

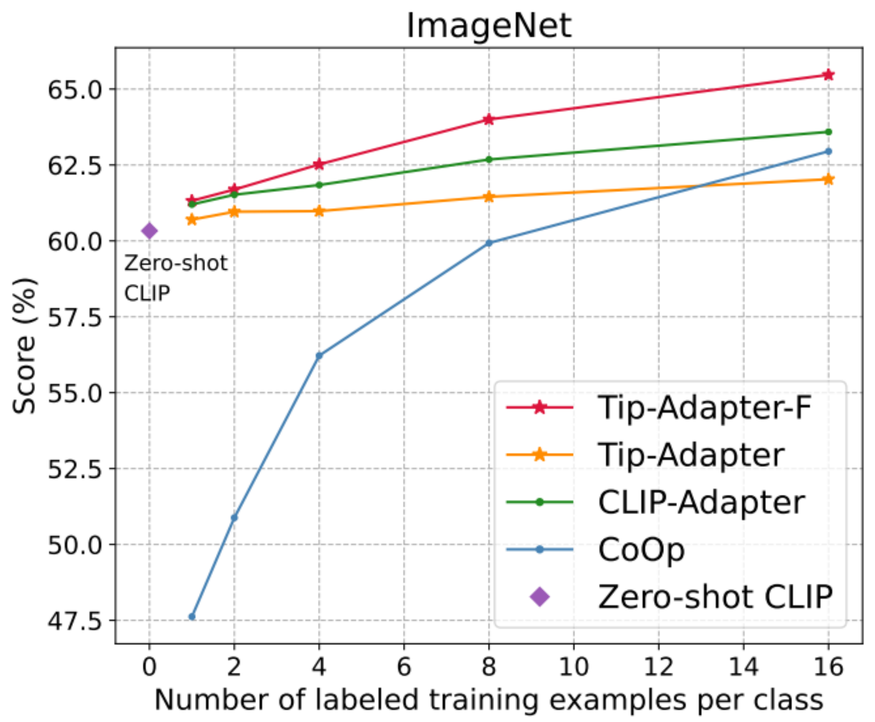


Figure 3. Classification accuracy of Tip-Adapter, Tip-Adapter-F and other models under different few-shot settings with CLIP-style pre-processing.

Models	Epochs	Time	Accuracy	Gain
Zero-shot CLIP [51]	0	0	60.33	0
Linear-probe CLIP [51]	-	13min	56.13	-4.20
CoOp [70]	200	14h 40min	62.95	+2.62
CLIP-Adapter [17]	200	50min	63.59	+3.26
Tip-Adapter	0	0	62.03	+1.70
Tip-Adapter-F	20	5min	65.51	+5.18

Table 3. Fine-tuning time and classification accuracy of 16-shot learning with different methods. The last column in blue records the performance gain relative to zero-shot CLIP.

Training time을 크게 줄임.

Implementation

- Cache train images and texts with CLIP image/text encoder: https://github.com/gaopengcuhk/Tip-Adapter/blob/main/utils.py#L38
- Search hyperparameters (α, β): https://github.com/gaopengcuhk/Tip-Adapter/blob/main/utils.py#L99
- Tip-Adapter: https://github.com/gaopengcuhk/Tip-Adapter/blob/main/main.py#L27
- Tip-Adapter-F: https://github.com/gaopengcuhk/Tip-Adapter/blob/main/main.py#L66

Tip-Adapter official implementation: https://github.com/gaopengcuhk/Tip-Adapter

Leveraging Pre-trained Generative Models

Is synthetic data from generative models ready for image recognition?

Ruifei He et al.

The University of Hong Kong, University of Oxford, ByteDance

ICLR 2023

GLIDE

- OpenAI의 text-to-image diffusion model: Stable Diffusion 이전에 공개되어 있던 가장 좋은 모델.

"a hedgehog using a calculator"

"a corgi wearing a red bowtie and a purple party hat"

"robots meditating in a vipassana retreat"

"a fall landscape with a small cottage next to a lake"

"a surrealist dream-like oil painting by salvador dalí of a cat playing checkers"

"a professional photo of a sunset behind the grand canyon"

"a high-quality oil painting of a psychedelic hamster dragon"

"an illustration of albert einstein wearing a superhero costume"

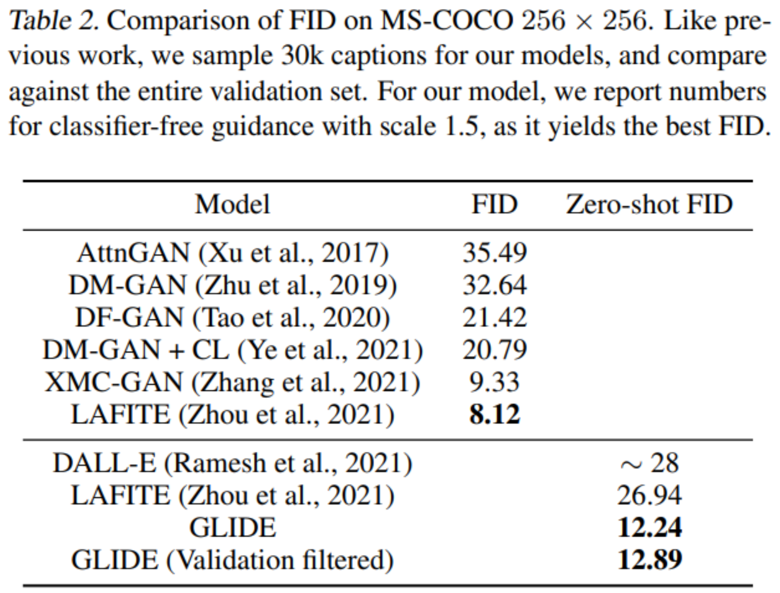


Table 2. Comparison of FID on MS-COCO 256 × 256. Like previous work, we sample 30k captions for our models, and compare against the entire validation set. For our model, we report numbers for classifier-free guidance with scale 1.5, as it yields the best FID.

Model	FID	Zero-shot FID
AttnGAN (Xu et al., 2017)	35.49	
DM-GAN (Zhu et al., 2019)	32.64	
DF-GAN (Tao et al., 2020)	21.42	
DM-GAN + CL (Ye et al., 2021)	20.79	
XMC-GAN (Zhang et al., 2021)	9.33	
LAFITE (Zhou et al., 2021)	**8.12**	
DALL-E (Ramesh et al., 2021)		∼ 28
LAFITE (Zhou et al., 2021)		26.94
GLIDE		**12.24**
GLIDE (Validation filtered)		**12.89**

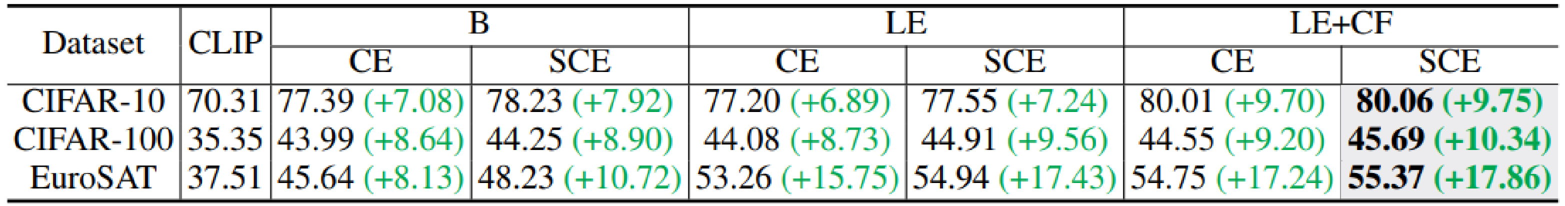
Nichol, Alex, et al. "Glide: Towards photorealistic image generation and editing with text-guided diffusion models." *ICML.* 2022.

Motivation

- 최근 text-to-image generation을 사람이 보기에 구분할 수 없을 정도로 굉장히 잘 하는데, 생성된 이미지들을 classification을 학습하는데 사용할 수 없을까?
- 생성되는 이미지들을 data-scares settings (zero-shot, few-shot)의 관점과 pre-training의 관점에서 각각 학습해본 후 성능을 측정해보자.
- 분석 위주의 논문으로, synthetic data를 이용해 성능 향상을 시키기 위한 다양한 트릭을 제안함.
- 본 논문에서 학습시키는 방법은 adapter를 사용하지 않은 단순 fine-tuning으로 zero-shot에서는 linear probing이 좋았고, few-shot에서는 CLIP image encoder를 fine-tuning 하였음.

Zero-shot: Synthetic Data Generation Strategy

- **Basic strategy (B):** Target의 label name을 text-to-image model의 입력으로 주어 이미지를 생성.
  - Text-to-image model에 짧고 단순한 입력만 주는 것은 생성하는 이미지의 diversity를 떨어뜨릴 수 있다.
- **Language Enhancement strategy (LE):** pre-trained word-to-sentence model T5를 이용하여 label name을 다양한 자연어 형태의 입력으로 바꿔준다.
  - E.g., “airplane” → “a white airplane hovering over a beach and a city.”
- **CLIP Filter strategy (CF):** 생성된 이미지 중에서는 label name을 잘 반영하지 못하거나 낮은 quality의 이미지가 있을 수 있다. 학습에 방해되지 않게끔 CLIP을 이용하여 low confidence 이미지들을 filtering out 시킨다.
- **Soft Cross-Entropy loss (SCE):** 정해진 threshold로 이미지를 제거하는 것 뿐 아니라, 학습 시에 normalized CLIP score를 이용하여 soft label로 label을 제공하면 data noise에 더 강건 해진다.

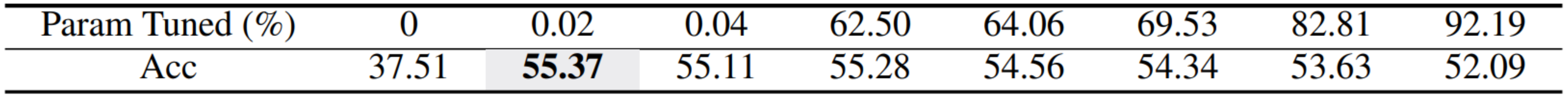


Dataset	CLIP	B		LE		LE+CF	
		CE	SCE	CE	SCE	CE	SCE
CIFAR-10	70.31	77.39 (+7.08)	78.23 (+7.92)	77.20 (+6.89)	77.55 (+7.24)	80.01 (+9.70)	**80.06 (+9.75)**
CIFAR-100	35.35	43.99 (+8.64)	44.25 (+8.90)	44.08 (+8.73)	44.91 (+9.56)	44.55 (+9.20)	**45.69 (+10.34)**
EuroSAT	37.51	45.64 (+8.13)	48.23 (+10.72)	53.26 (+15.75)	54.94 (+17.43)	54.75 (+17.24)	**55.37 (+17.86)**

Table 2: Ablation study on **Language Enhancement (LE), CLIP-based Filtering (CF), and Soft-target Cross-Entropy (SCE).**

Zero-shot: Efficiency of Synthetic Data

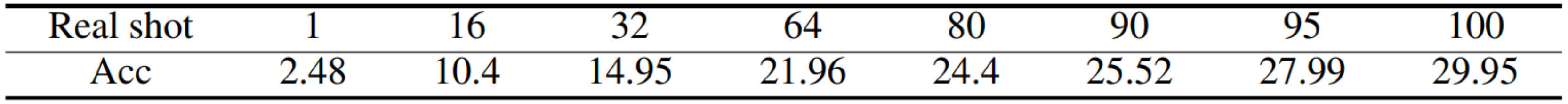
- Synthetic data가 효율적으로 학습시킬 수 있는 parameter의 수



Param Tuned (%)	0	0.02	0.04	62.50	64.06	69.53	82.81	92.19
Acc	37.51	**55.37**	55.11	55.28	54.56	54.34	53.63	52.09

Table 3: **Parameters tuned v.s. Accuracy.** Dataset: EuroSAT.

- Synthetic data와 real data의 efficiency 비교 (from scratch)
  - Synthetic data는 500 images per class로 학습했을 때 성능: 28.74%

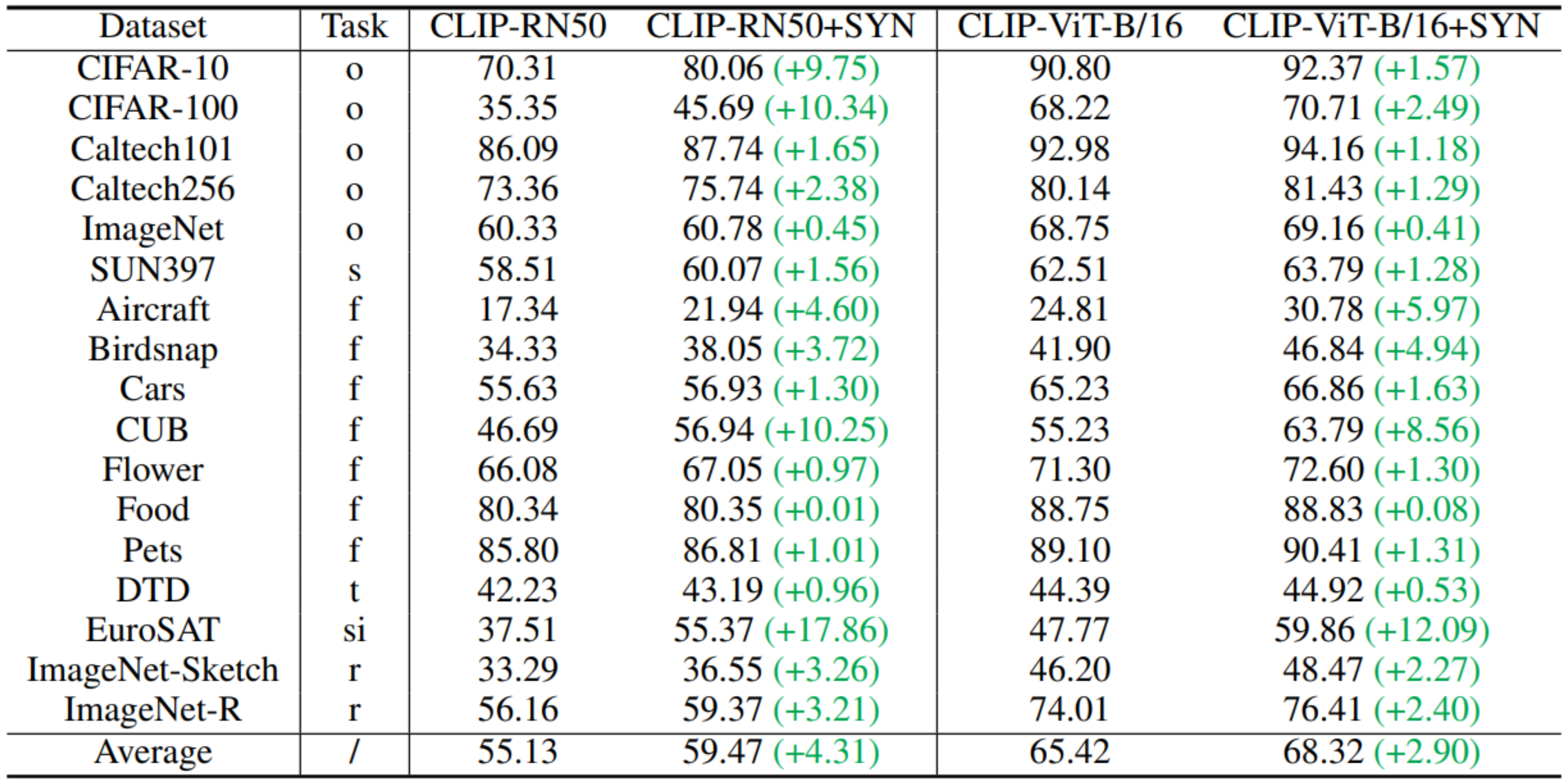


Real shot	1	16	32	64	80	90	95	100
Acc	2.48	10.4	14.95	21.96	24.4	25.52	27.99	29.95

Table 4: **Setting when training from scratch.** Dataset: CIFAR-100.

  - 성능을 더 높이려고 할 수록 그 차이는 커질 것으로 예상됨.

Zero-shot: Results



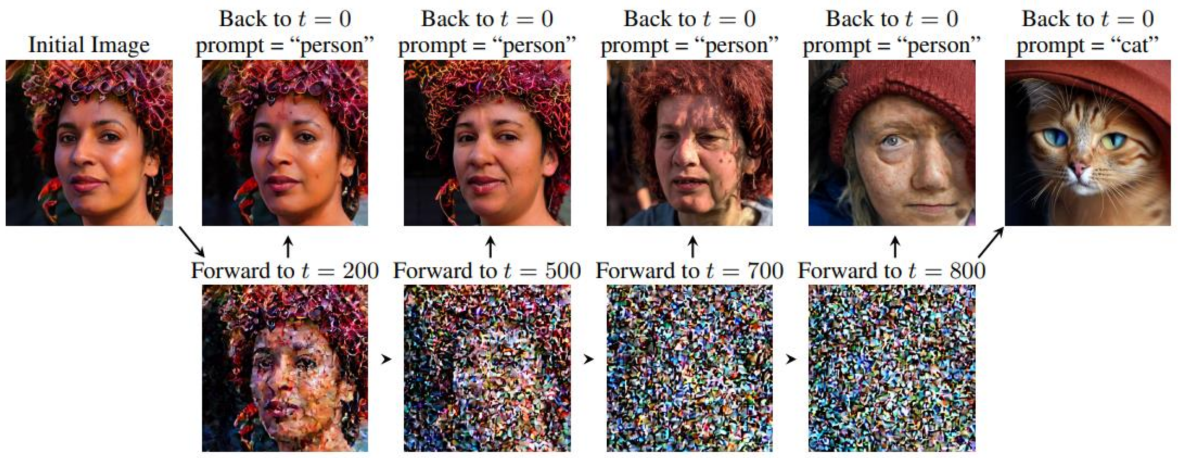
Dataset	Task	CLIP-RN50	CLIP-RN50+SYN	CLIP-ViT-B/16	CLIP-ViT-B/16+SYN
CIFAR-10	o	70.31	80.06 (+9.75)	90.80	92.37 (+1.57)
CIFAR-100	o	35.35	45.69 (+10.34)	68.22	70.71 (+2.49)
Caltech101	o	86.09	87.74 (+1.65)	92.98	94.16 (+1.18)
Caltech256	o	73.36	75.74 (+2.38)	80.14	81.43 (+1.29)
ImageNet	o	60.33	60.78 (+0.45)	68.75	69.16 (+0.41)
SUN397	s	58.51	60.07 (+1.56)	62.51	63.79 (+1.28)
Aircraft	f	17.34	21.94 (+4.60)	24.81	30.78 (+5.97)
Birdsnap	f	34.33	38.05 (+3.72)	41.90	46.84 (+4.94)
Cars	f	55.63	56.93 (+1.30)	65.23	66.86 (+1.63)
CUB	f	46.69	56.94 (+10.25)	55.23	63.79 (+8.56)
Flower	f	66.08	67.05 (+0.97)	71.30	72.60 (+1.30)
Food	f	80.34	80.35 (+0.01)	88.75	88.83 (+0.08)
Pets	f	85.80	86.81 (+1.01)	89.10	90.41 (+1.31)
DTD	t	42.23	43.19 (+0.96)	44.39	44.92 (+0.53)
EuroSAT	si	37.51	55.37 (+17.86)	47.77	59.86 (+12.09)
ImageNet-Sketch	r	33.29	36.55 (+3.26)	46.20	48.47 (+2.27)
ImageNet-R	r	56.16	59.37 (+3.21)	74.01	76.41 (+2.40)
Average	/	55.13	59.47 (+4.31)	65.42	68.32 (+2.90)

Table 1: **Main Results on Zero-shot Image Recognition.** All results are top-1 accuracy on test set. o: object-level. s: scene-level. f: fine-grained. t: textures. si: satellite images. r: robustness.

모든 데이터셋에서 성능을 향상시킬 수 있었음.
다만, 실험 과정에서 validation set을 사용했기 때문에 진정한 zero-shot인지 공정성을 의심해 보아야 함.

Few-shot: Synthetic Data Generation Strategy

- **Basic strategy (B):** Zero-shot에서 가장 좋았던 LE+CF+SCE를 기본으로 설정
- Few-shot 이미지를 가지고 있으므로 가지고 있는 이미지를 활용하는 추가 strategy를 세움.
- **Real Filtering (RF):** 생성된 이미지의 CLIP embedding이 다른 label의 이미지 CLIP embedding과 가까이 있다면 filtering out.
- **Real Guidance (RG):** Generation 보다는 augmentation에 가까운 방법으로, 가지고 있는 few-shot 이미지에 noise를 뿌렸다가 제거하는 방식으로 data를 augmentation하는 방법.

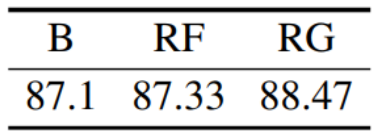


Initial Image
Back to $t = 0$ prompt = "person"
Back to $t = 0$ prompt = "person"
Back to $t = 0$ prompt = "person"
Back to $t = 0$ prompt = "person"
Back to $t = 0$ prompt = "cat"
Forward to $t = 200$
Forward to $t = 500$
Forward to $t = 700$
Forward to $t = 800$

이해를 돕기 위한 같은 시기에 나온 다른 논문 (Boomerang)

# shots	1	2	4	8	16
t^*	50	40	35	20	15

전체 100번의 steps 중 사용된 RG steps



B	RF	RG
87.1	87.33	88.47

Table 5: Ablation for Basic strategy (**B**), Real Filtering (**RF**), Real Guidance (**RG**) on EuroSAT, 16 shot.

Luzi, Lorenzo, et al. "Boomerang: Local sampling on image manifolds using diffusion models." *arXiv.* 2022.

Few-shot: Results

- CT w. Syn은 CT w. init을 포함한 모든 baselines보다 좋은 성능을 보여주었다.
- **즉, synthetic data가 few-shot에서도 성능 향상에 도움을 주었다.**
- Phase-wise training (syn→real, real→syn)보다 naïve하게 섞어서 학습하는 것이 성능이 더 좋았다.
  - Real, syn sample에 대해서 1:1로 update를 진행하지만, synthetic에 대해서는 SCE를 진행하고, batch size 도 32, 512로 다르다.

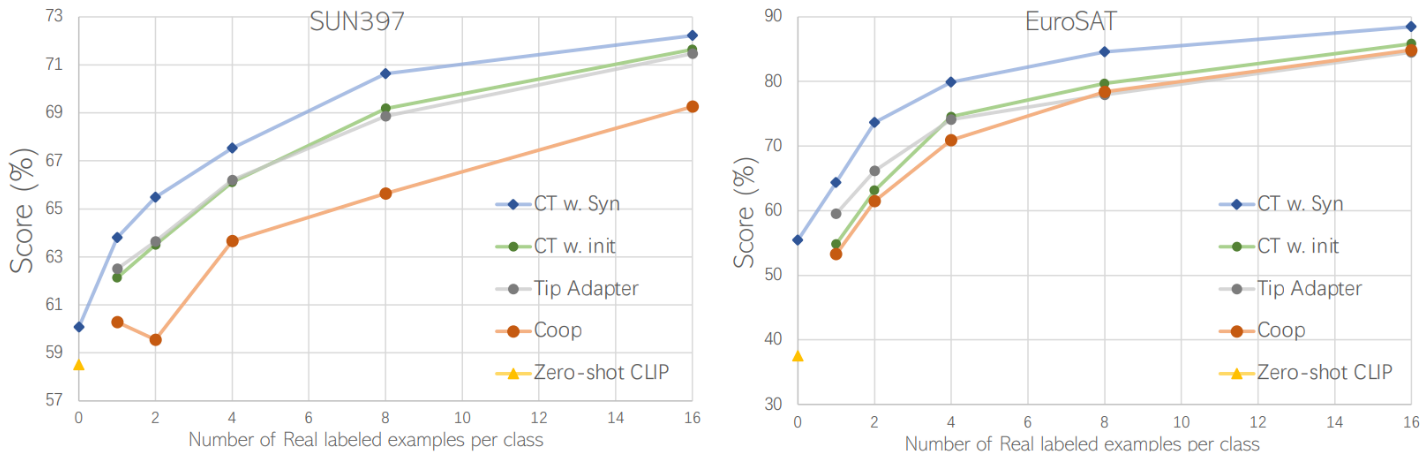
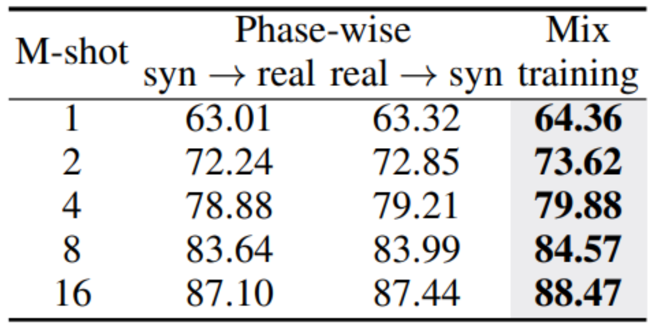


Figure 1: Results for few-shot image recognition. Results on all 8 datasets are provided in Appendix.



M-shot	Phase-wise syn → real	Phase-wise real → syn	Mix training
1	63.01	63.32	**64.36**
2	72.24	72.85	**73.62**
4	78.88	79.21	**79.88**
8	83.64	83.99	**84.57**
16	87.10	87.44	**88.47**

Table 7: **Mix training works better** for few-shot tasks on EuroSAT.

Few-shot: Frozen Batch Normalization Strategy

- Linear probing을 하는 것보다, 전체 network를 end-to-end fine-tune하되, batch norm의 normalization statistics을 freeze하는 방법론.
- CT w. init이 Tip-Adapter, CoOp보다 좋거나 비슷한 성능을 내는 이유도 Frozen BN strategy에 있다.
- 특히 synthetic data가 들어오면서 distribution shift를 걱정하는 지금과 같은 상황에서 더욱 적합한 방법론으로 생각된다.

```
def freeze_BN(model):
    for module in model.modules():
        # print(module)
        if isinstance(module, torch.nn.BatchNorm2d):
            module.eval()
    return model
```

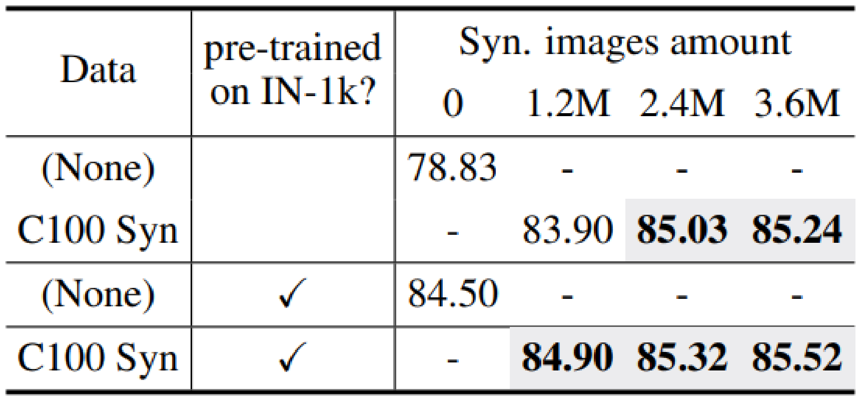
Implementation of freeze batch norm

Train data	Freeze BN?	Test Acc
Real		75.31
Real	✓	**85.63**
Syn		44.73
Syn	✓	**55.37**

Table 6: **Frozen BN works better** for 16-shot settings on EuroSAT.

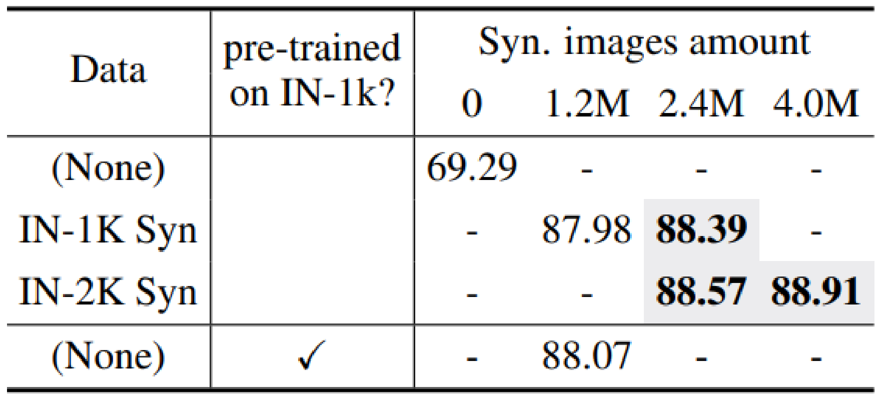
Pre-training: Classification

- Synthetic data를 pre-training에 사용하기 위해서 큰 데이터셋을 구축함.
- CIFAR-100을 위해서 CIFAR-100의 label name을 이용해서 pre-training한 결과.
- CIFAR-100과 무관하게 IN-1k, IN-2k의 label name을 이용해서 pre-training한 결과.
- 각각 기존에 사용하던 IN-1k pre-trained model의 성능보다 더 좋은 것을 확인하였음.



Data	pre-trained on IN-1k?	Syn. images amount			
		0	1.2M	2.4M	3.6M
(None)		78.83	-	-	-
C100 Syn		-	83.90	**85.03**	**85.24**
(None)	✓	84.50	-	-	-
C100 Syn	✓	-	**84.90**	**85.32**	**85.52**

Table 8: Results on CIFAR-100 with **downstream-aware supervised pre-training**. C100: CIFAR100.

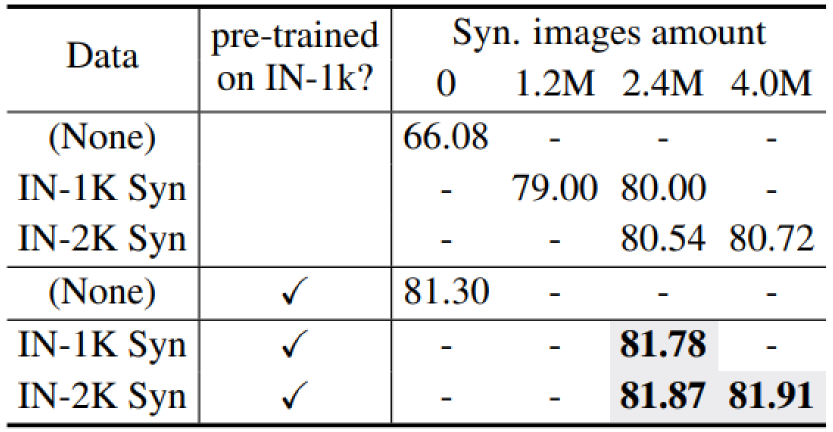


Data	pre-trained on IN-1k?	Syn. images amount			
		0	1.2M	2.4M	4.0M
(None)		69.29	-	-	-
IN-1K Syn		-	87.98	**88.39**	-
IN-2K Syn		-	-	**88.57**	**88.91**
(None)	✓	-	88.07	-	-

Table 9: Results on CIFAR-100 with **downstream-agnostic supervised pre-training**. Backbone: DeiT-S.

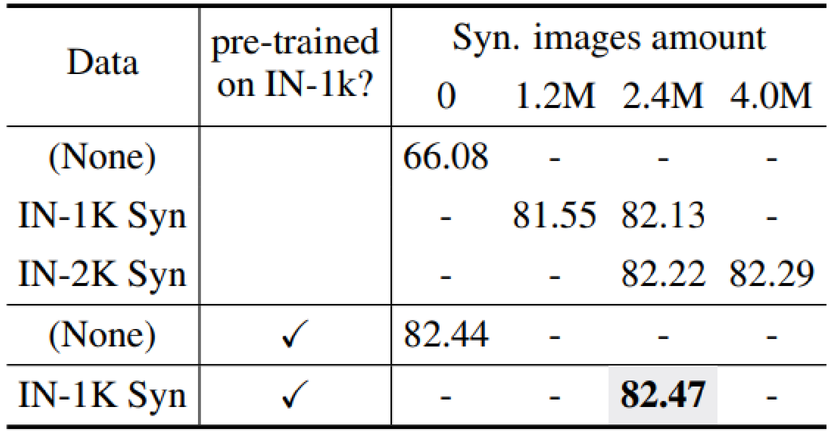
Pre-training: Object Detection

- Classification외에 object detection의 pre-training으로도 성능이 향상되는지 확인하였음.
- PASCAL VOC 데이터셋에 대해서 각각 classification 방식의 pre-training과 self-supervised learning 방식의 pre-training 모두 성능이 향상되는 것을 확인하였음.
- **다만 그 차이가 미미하여 생성에 사용하는 cost 및 학습하는 cost 대비 아직은 비효율적임.**



Data	pre-trained on IN-1k?	Syn. images amount			
		0	1.2M	2.4M	4.0M
(None)		66.08	-	-	-
IN-1K Syn		-	79.00	80.00	-
IN-2K Syn		-	-	80.54	80.72
(None)	✓	81.30	-	-	-
IN-1K Syn	✓	-	-	**81.78**	-
IN-2K Syn	✓	-	-	**81.87**	**81.91**

Table 10: Results for object detection on PASCAL VOC with **downstream-agnostic supervised pre-training**, all results are reported in AP_{50}.



Data	pre-trained on IN-1k?	Syn. images amount			
		0	1.2M	2.4M	4.0M
(None)		66.08	-	-	-
IN-1K Syn		-	81.55	82.13	-
IN-2K Syn		-	-	82.22	82.29
(None)	✓	82.44	-	-	-
IN-1K Syn	✓	-	-	**82.47**	-

Table 11: Results for object detection on PASCAL VOC with **downstream-agnostic self-supervised pre-training (Moco v2)**, all results are reported in AP_{50}.

Prompt, Generate, then Cache: Cascade of Foundation Models makes Strong Few-shot Learners

Renrui Zhang*, Xiangfei Hu* et al.

Shanghai Artificial Intelligence Laboratory, The Chinese University of Hong Kong, University of Chinese Academy of Sciences

CVPR 2023

Motivation: Cascade of Foundation Models

- 현재 다양한 방식으로 pre-training 되어 있는 foundation 모델이 존재하는데, 이 모델들을 하나의 프레임워크로 묶어 data-scares settings에서 classification 성능을 극대화 해보자.

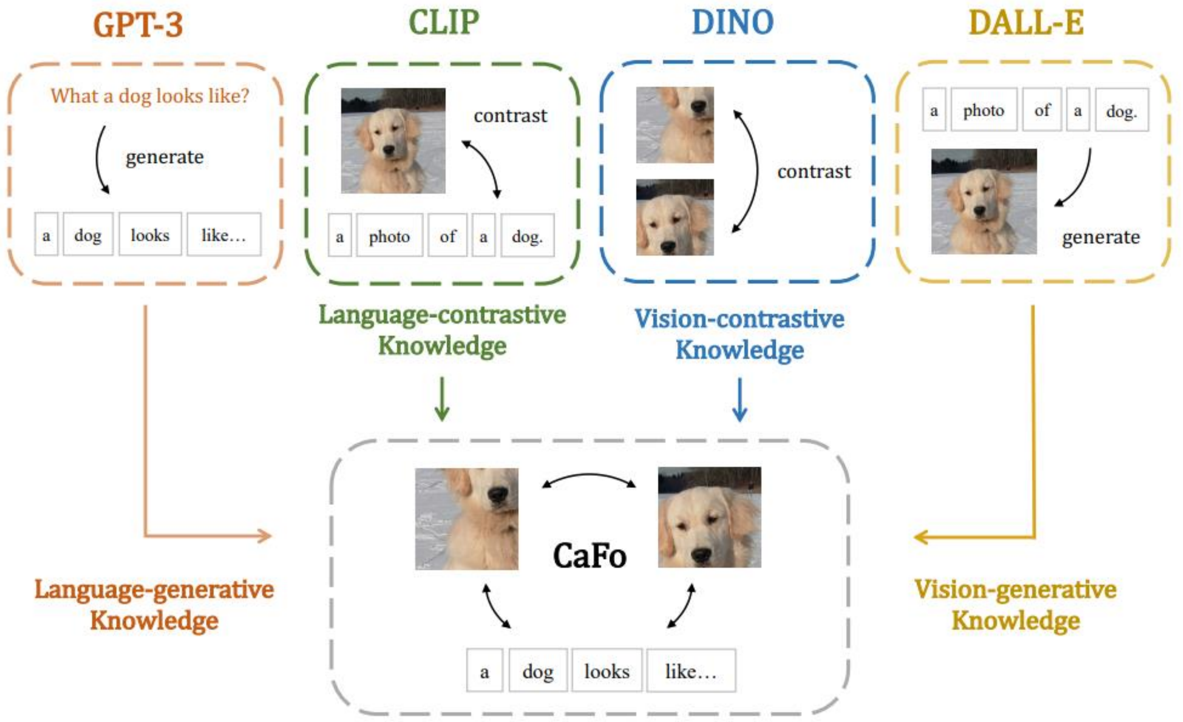
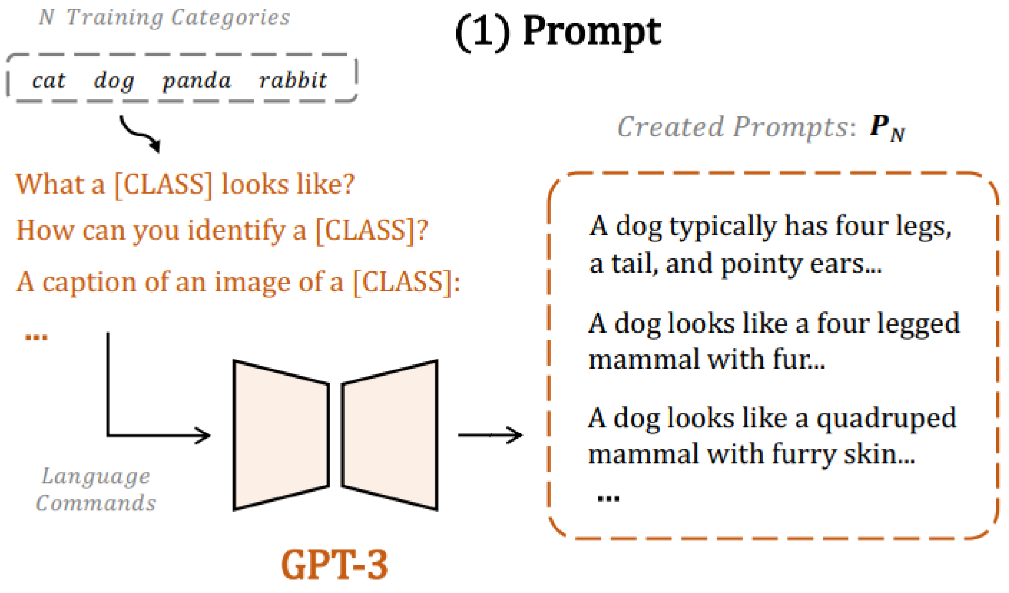


Figure 1. **The Cascade Paradigm of CaFo.** We adaptively incorporate the knowledge from four types of pre-training methods and achieve a strong few-shot learner.

사용한 데이터셋도 모두 다르고, 학습 방식도 모두 다른 네 foundation models

Prompt with GPT-3

- CLIP text encoder에 들어갈 입력을 단순히 “a photo of a {label}” 등으로 정해져 있는 prompt로 정하는 것이 아니라 GPT-3를 이용하여 각 label name에 대해서 rich한 description을 만드는 과정.
- 생성된 P_N은 CLIP text encoder에 들어가서 values로 사용된다 ($\text{CLIP}_{\text{text}}(P_N)$).



N Training Categories
cat dog panda rabbit
(1) Prompt
What a [CLASS] looks like?
How can you identify a [CLASS]?
A caption of an image of a [CLASS]:
...
Language Commands
GPT-3
Created Prompts: P_N
A dog typically has four legs, a tail, and pointy ears...
A dog looks like a four legged mammal with fur...
A dog looks like a quadruped mammal with furry skin...
...

Figure 2. **Prompt with GPT-3 [4]**. As the first step in CaFo, we utilize the pre-trained GPT-3 to produce prompts with rich linguistic semantics for CLIP’s textual encoder.

$$P_N = \text{GPT3}(\text{Commands})$$

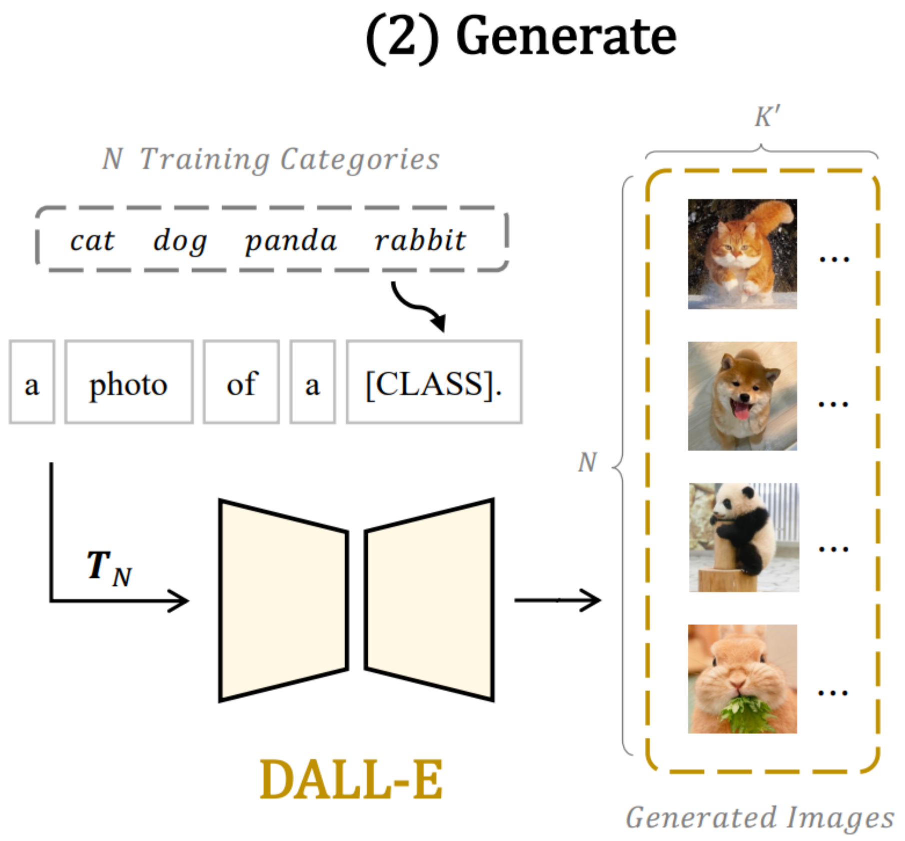
- “What a [CLASS] looks like?”
- “How can you identify a [CLASS]?”
- “A caption of an image of a [CLASS]:”

Pratt, Sarah, et al. "What does a platypus look like? generating customized prompts for zero-shot image classification." *arXiv.* 2022.

Generate via DALL-E

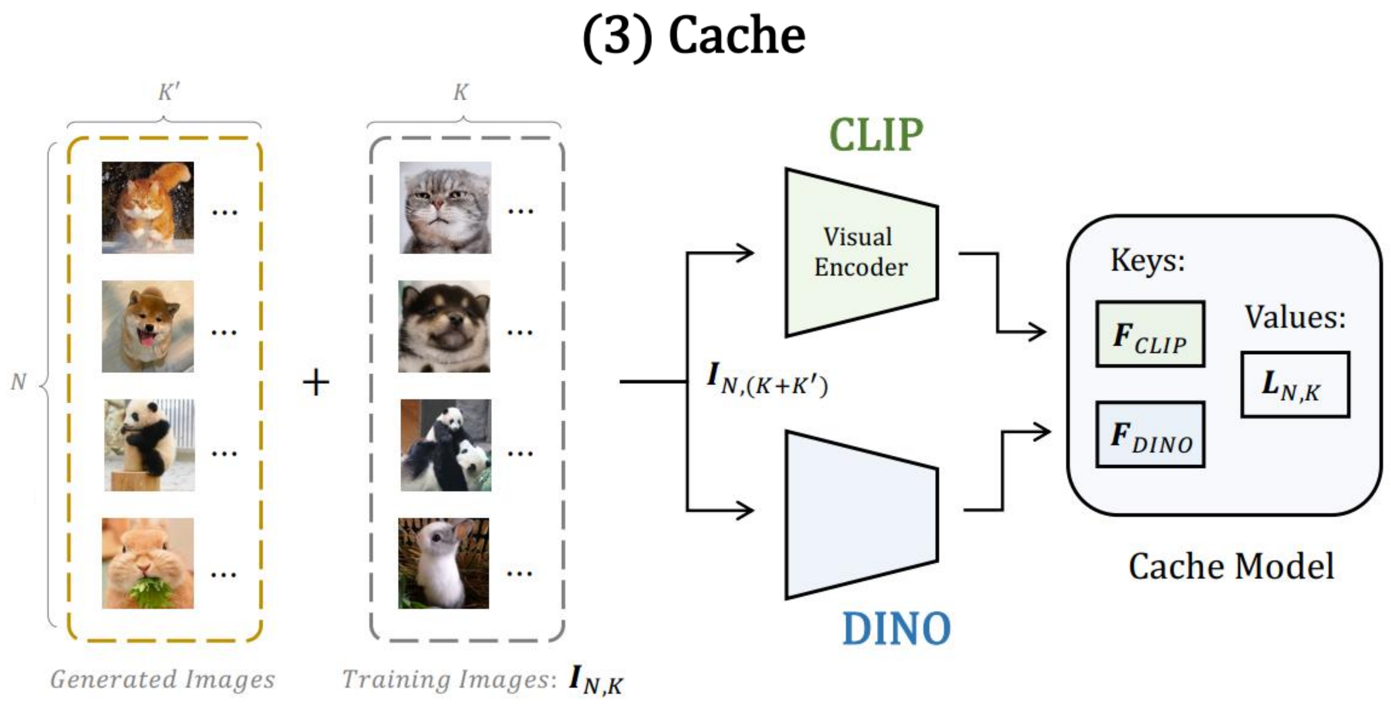
- Zero-shot, few-shot classification에서 성능 향상을 위해 IsSyntheticData처럼 이미지를 생성하여 데이터셋의 크기를 증강시킴.
- “a photo of a [CLASS]”의 형태로 입력을 주었으며, N-way K-shot problem을 N-way $(K + K')$-shot problem으로 바꿔주었다.
- Data expansion이라고도 부르며 totally zero-shot.

$$I_{N,(K+K')} = \{DALLE(T_N), I_{N,K}\}$$



Cache by CLIP and DINO

- 생성된 이미지와 few-shot setting의 training 이미지들은 Tip-Adapter에서 진행한 것 처럼 CLIP과 DINO에 의해 cache된다.



$$F_{\text{CLIP}} = \text{CLIP}_{\text{vis}}\left(I_{N,(K+K')}\right)$$
$$F_{\text{DINO}} = \text{DINO}\left(I_{N,(K+K')}\right)$$
$$L_{\text{onehot}} = \text{one} - \text{hot label}$$

Adaptive Inference

- 최종적으로 세 가지 종류의 predicted logits을 얻게 되었다.
- $p_{ZS} = f_{\text{CLIP}} \text{CLIP}_{\text{text}}(P_N)^T,\ p_{\text{CLIP}} = \varphi\left(f_{\text{CLIP}} F_{\text{CLIP}}^T\right) L_{\text{onehot}},\ p_{\text{DINO}} = \varphi\left(f_{\text{DINO}} F_{\text{DINO}}^T\right) L_{\text{onehot}}$
  - Where $\varphi(x) = \exp\left(-\beta(1-x)\right)$인 non-linear activation.
- 이 셋을 ensemble하여 하나의 logit을 얻어낼 것이다.

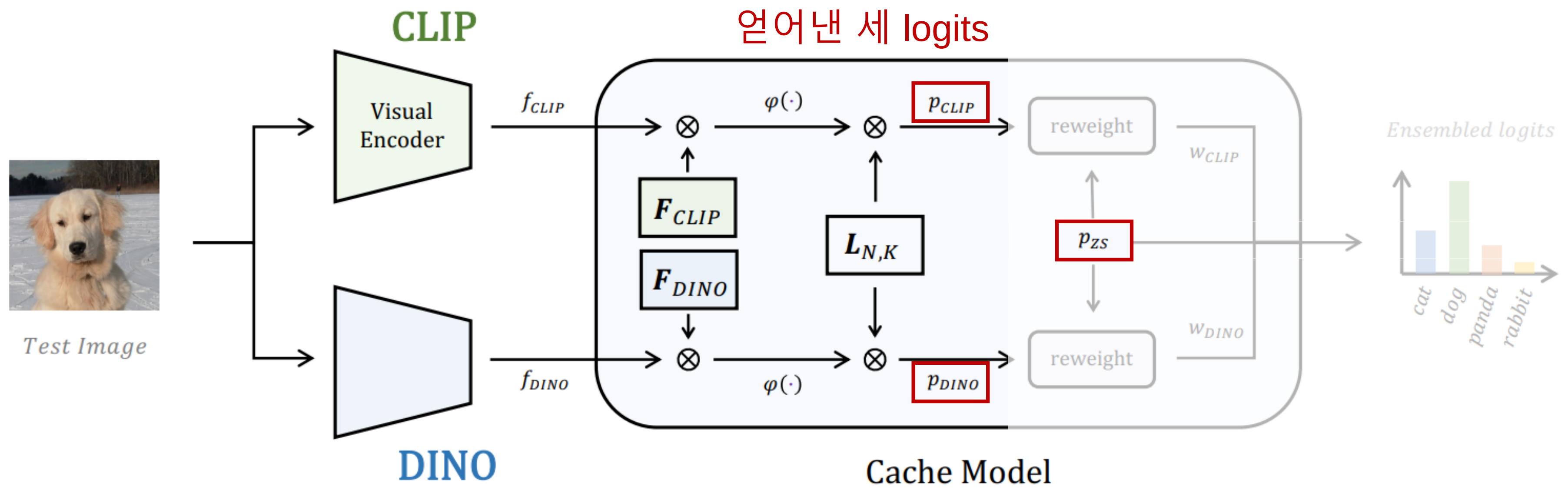


Figure 4. **Adaptive Inference with Cache Model.** We regard the test image as a query and retrieves CLIP and DINO's knowledge from the corresponding two keys in the cache model. Then, we calculate the distribution similarities between different classification logits for adaptive ensemble.

Adaptive Inference

- p_{ZS}을 가장 신뢰하기 때문에, p_{CLIP}과 p_{DINO}을 p_{ZS}에 projection하여 reweighting을 진행하였다.
  - 안정적인 대신 Zero-shot CLIP의 logit을 크게 벗어나지 못한다.
- $p_{\text{ens}} = p_{ZS} + \alpha(p_{\text{CLIP}} \cdot \text{softmax}(w)_0 + p_{\text{DINO}} \cdot \text{softmax}(w)_1)$ where $w = [p_{CLIP} \cdot p_{ZS}, p_{DINO} \cdot p_{ZS}]$.
- CLIP, DINO에 한정되지 않고, 다양한 cache model에 대해서 adaptive inference를 진행할 수 있다.

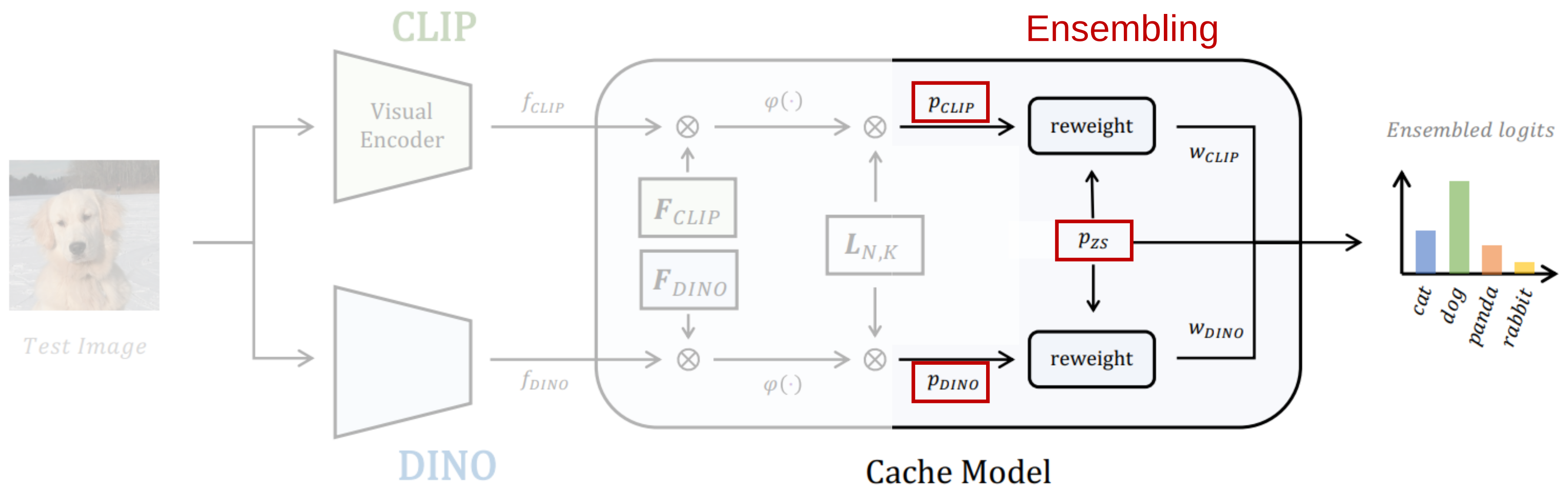


Figure 4. **Adaptive Inference with Cache Model.** We regard the test image as a query and retrieves CLIP and DINO's knowledge from the corresponding two keys in the cache model. Then, we calculate the distribution similarities between different classification logits for adaptive ensemble.

Performance Comparison on ImageNet

- State-of-the-art performance on few-shot ImageNet1k.
- 학습하는데 걸리는 시간도 Tip-Adapter를 따랐기 때문에 빠르다.

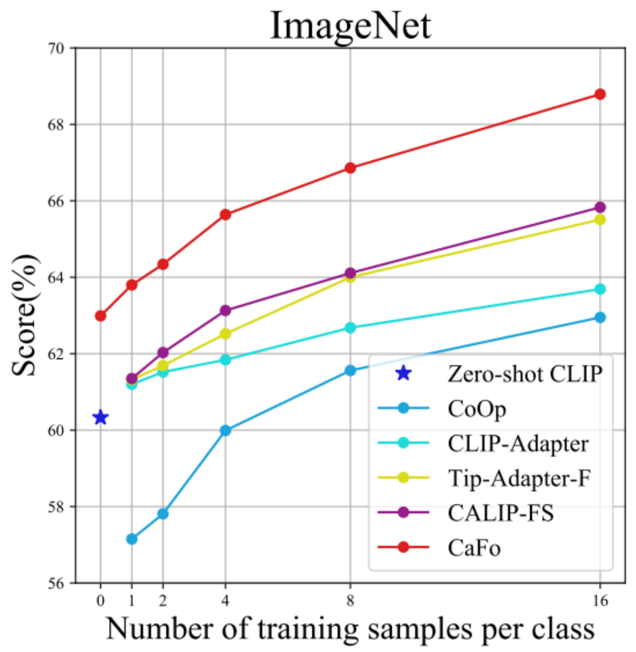
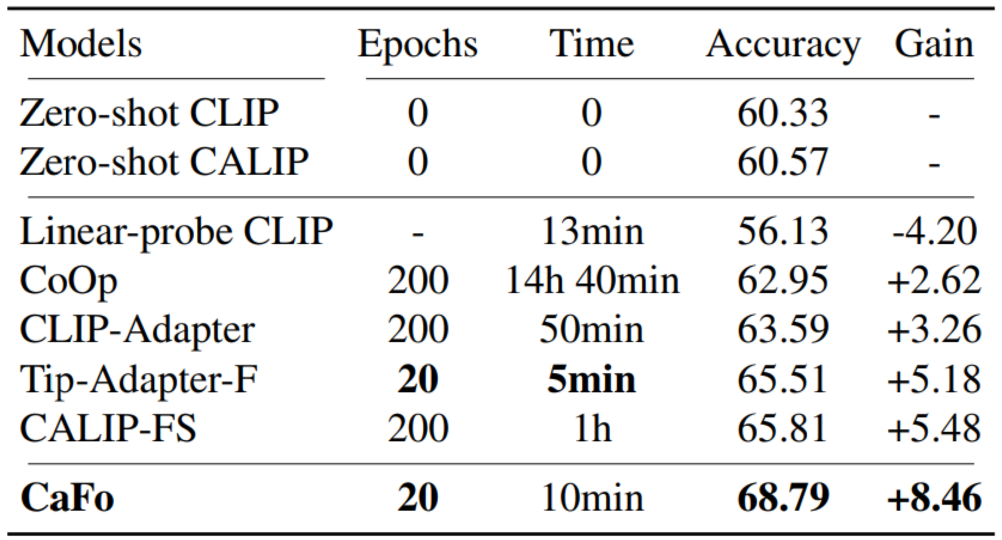


Figure 5. **Performance (%) Comparison on ImageNet.** We compare CaFo with other methods for different few-shot settings.

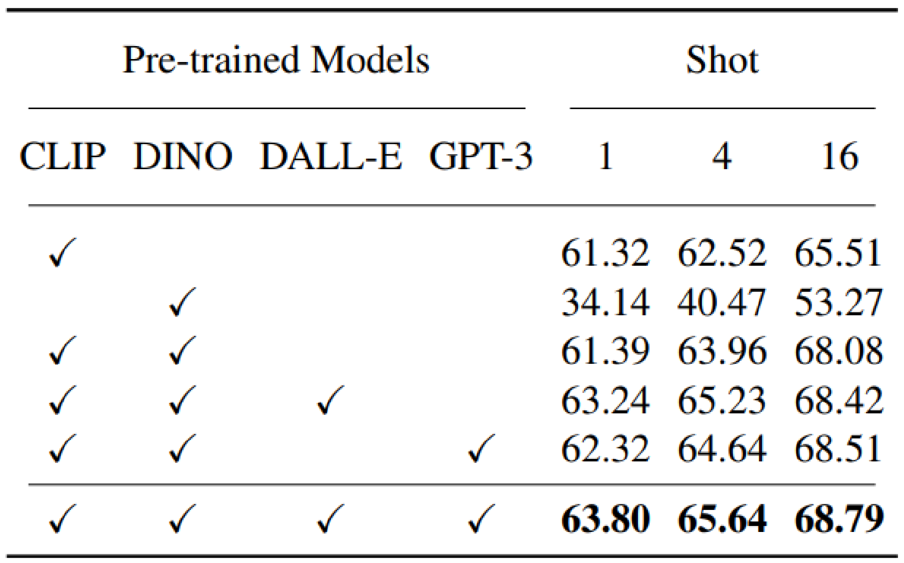


Models	Epochs	Time	Accuracy	Gain
Zero-shot CLIP	0	0	60.33	-
Zero-shot CALIP	0	0	60.57	-
Linear-probe CLIP	-	13min	56.13	-4.20
CoOp	200	14h 40min	62.95	+2.62
CLIP-Adapter	200	50min	63.59	+3.26
Tip-Adapter-F	**20**	**5min**	65.51	+5.18
CALIP-FS	200	1h	65.81	+5.48
CaFo	**20**	10min	**68.79**	**+8.46**

Table 1. **Efficiency Comparison on ImageNet.** We test the training time with a single A100 GPU under 16-shot setting.

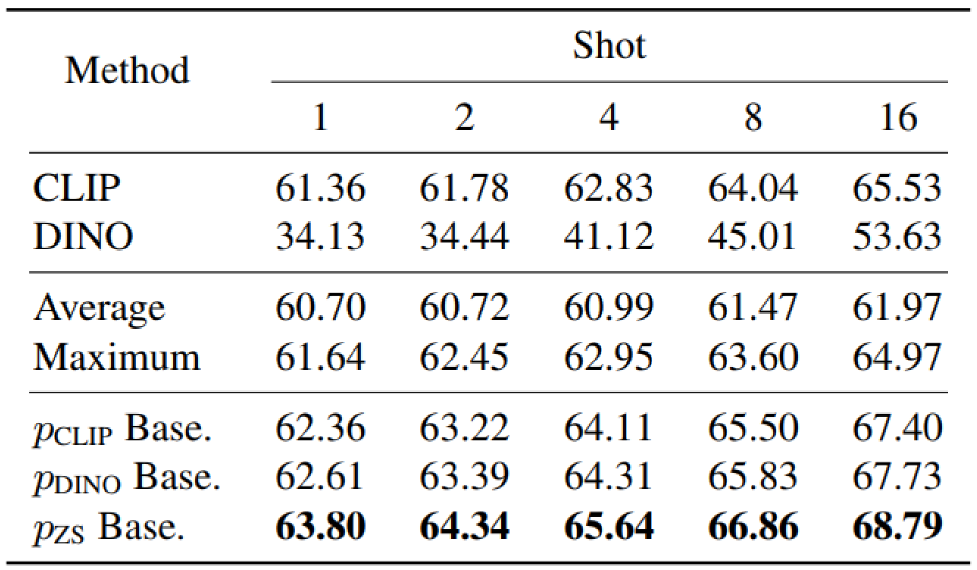
Ablation Studies

- 각각의 pre-trained models에 대해서 ablation study를 진행한 결과.
- Adaptive inference의 ablation study 결과.



Pre-trained Models				Shot		
CLIP	DINO	DALL-E	GPT-3	1	4	16
✓				61.32	62.52	65.51
	✓			34.14	40.47	53.27
✓	✓			61.39	63.96	68.08
✓	✓	✓		63.24	65.23	68.42
✓	✓		✓	62.32	64.64	68.51
✓	✓	✓	✓	**63.80**	**65.64**	**68.79**

Table 4. **Ablation Study (%) of Cascaded Models.** We ablate different pre-trained models on ImageNet with 1, 4, and 16 shots.



Method	Shot				
	1	2	4	8	16
CLIP	61.36	61.78	62.83	64.04	65.53
DINO	34.13	34.44	41.12	45.01	53.63
Average	60.70	60.72	60.99	61.47	61.97
Maximum	61.64	62.45	62.95	63.60	64.97
p_{CLIP} Base.	62.36	63.22	64.11	65.50	67.40
p_{DINO} Base.	62.61	63.39	64.31	65.83	67.73
p_{ZS} Base.	**63.80**	**64.34**	**65.64**	**66.86**	**68.79**

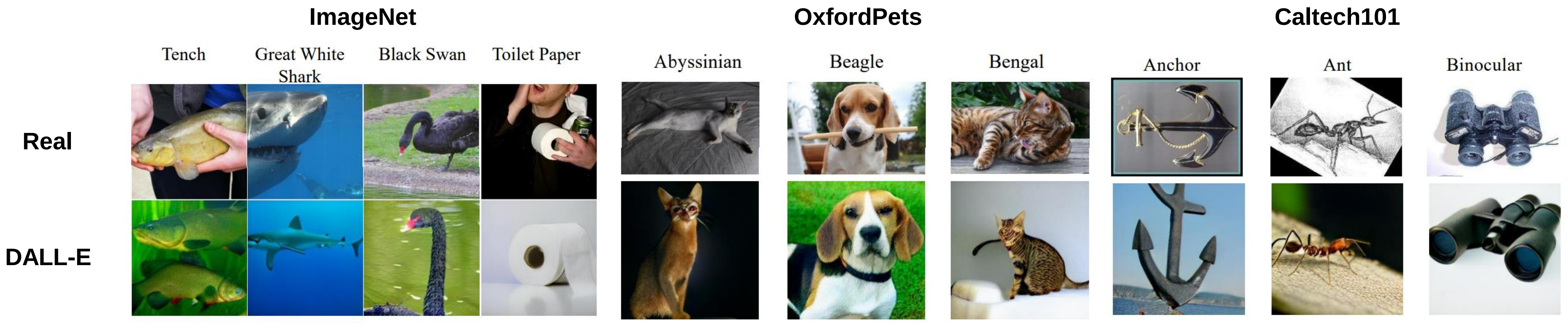
Table 5. **Ablation Study (%) of Adaptive Inference.** We conduct different ensemble methods of cache model on ImageNet.

Generated Number via DALL-E

- 생성 이미지 수에 따른 few-shot classification performance.
- DALL-E에서 생성한 이미지가 눈으로 보기에 구분할 수 없을 정도로 잘 생성되었음에도 생성 사진을 늘렸을 때 성능 향상이 차이가 없거나 줄어든다.
- 생성 모델을 더 잘 활용할 수 있는 방법이 필요한 것으로 보임.

DALL-E	1	2	4	8	16
1	63.29	64.06	65.11	66.48	68.64
2	63.66	**64.34**	65.37	**66.86**	**68.79**
4	63.71	64.33	65.35	66.75	68.61
8	**63.80**	64.26	**65.64**	66.68	68.76
16	63.68	64.16	65.40	66.57	68.41

Table 6. **Ablation Study (%) of Generated Number via DALL-E.** We compare different shot numbers on ImageNet.



Qualitative results of the generated dataset via DALL-E.

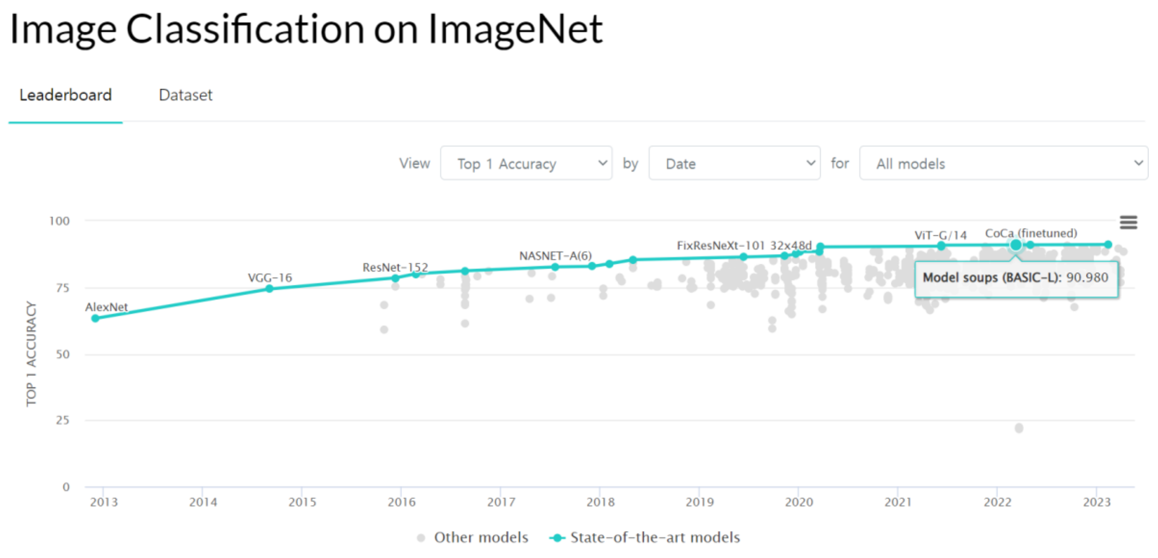
Conclusion

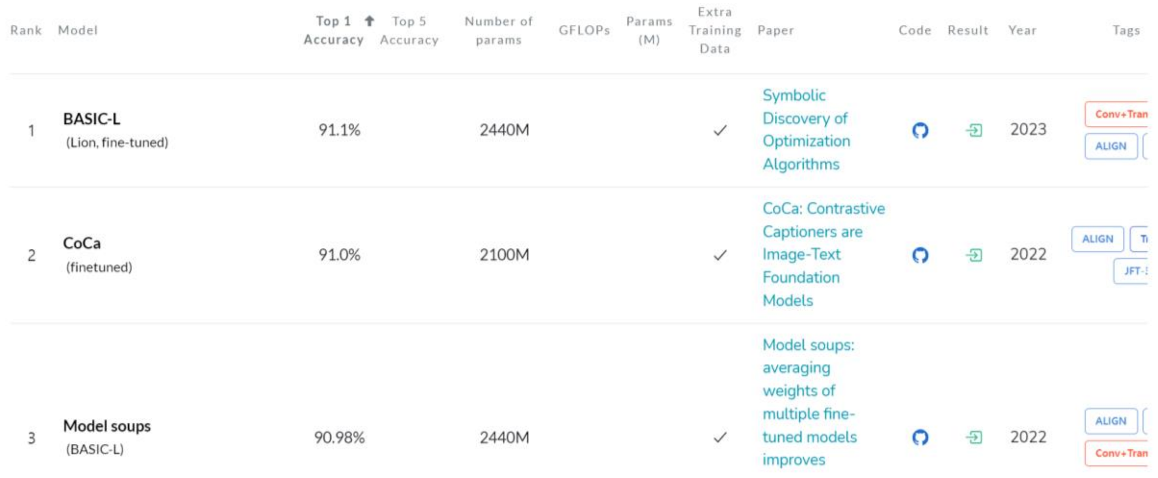
- CLIP을 통해 zero-shot classification을 진행할 수 있고, adapter를 활용하면 zero-shot에서의 성능 하락 없이 few-shot을 학습할 수 있다.
- Synthetic data는 zero-shot, few-shot classification에서 유의미한 성능 향상을 보여주었지만, fully supervised learning에 가까워 질 수록 효과가 미미하다.
- Transfer learning을 위한 pre-training으로 사용할 때는 성능이 ImageNet1k pre-trained와 비슷한 정도로 유의미한 데이터이다.
- Synthetic data의 수에 따른 성능 향상은 굉장히 미미했다.
    - CaFo의 경우 synthetic data를 넣어줄 때, 2-shot 이후로 효과가 미미하였다.
- CaFo처럼 다양한 foundation model을 사용하면 성능을 극대화 할 수 있다.

Session 2:
SOTA Classification Methods

Conventional Recipe: Hyper-parameters

- **Hyper-parameter search**
  - 다양한 hyperparameters에 대해서 multiple models을 학습한다.
  - 그 중 validation set에서 가장 높은 성능을 보인 **하나의 모델만 선택하고, 나머지는 다 버린다.**
- **Ensemble**
  - 다양한 hyperparameters로 찾은 다양한 모델들에 대해서 **모두 inference를 진행한다.**
  - 각각의 models에서 나온 logits을 average하여 최종 logit을 계산한다.



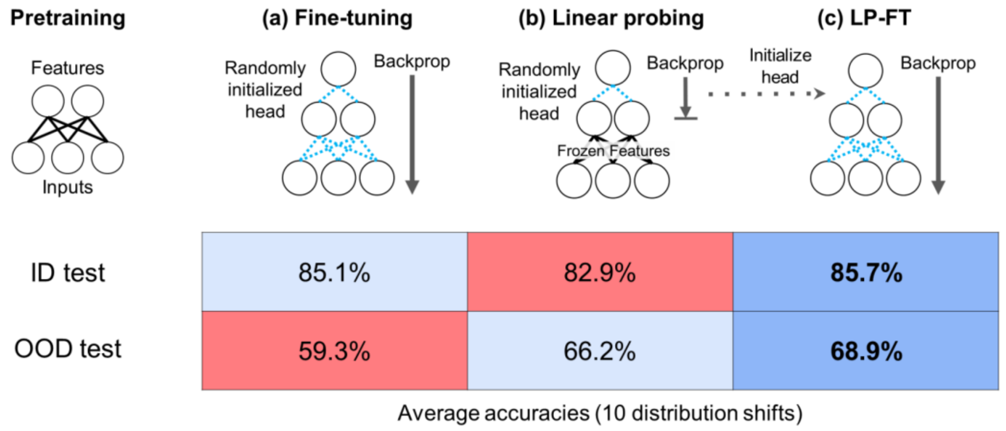


Rank	Model	Top 1 Accuracy	Top 5 Accuracy	Number of params	GFLOPs	Params (M)	Extra Training Data	Paper	Code	Result	Year	Tags
1	**BASIC-L** (Lion, fine-tuned)	91.1%		2440M			✓	Symbolic Discovery of Optimization Algorithms			2023	Conv+Tran ALIGN
2	**CoCa** (finetuned)	91.0%		2100M			✓	CoCa: Contrastive Captioners are Image-Text Foundation Models			2022	ALIGN JFT-
3	**Model soups** (BASIC-L)	90.98%		2440M			✓	Model soups: averaging weights of multiple fine-tuned models improves			2022	ALIGN Conv+Tran

ImageNet benchmark https://paperswithcode.com/sota/image-classification-on-imagenet

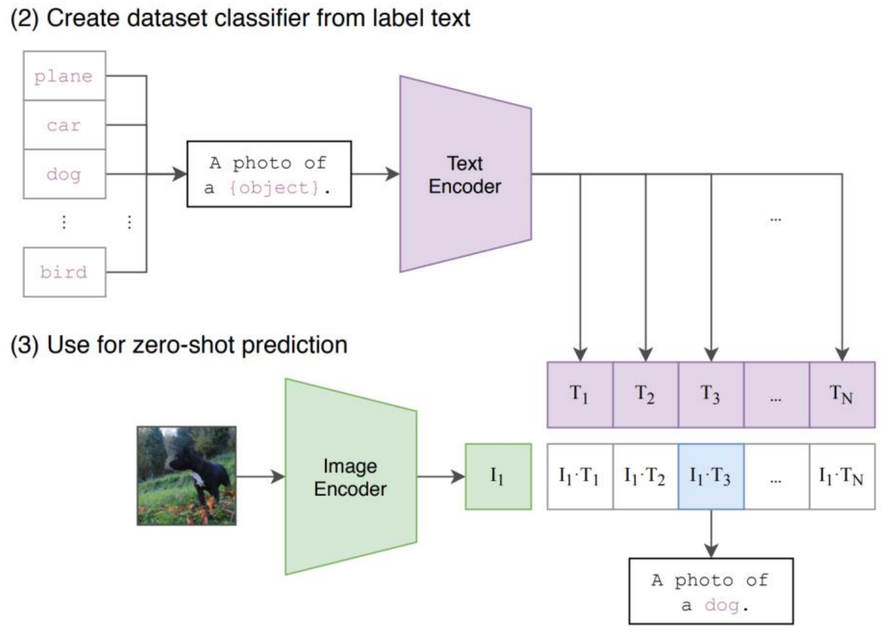
Conventional Recipe: Pre-training and Fine-tuning

- Linear probing before Fine-tuning (**LP-FT**)
  - Zero-shot initialization can be used with CLIP/ALIGN text encoder.



	(a) Fine-tuning	(b) Linear probing	(c) LP-FT
ID test	85.1%	82.9%	**85.7%**
OOD test	59.3%	66.2%	**68.9%**

Figure 1: Given a good feature extractor (top-left), a randomly initialized head is added to map features to outputs and we can (a) fine-tune all the model parameters or (b) linear-probe, which freezes the feature extractor and trains only the head. We run experiments on ten distribution shifts. Fine-tuning does well when the test example is sampled from the fine-tuning distribution (ID), but can underperform on test examples sampled from OOD distributions (when the distribution shift is large). (c) Our theory indicates that fine-tuning can distort the pretrained feature extractor and lead to poor OOD accuracy, but initializing with a linear probed head can fix this—empirically LP-FT gets better accuracies both ID and OOD.



(2) Create dataset classifier from label text
plane
car
dog
bird
A photo of a {object}.
Text Encoder
(3) Use for zero-shot prediction
T_1 T_2 T_3 ... T_N
Image Encoder
I_1
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
A photo of a dog.

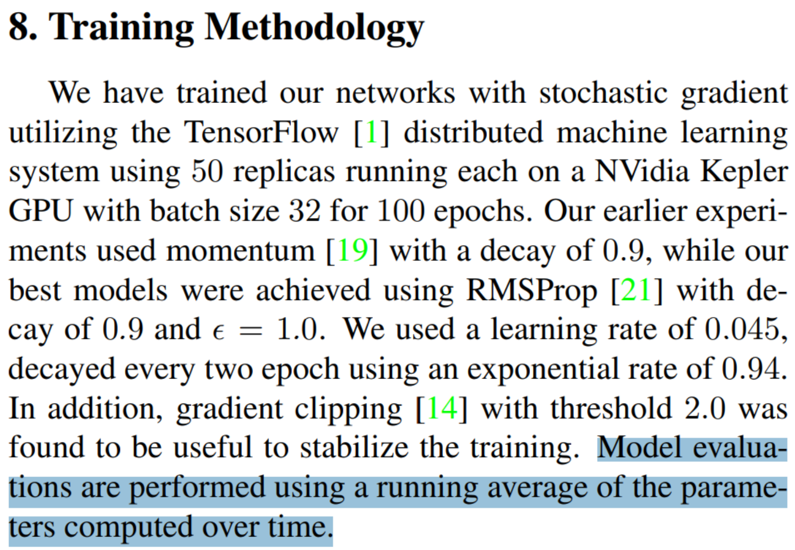
Zero-shot initialization of CLIP

Kumar, Ananya, et al. "Fine-tuning can distort pretrained features and underperform out-of-distribution." *ICLR.* 2022.
Radford, Alec, et al. "Learning transferable visual models from natural language supervision." *ICML.* 2021.

Fully Supervised Learning: Weight mixing approaches

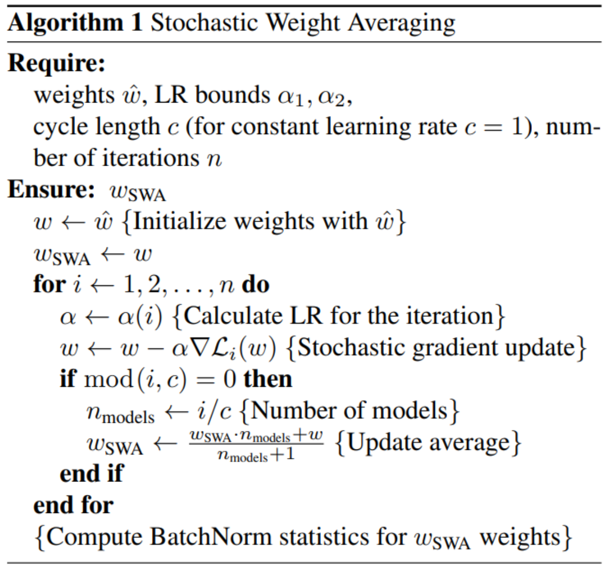
Averaging Model Weights: Same Optimization Trajectory

- 모델의 가중치들을 평균 내어 ensemble의 효과를 얻는 것은 Inception-v3부터 사용된 방법론이다.
  - EMA의 형태로 주로 사용되며, GAN이나 Diffusion model에서도 안정적인 결과를 얻기 위해 종종 사용된다.
- 대부분의 모델의 가중치를 평균 내는 방법론들은 동일한 optimization trajectory에 존재하는 모델들의 가중치를 평균 내어 모델의 variance를 줄인다. E.g., EMA, SWA, SGDR, etc.



8. Training Methodology

We have trained our networks with stochastic gradient utilizing the TensorFlow [1] distributed machine learning system using 50 replicas running each on a NVidia Kepler GPU with batch size 32 for 100 epochs. Our earlier experiments used momentum [19] with a decay of 0.9, while our best models were achieved using RMSProp [21] with decay of 0.9 and $\epsilon = 1.0$. We used a learning rate of 0.045, decayed every two epoch using an exponential rate of 0.94. In addition, gradient clipping [14] with threshold 2.0 was found to be useful to stabilize the training. Model evaluations are performed using a running average of the parameters computed over time.



Algorithm 1 Stochastic Weight Averaging

```
Require:
  weights $\hat{w}$, LR bounds $\alpha_1, \alpha_2$,
  cycle length $c$ (for constant learning rate $c = 1$), number of iterations $n$
Ensure: $w_{\text{SWA}}$
  $w \leftarrow \hat{w}$ {Initialize weights with $\hat{w}$}
  $w_{\text{SWA}} \leftarrow w$
  for $i \leftarrow 1, 2, \ldots, n$ do
    $\alpha \leftarrow \alpha(i)$ {Calculate LR for the iteration}
    $w \leftarrow w - \alpha \nabla \mathcal{L}_i(w)$ {Stochastic gradient update}
    if $\mathrm{mod}(i, c) = 0$ then
      $n_{\text{models}} \leftarrow i/c$ {Number of models}
      $w_{\text{SWA}} \leftarrow \frac{w_{\text{SWA}} \cdot n_{\text{models}} + w}{n_{\text{models}} + 1}$ {Update average}
    end if
  end for
  {Compute BatchNorm statistics for $w_{\text{SWA}}$ weights}
```

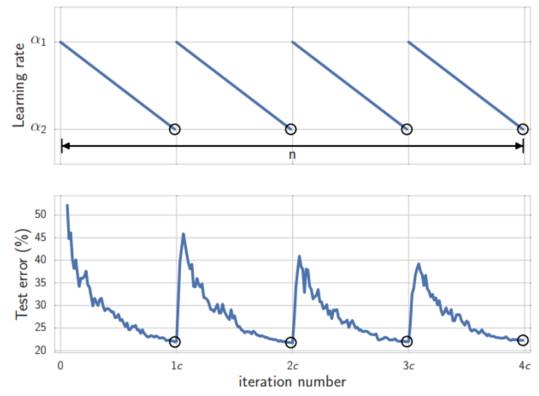


Figure 2: **Top**: cyclical learning rate as a function of iteration. **Bottom**: test error as a function of iteration for cyclical learning rate schedule with Preactivation-ResNet-164 on CIFAR-100. Circles indicate iterations corresponding to the minimum learning rates.

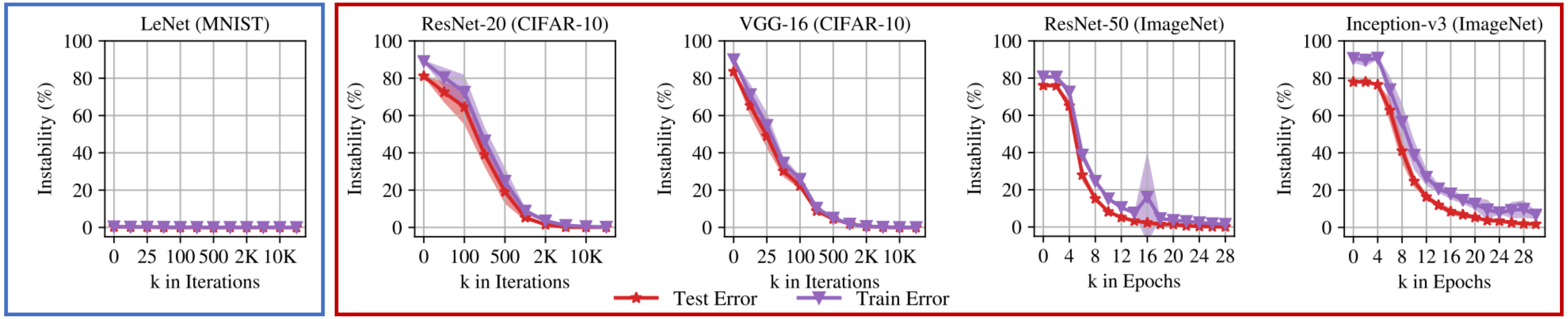
EMA in Inception-v3 Stochastic Weight Average

Szegedy, Christian, et al. "Rethinking the inception architecture for computer vision." *CVPR*. 2016.
Izmailov, Pavel, et al. "Averaging weights leads to wider optima and better generalization." *UAI.* 2018.
Loshchilov, Ilya, and Frank Hutter. "Sgdr: Stochastic gradient descent with warm restarts." *ICLR.* 2017.

Averaging Model Weights: Different Optimization Trajectory

- 다양한 데이터셋에 대해서 k번째 iteration까지 optimization trajectory를 공유했을 때, 최종 학습된 모델들의 weight를 평균 낸 weight의 성능 그래프.
- 제일 단순한 **MNIST**를 제외하고는 모두 optimization trajectory를 충분히 공유하지 않았을 때, **weight를 average하는 것은 굉장히 큰 성능 하락을 보여준다.**
    - Same hyperparameter configuration with the same random initialization, but different data order.
- **반대로 optimization trajectory를 충분히 공유한다면, 성능이 크게 하락하지 않은 것을 알 수 있다.**

Nagarajan and Kolter, 2019



Frankle et al., 2020

Figure 3. Linear interpolation instability when starting from step k. Each line is the mean and standard deviation across three initializations and three data orders (nine samples in total).

Nagarajan, Vaishnavh, and J. Zico Kolter. "Uniform convergence may be unable to explain generalization in deep learning." *NeurIPS.* 2019.
Frankle, Jonathan, et al. "Linear mode connectivity and the lottery ticket hypothesis." *ICML*, 2020.

Robust fine-tuning of zero-shot models

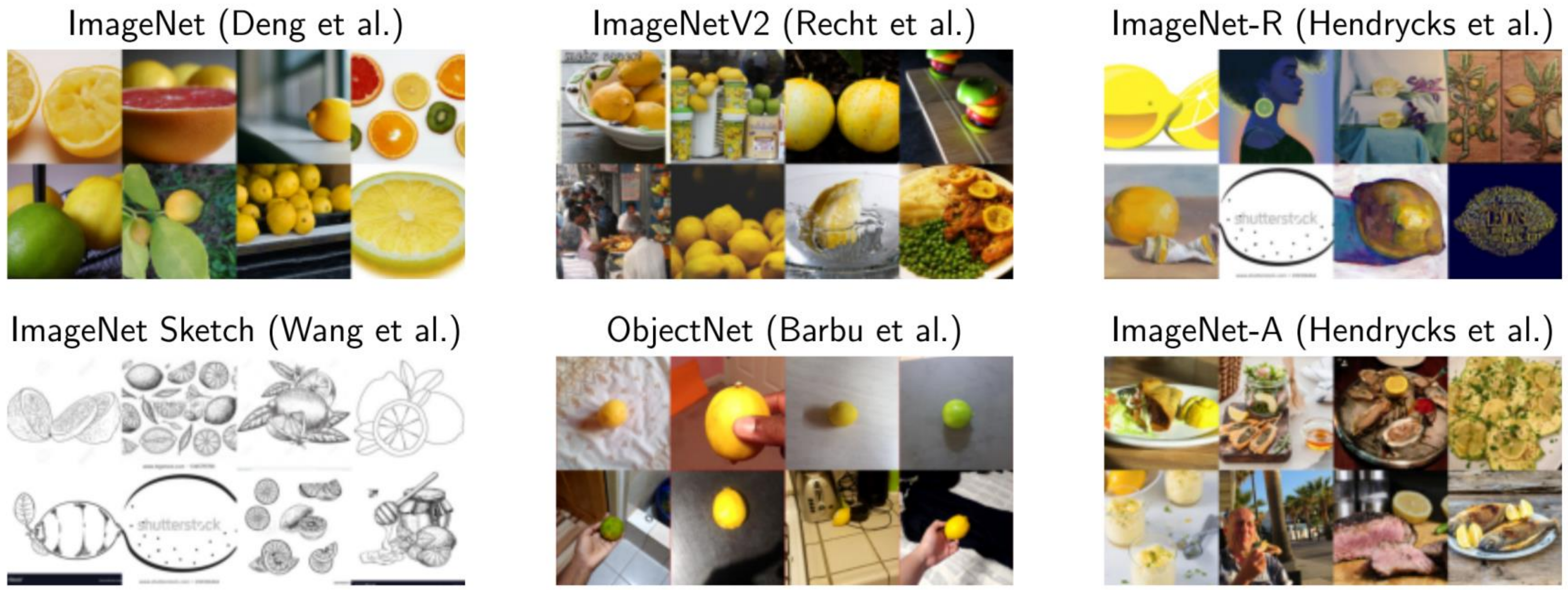
Mitchell Wortsman*, Gabriel Ilharco* et al.

University of Washington, OpenAI, Columbia University, Google Brain,
Allen Institute for Artificial Intelligence, Toyota Research Institute

CVPR 2022

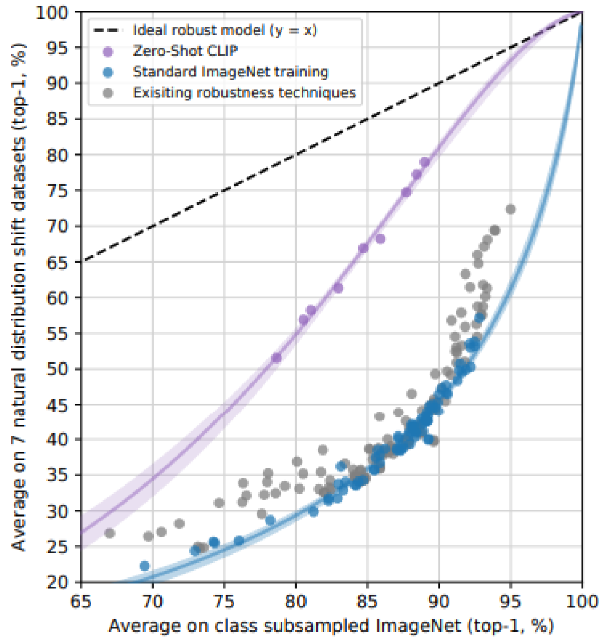
Distribution Shift

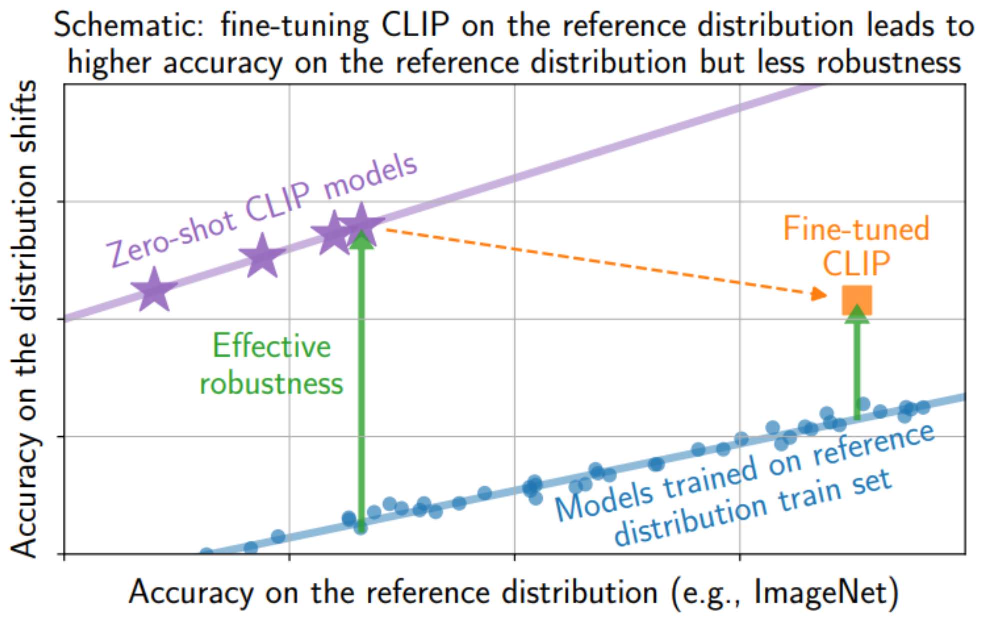
- ImageNet-V2: a reproduction of the ImageNet test set with distribution shift
- ImageNet-R: renditions (e.g., sculptures, paintings) for 200 ImageNet classes
- ImageNet-Sketch: which contains sketches instead of natural images
- ObjectNet: a test set of objects in various scenes with 113 classes overlapping with ImageNet
- ImageNet-A: a test set of natural images misclassified by a ResNet-50 for 200 ImageNet classes.


ImageNet (Deng et al.)
ImageNetV2 (Recht et al.)
ImageNet-R (Hendrycks et al.)
ImageNet Sketch (Wang et al.)
ObjectNet (Barbu et al.)
ImageNet-A (Hendrycks et al.)

Robustness of CLIP

- **Robustness: input distribution에 대한 robustness**로 같은 label space를 가지는 다른 domain의 이미지들을 분류하는 성능으로 robustness를 측정한다. E.g., ImageNetV2, ImageNet-Sketch, etc.
- CLIP은 대규모 데이터셋으로 학습했기 때문에 robustness가 다른 classification 모델들에 비해 확연히 좋은 것을 알 수 있다.
- **하지만, CLIP 역시 특정 데이터셋에 대해 fine-tuning을 진행하면 robustness가 떨어진다.**

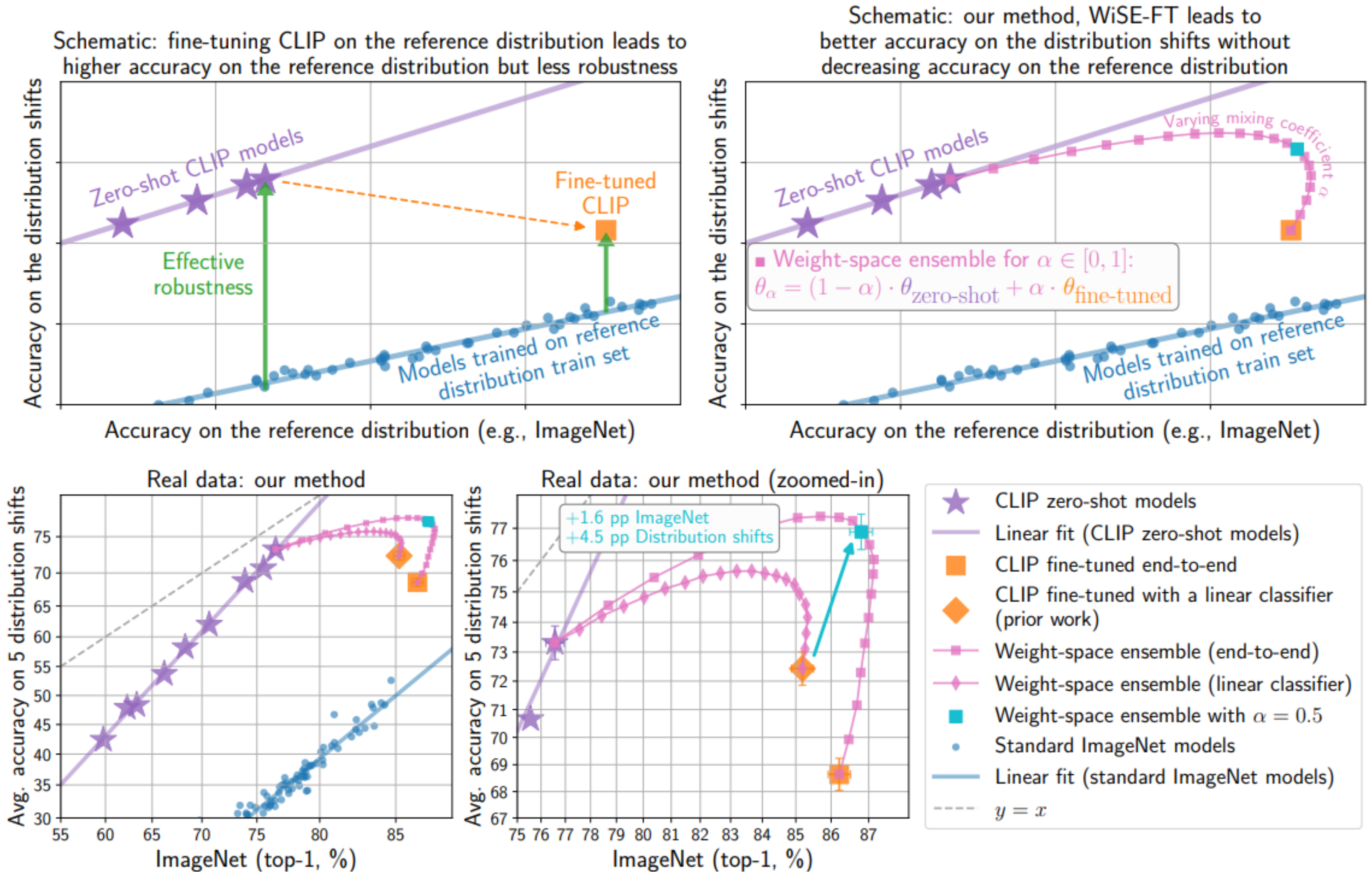




Schematic: fine-tuning CLIP on the reference distribution leads to higher accuracy on the reference distribution but less robustness
Zero-shot CLIP models
Fine-tuned CLIP
Effective robustness
Models trained on reference distribution train set
Accuracy on the distribution shifts
Accuracy on the reference distribution (e.g., ImageNet)

WiSE-FT: Weight-space Ensemble

- Zero-shot CLIP weight와 Fine-tuning의 weight를 interpolation하여 ensemble을 진행함.



각 end-point보다도 robustness와
in-distribution accuracy가 높아짐.
⇒ Model soups에서 visualization

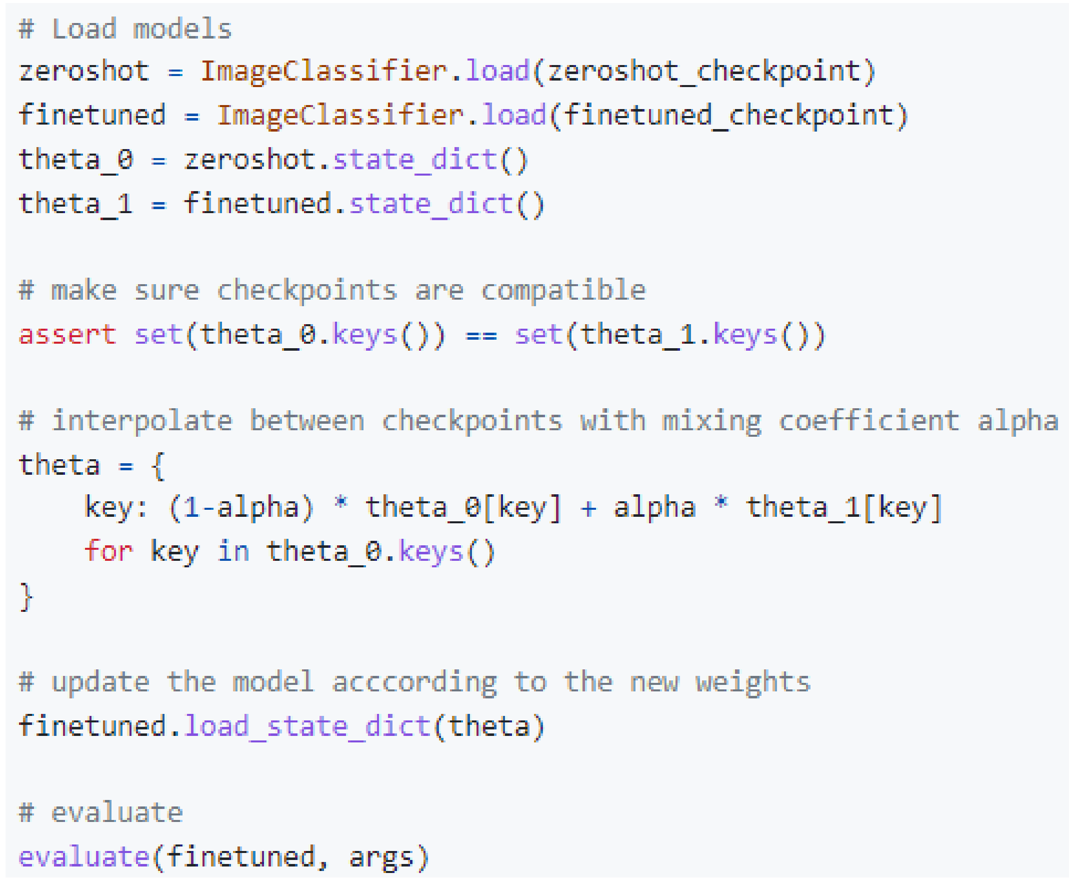
WiSE-FT: Weight-space Ensemble

- Formulate classifier
  - $f(x,\theta) = g(x,\mathbf{V}_{\text{enc}})^T\mathbf{W}_{\text{classifier}}$ where $\mathbf{W}_{\text{classifier}} \in \mathbb{R}^{d\times k}$ and $\theta = [\mathbf{V}_{\text{enc}},\mathbf{W}_{\text{classifier}}]$.
- Standard fine-tuning:
  - Solve $\underset{\theta}{\text{argmin}}\{\sum_{(x_i,y_i)\in S_{ref}^{tr}} \ell(f(x_i,\theta),y_i) + \lambda R(\theta)\}$.
- Weight-space ensembling: For a mixing coefficient $\alpha \in [0,1]$,
  - $\text{wse}(x,\alpha) = f(x,(1-\alpha)\cdot\theta_0 + \alpha\cdot\theta_1)$

- Linear probing에 대한 WiSE-FT는 traditional output-space ensembling과 정확히 동치
  - $f(x,(1-\alpha)\cdot\theta_0 + \alpha\cdot\theta_1) = (1-\alpha)\cdot g(x,\mathbf{V}_{\text{enc}})^T\mathbf{W}_{\text{zero-shot}} + \alpha\cdot g(x,\mathbf{V}_{\text{enc}})^T\mathbf{W}_{\text{classifier}}$
    $= (1-\alpha)\cdot f(x,\theta_0) + \alpha\cdot f(x,\theta_1)$
  - End-to-end fine-tuning은 동치는 아니지만, same optimization trajectory에서 averaging weight을 하던 것과 비슷한 작동 방식으로 해석할 수 있음.

Implementation

- https://github.com/mlfoundations/wise-ft#run-wise-ft



```
# Load models
zeroshot = ImageClassifier.load(zeroshot_checkpoint)
finetuned = ImageClassifier.load(finetuned_checkpoint)
theta_0 = zeroshot.state_dict()
theta_1 = finetuned.state_dict()

# make sure checkpoints are compatible
assert set(theta_0.keys()) == set(theta_1.keys())

# interpolate between checkpoints with mixing coefficient alpha
theta = {
    key: (1-alpha) * theta_0[key] + alpha * theta_1[key]
    for key in theta_0.keys()
}

# update the model acccording to the new weights
finetuned.load_state_dict(theta)

# evaluate
evaluate(finetuned, args)
```

Conv, BN, Self-Attention, … 가리지 않고 모든 layer 에 대해서 parameters를 average한다.

Comparison with Individual Fine-tuned Models

- 다양한 hyperparameters로 fine-tuning한 각각의 네트워크보다 weight-space ensembling이 in-distribution, robustness 모두 더 좋은 것을 확인할 수 있다.

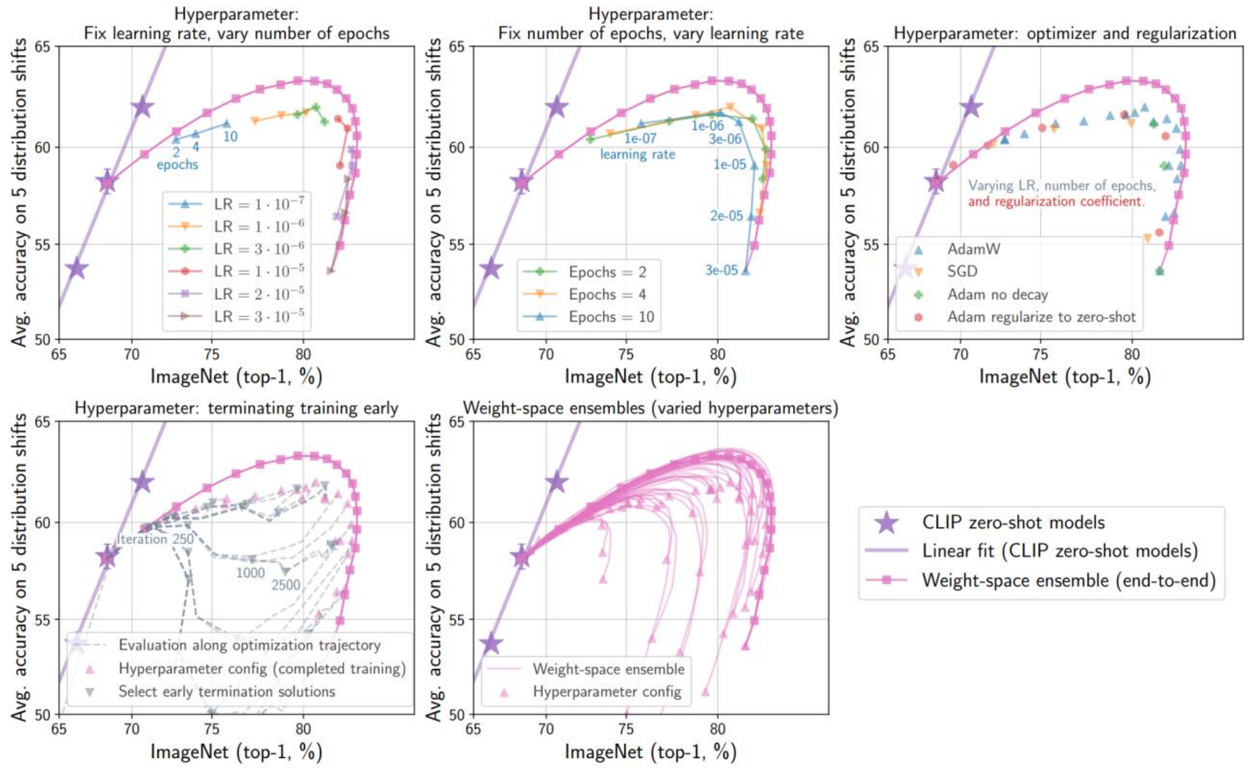
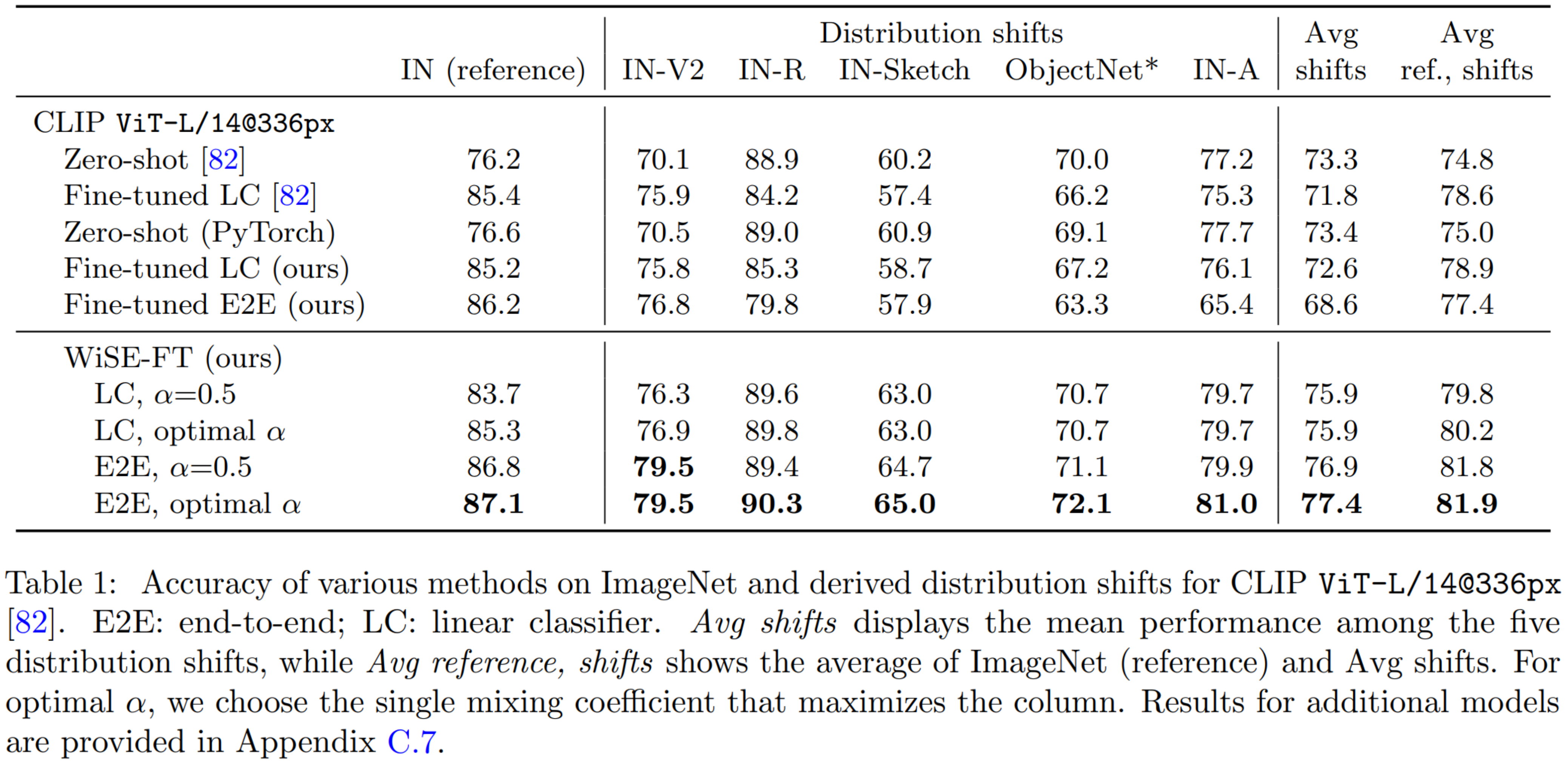


Figure 3: The robustness of fine-tuned models varies substantially under even small changes in hyperparameters. Applying WiSE-FT addresses this brittleness and can remove the trade-off between accuracy on the reference and shifted distributions. Results shown for CLIP ViT-B/16 fine-tuned with cosine-annealing learning rate schedule and all models in the top left and top middle plots are fine-tuned with AdamW [61]. Moreover, *regularize to zero-shot* appends the regularizer $\lambda\|\theta - \theta_0\|_2^2$ to the fine-tuning objective, where θ_0 are the parameters of the zero-shot model.

Results: CLIP trained on ImageNet



	IN (reference)	Distribution shifts IN-V2	IN-R	IN-Sketch	ObjectNet*	IN-A	Avg shifts	Avg ref., shifts
CLIP `ViT-L/14@336px`								
Zero-shot [82]	76.2	70.1	88.9	60.2	70.0	77.2	73.3	74.8
Fine-tuned LC [82]	85.4	75.9	84.2	57.4	66.2	75.3	71.8	78.6
Zero-shot (PyTorch)	76.6	70.5	89.0	60.9	69.1	77.7	73.4	75.0
Fine-tuned LC (ours)	85.2	75.8	85.3	58.7	67.2	76.1	72.6	78.9
Fine-tuned E2E (ours)	86.2	76.8	79.8	57.9	63.3	65.4	68.6	77.4
WiSE-FT (ours)								
LC, α=0.5	83.7	76.3	89.6	63.0	70.7	79.7	75.9	79.8
LC, optimal α	85.3	76.9	89.8	63.0	70.7	79.7	75.9	80.2
E2E, α=0.5	86.8	**79.5**	89.4	64.7	71.1	79.9	76.9	81.8
E2E, optimal α	**87.1**	**79.5**	**90.3**	**65.0**	**72.1**	**81.0**	**77.4**	**81.9**

Table 1: Accuracy of various methods on ImageNet and derived distribution shifts for CLIP `ViT-L/14@336px` [82]. E2E: end-to-end; LC: linear classifier. *Avg shifts* displays the mean performance among the five distribution shifts, while *Avg reference, shifts* shows the average of ImageNet (reference) and Avg shifts. For optimal α, we choose the single mixing coefficient that maximizes the column. Results for additional models are provided in Appendix C.7.

Results: Generalization

- ImageNet이 아닌 데이터셋에서도 잘 작동함.

	ImageNet	CIFAR10	CIFAR100	Cars	DTD	SUN397	Food101
Standard fine-tuning	86.2	98.6	92.2	91.6	81.9	80.7	94.4
WiSE-FT (α=0.5)	86.8 (+0.6)	99.3 (+0.7)	93.3 (+1.1)	93.3 (+1.7)	84.6 (+2.8)	83.2 (+2.5)	96.1 (+1.6)
WiSE-FT (opt. α)	87.1 (+0.9)	99.5 (+0.8)	93.4 (+1.2)	93.6 (+2.0)	85.2 (+3.3)	83.3 (+2.6)	96.2 (+1.8)

Table 2: Beyond robustness, WiSE-FT can improve accuracy after fine-tuning on several datasets.

- CLIP외의 foundation models에서도 조금씩 다르기는 하지만 비슷한 경향을 띔.

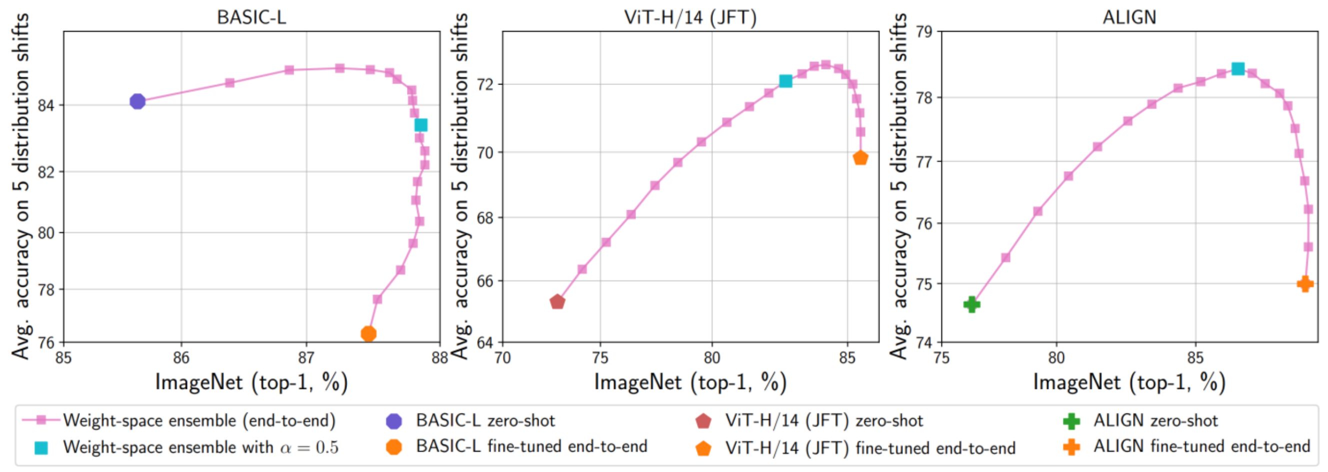
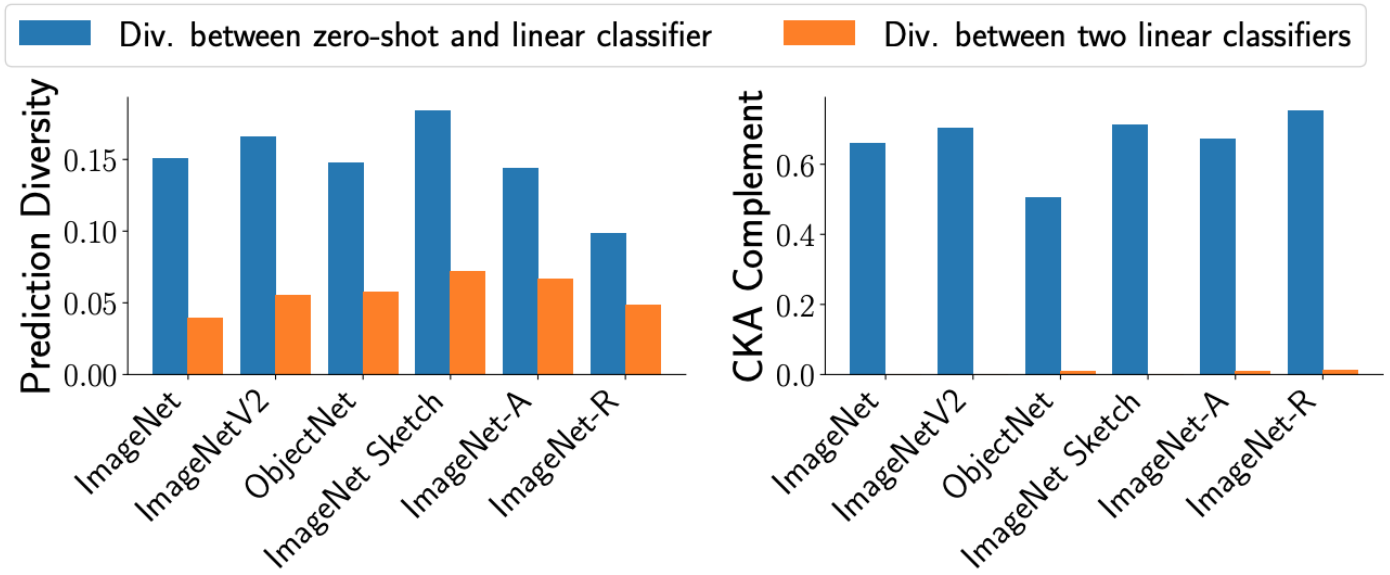


Figure 4: WiSE-FT applied to BASIC-L [77], a ViT-H/14 [21] model pre-trained on JFT-300M [93] and ALIGN [45].

Zero-shot and fine-tuned models are complementary

Zero-shot and fine-tuned models are diverse.

- 두 모델의 결과가 diverse하지 않으면 ensemble의 효과가 없음.
- **Prediction diversity:** 두 모델의 예측이 다른데 둘 중 하나는 정답일 확률.
- **Centered Kernel Alignment Complement:** NN representation의 similarity를 비교하는 방법 중 하나.



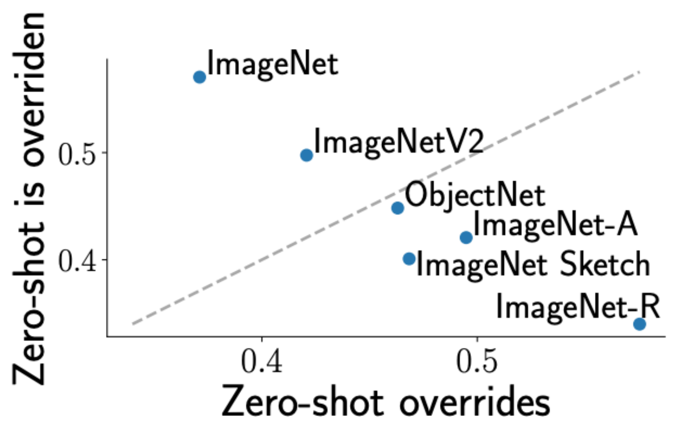
Measure diversity between two neural networks.

Kornblith, Simon, et al. "Similarity of neural network representations revisited." *ICML*. 2019.

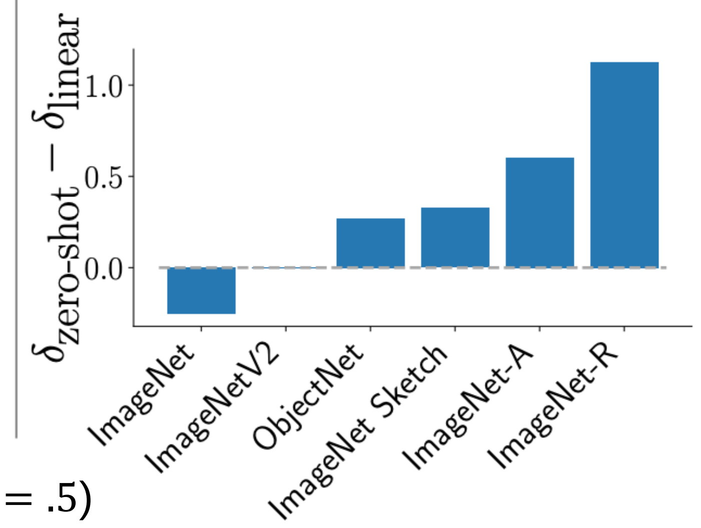
Zero-shot and fine-tuned models are complementary

Models are more confident where they excel.

- Zero-shot과 fine-tuned의 결과가 다를 때 average weight의 결과가 zero-shot을 따르면 Zero-shot overrides, fine-tuned의 결과를 따르면 Zero-shot is overridden.
- Margin δ: 가장 큰 logit과 두 번째로 큰 logit의 차이
  - Softmax 통과 시 confidence로 해석할 수 있음.



Zero-shot Overrides and Overridden ($\alpha = .5$)



Margin of various datasets.

- Zero-shot, fine-tuned model 각각 더 정확한 곳에서 confidence가 높음.
- 또한, confidence가 높을 때, logit ensemble 처럼 weight mixing 후 logit이 지배적임.

An Error Landscape Perspective

- Observation 1.

$$\mathrm{Acc}\big((1-\alpha)\cdot\theta_0+\alpha\cdot\theta_1\big) \geq (1-\alpha)\cdot\mathrm{Acc}(\theta_0)+\alpha\cdot\mathrm{Acc}(\theta_1)$$

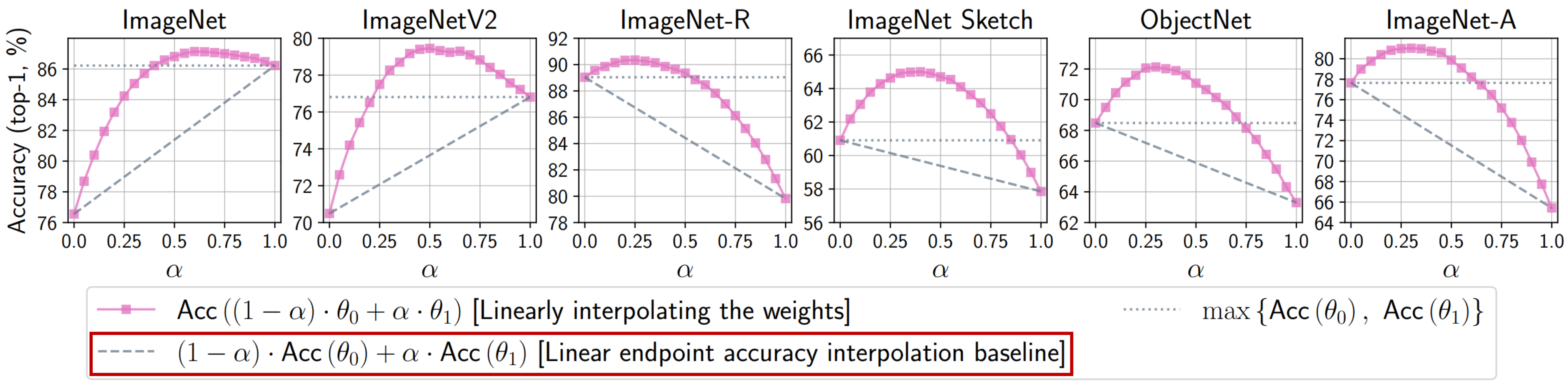


Figure 6: On ImageNet and the main distribution shifts we consider, linearly interpolating between the weights of θ_0 and θ_1 exceeds the baseline of linearly interpolating the accuracies of the two models for all α (Observation 1). Moreover, there exists an α for which WiSE-FT outperforms both the zero-shot and fine-tuned models (Observation 2).

Model soups에서 더 자세히 다룰 예정

An Error Landscape Perspective

- Observation 2.

$$\mathrm{Acc}\big((1-\alpha)\cdot\theta_0+\alpha\cdot\theta_1\big)\geq \max\{\mathrm{Acc}(\theta_0),\mathrm{Acc}(\theta_1)\}$$

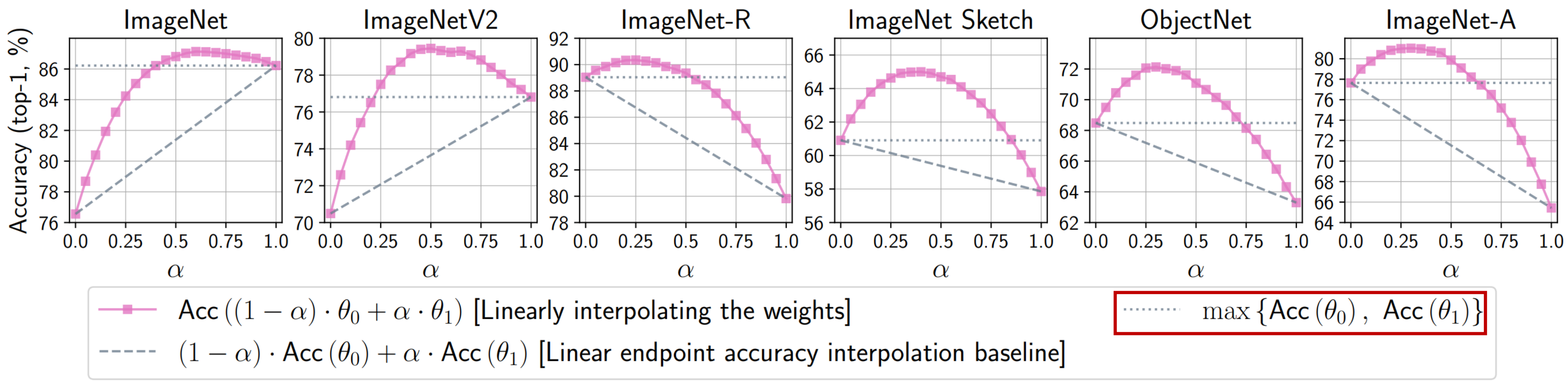


Figure 6: On ImageNet and the main distribution shifts we consider, linearly interpolating between the weights of θ_0 and θ_1 exceeds the baseline of linearly interpolating the accuracies of the two models for all α (Observation 1). Moreover, there exists an α for which WiSE-FT outperforms both the zero-shot and fine-tuned models (Observation 2).

Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time

Mitchell Wortsman et al.

University of Washington, Columbia University, Google Brain,
Meta AI Research, Tel Aviv University.

ICML 2022

Presented by Minho Park

Model Soups: Performance

- Model soups은 WiSE-FT의 후속 논문으로 additional training cost 없이 best individual model 보다도 성능이 높다.

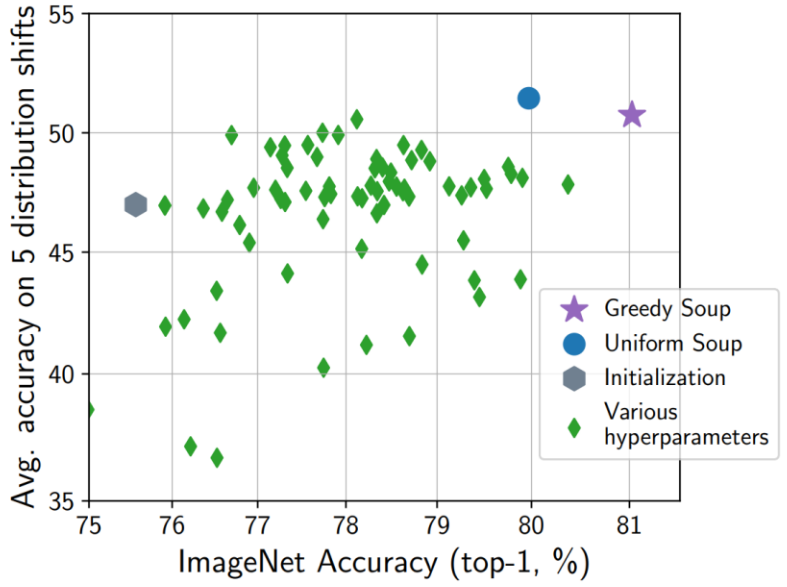


Figure 1: *Model soups* improve accuracy over the best individual model when performing a large, random hyperparameter search for fine-tuning a CLIP ViT-B/32 model on ImageNet. The *uniform soup* (blue circle) averages all fine-tuned models (green diamonds) in a random hyperparameter search over learning rate, weight-decay, iterations, data augmentation, mixup, and label smoothing. The *greedy soup* adds models sequentially to the model soup, keeping a model in the soup if accuracy on the held-out validation set does not decrease.

Method	ImageNet acc. (top-1, %)	Distribution shifts
ViT-G (Zhai et al., 2021)	90.45	–
CoAtNet-7 (Dai et al., 2021)	90.88	–
Our models/evaluations based on ViT-G:		
ViT-G (reevaluated)	90.47	82.06
Best model in hyperparam search	90.78	84.68
Greedy soup	**90.94**	**85.02**

Table 1: *Model soups* improve accuracy over the best individual model when fine-tuning a JFT-3B pre-trained ViT-G/14 model on ImageNet. Instead of selecting the best model from a hyperparameter sweep during fine-tuning, *model soups* average the weights of multiple fine-tuned models. To evaluate performance under distribution shift we consider average accuracy on ImageNet-V2, ImageNet-R, ImageNet-Sketch, ObjectNet, and ImageNet-A. Additional details are provided by Table 4 and Section 3.3.2, while analogous results for BASIC (Pham et al., 2021) are in Appendix C.

Uniform Soup

- 본 연구의 primary methods.
- Uniform soup은 모든 fine-tuned models θ_i의 평균을 내어 ensemble model의 weight를 계산한다.

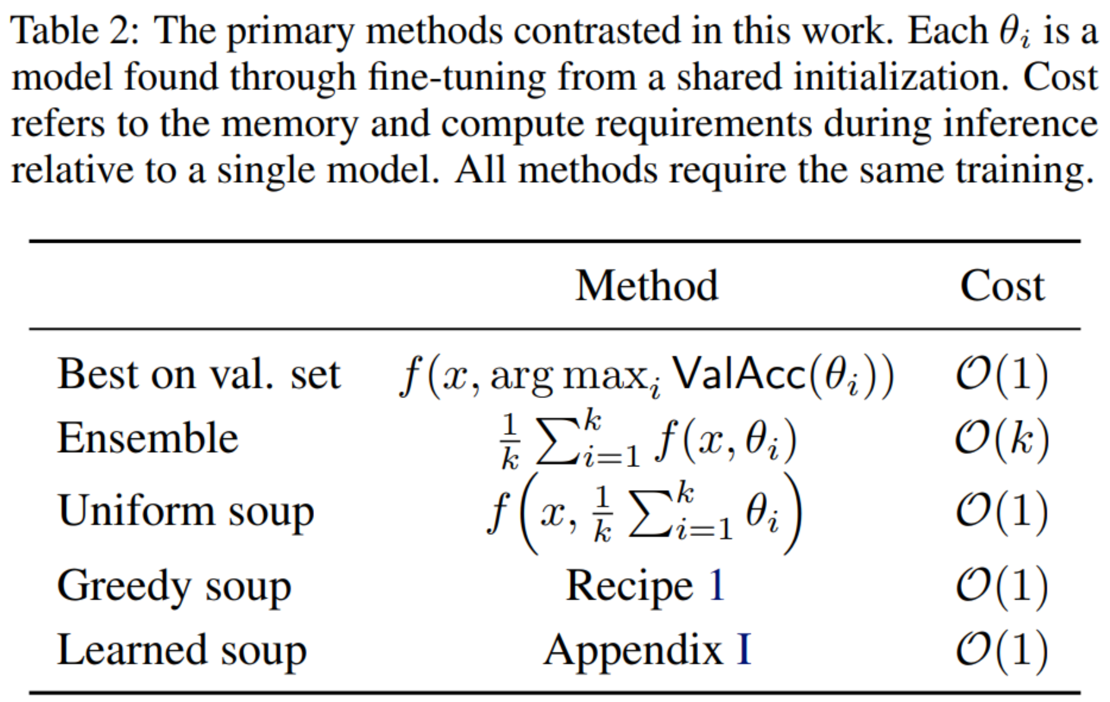
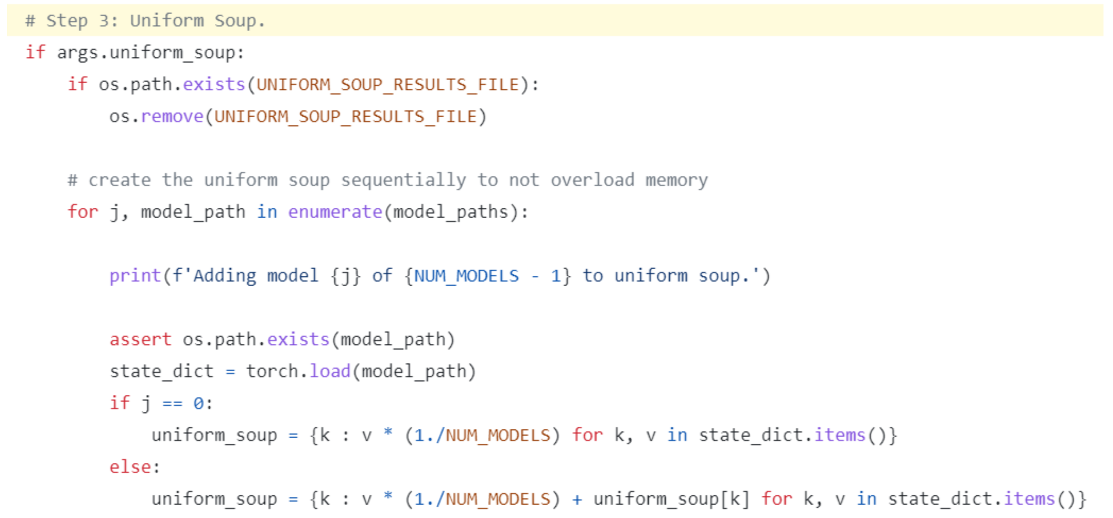


Table 2: The primary methods contrasted in this work. Each θ_i is a model found through fine-tuning from a shared initialization. Cost refers to the memory and compute requirements during inference relative to a single model. All methods require the same training.

	Method	Cost
Best on val. set	$f(x, \arg\max_i \mathsf{ValAcc}(\theta_i))$	$\mathcal{O}(1)$
Ensemble	$\frac{1}{k}\sum_{i=1}^{k} f(x, \theta_i)$	$\mathcal{O}(k)$
Uniform soup	$f\left(x, \frac{1}{k}\sum_{i=1}^{k} \theta_i\right)$	$\mathcal{O}(1)$
Greedy soup	Recipe 1	$\mathcal{O}(1)$
Learned soup	Appendix I	$\mathcal{O}(1)$

+ Greedy Ensemble



```
# Step 3: Uniform Soup.
if args.uniform_soup:
    if os.path.exists(UNIFORM_SOUP_RESULTS_FILE):
        os.remove(UNIFORM_SOUP_RESULTS_FILE)

    # create the uniform soup sequentially to not overload memory
    for j, model_path in enumerate(model_paths):

        print(f'Adding model {j} of {NUM_MODELS - 1} to uniform soup.')

        assert os.path.exists(model_path)
        state_dict = torch.load(model_path)
        if j == 0:
            uniform_soup = {k : v * (1./NUM_MODELS) for k, v in state_dict.items()}
        else:
            uniform_soup = {k : v * (1./NUM_MODELS) + uniform_soup[k] for k, v in state_dict.items()}
```

Average every parameters
Weight, bias, batch norm, …

Intuition: Error Landscape Visualization

- Use zero-shot initialization θ_0, and two different fine-tuned models θ_1, θ_2,
  - 세 점이 있으니 평면을 만들 수 있고, 이제 loss landscape을 2D로 visualization할 수 있다.

1. Interpolating the weights of two fine-tuned solutions can **get higher accuracy than individual model.**

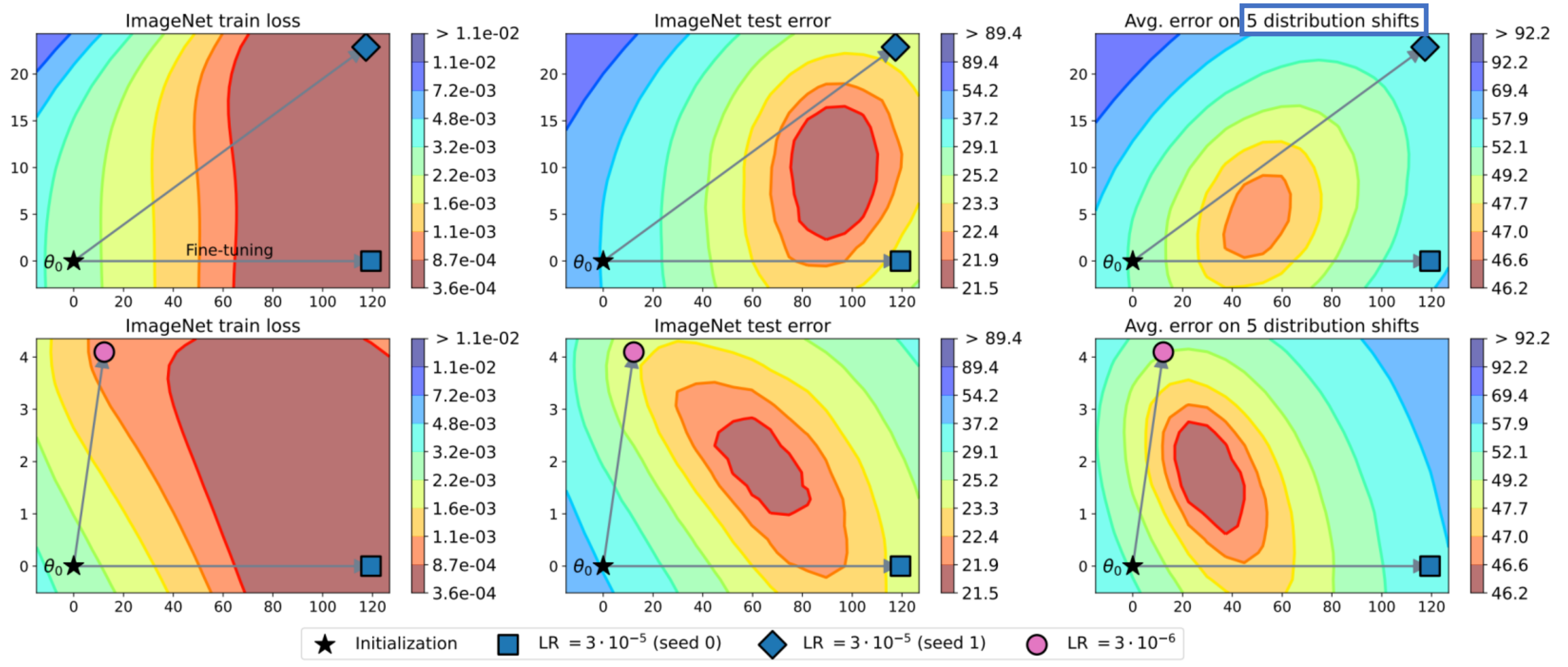
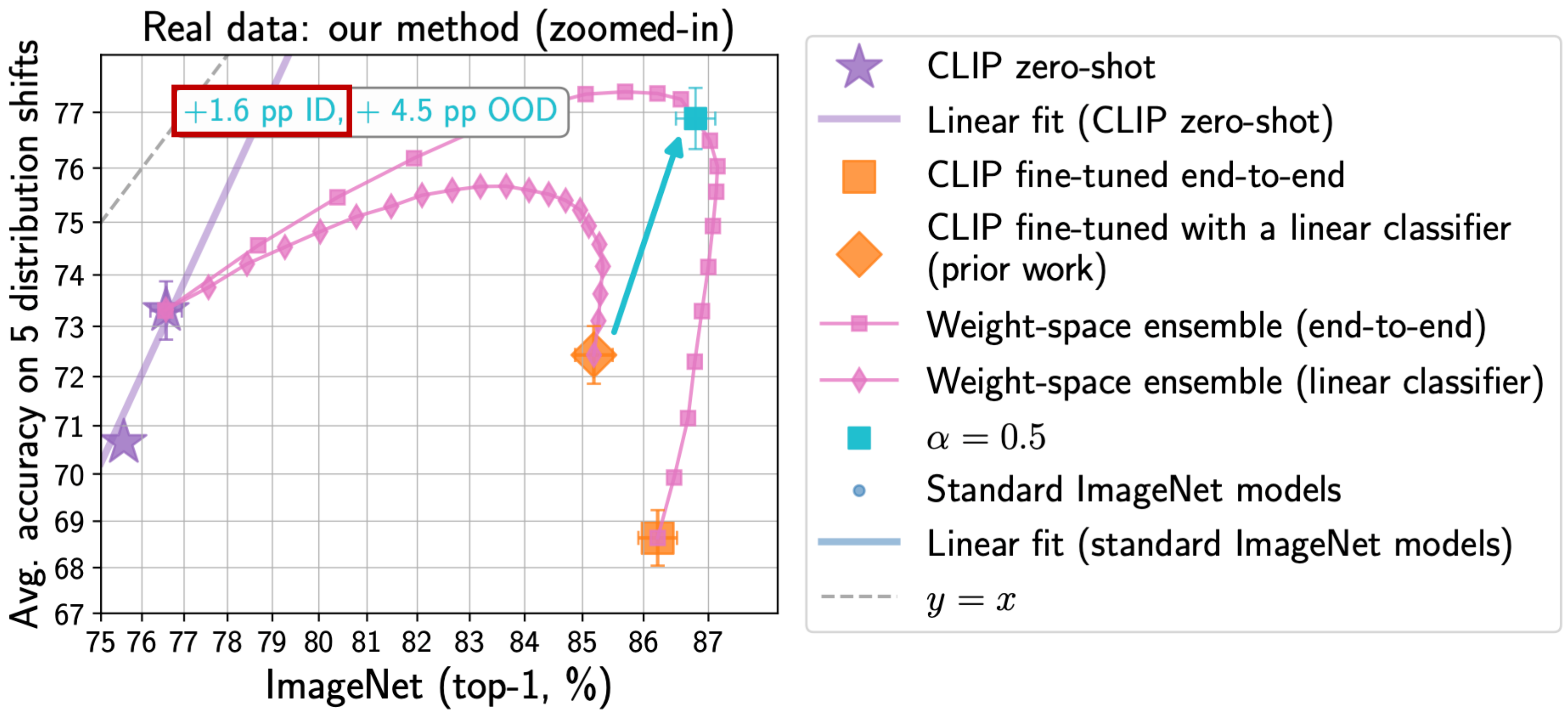


Figure 2: The solution with the highest accuracy is often not a fine-tuned model but rather lies between fine-tuned models. This figure shows loss and error on a two dimensional slice of the loss and error landscapes. We use the zero-shot initialization θ_0 and fine-tune twice (illustrated by the gray arrows), independently, to obtain solutions θ_1 and θ_2. As in Garipov et al. (2018), we obtain an orthonormal basis u_1, u_2 for the plane spanned by these models, and the x and y-axis show movement in parameter space in these directions, respectively.

5 distribution shifts
- ImageNetV2
- ImageNet-R
- ImageNet-Sketch
- ObjectNet
- ImageNet-A

Related Work: WiSE-FT

- WiSE-FT에서는 robustness에 주목하였지만, Model soups에서는 in-distribution accuracy에 주목함.
- **이를 잘 이용하면 zero-cost로 state-of-the-art performance를 기록할 수 있지 않을까?**



Same author

Wortsman, Mitchell, et al. "Robust fine-tuning of zero-shot models." *CVPR*. 2022.

Intuition: Error Landscape Visualization

- **너무 큰 learning rate으로 학습된** fine-tuned model과 interpolation을 진행하면 interpolation 중간에 error가 커지는 현상을 보인다.
  - 본 실험에서 사용한 AdamW의 경우, $3 \cdot 10^{-4}$는 너무 큰 learning rate임.

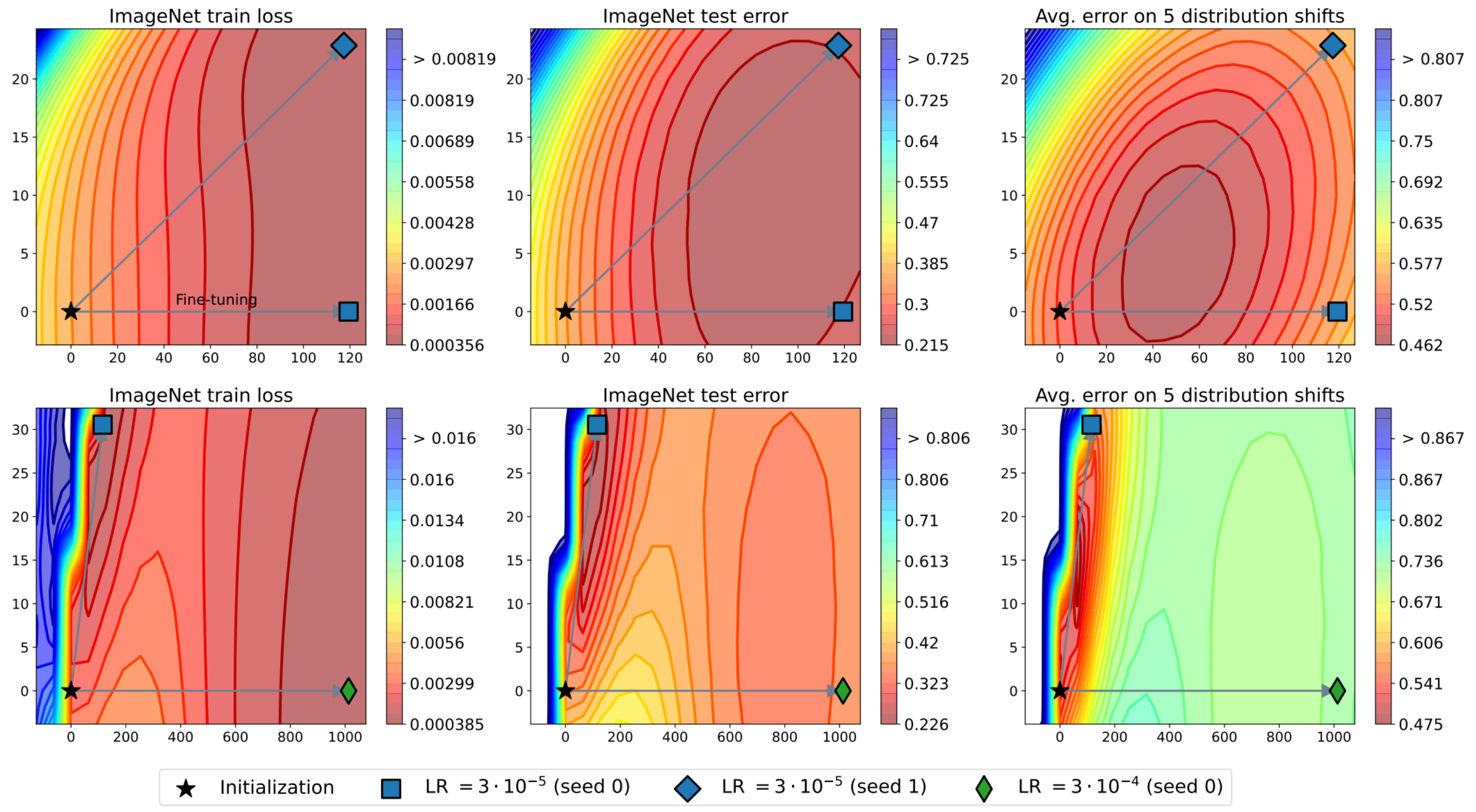


Figure J.1: Replicating Figure 2 with a 10x larger learning rate instead of 10x smaller in the second row.

Intuition: The Advantage of Averaging Solution

2. **More uncorrelated solutions** may lead to higher accuracy on the linear interpolation path.

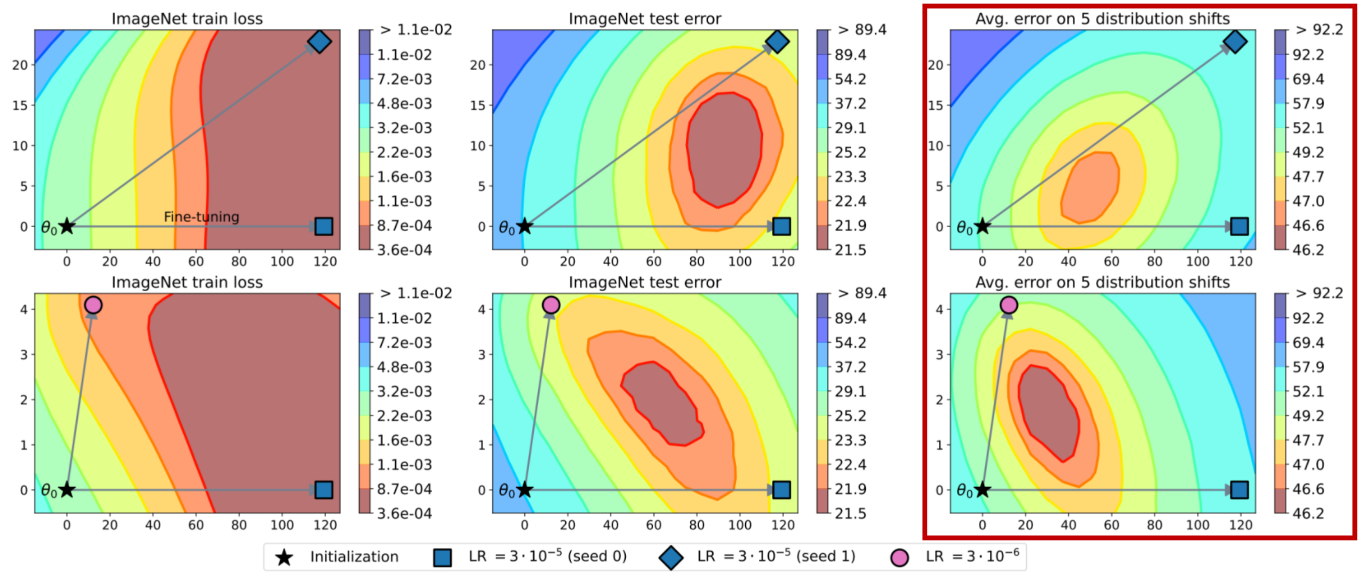


Figure 2: The solution with the highest accuracy is often not a fine-tuned model but rather lies between fine-tuned models. This figure shows loss and error on a two dimensional slice of the loss and error landscapes. We use the zero-shot initialization θ_0 and fine-tune twice (illustrated by the gray arrows), independently, to obtain solutions θ_1 and θ_2. As in Garipov et al. (2018), we obtain an orthonormal basis u_1, u_2 for the plane spanned by these models, and the x and y-axis show movement in parameter space in these directions, respectively.

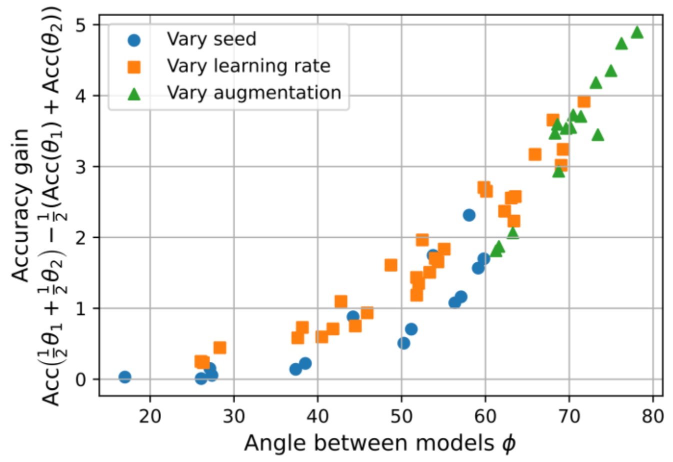


Figure 3: The advantage of averaging solutions (y-axis) is correlated with the angle ϕ between between solutions, while varying hyperparameter configurations between pairs enables a larger ϕ. Each point corresponds to a pair of models θ_1, θ_2 that are fine-tuned independently from a shared initialization θ_0 with different hyperparameter configurations. The angle ϕ between between solutions refers to the angle between $\theta_1 - \theta_0$ and $\theta_2 - \theta_0$ (i.e., the initialization is treated as the origin). Accuracy is averaged over ImageNet and the five distribution shifts described in Section 3.1.

How about $\mathrm{Acc}\left(\frac{1}{2}\theta_1 + \frac{1}{2}\theta_2\right) - \max\{\mathrm{Acc}(\theta_1), \mathrm{Acc}(\theta_2)\}$?

One Dimensional Hyperparameter Grids

- 하나의 hyperparameter를 기준으로 individual model들이 있을 때, 임의의 두 모델을 평균 낸 것과 best individual model 중 어느 것이 더 성능이 높을까?
- **The average of the endpoints often outperforms the best individual model in the grid.**
  - Learning rate 10^{-4}과 mixing하는 경우를 제외하고, 모든 경우 성능이 향상되었다.
  - Exception의 경우 이전 슬라이드인 landscape visualization with larger learning rate을 참고.

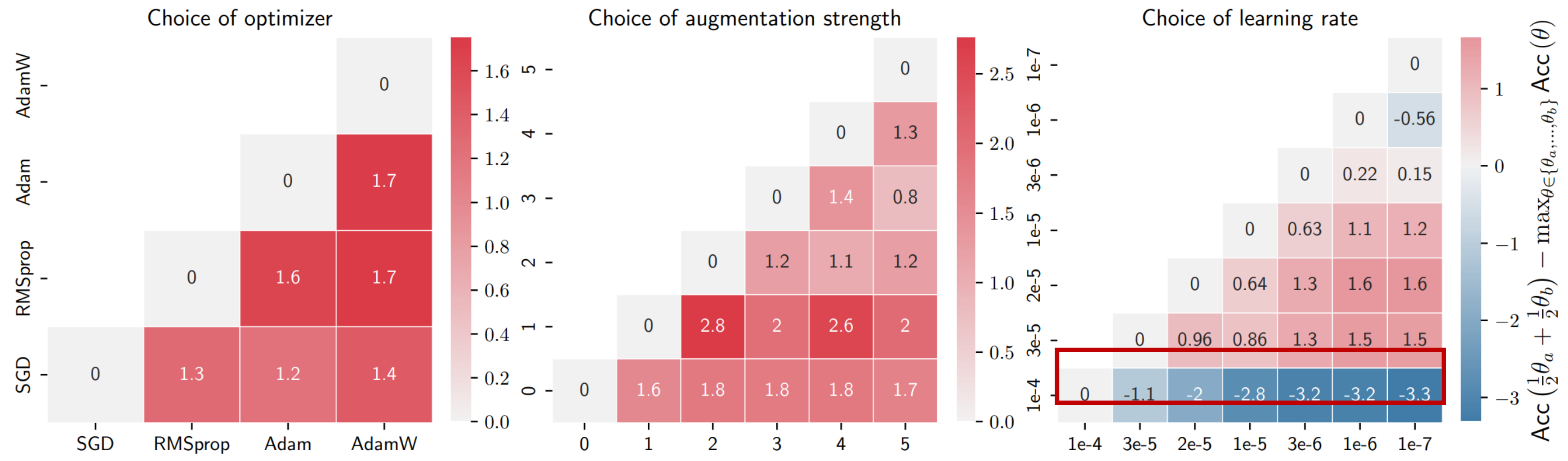
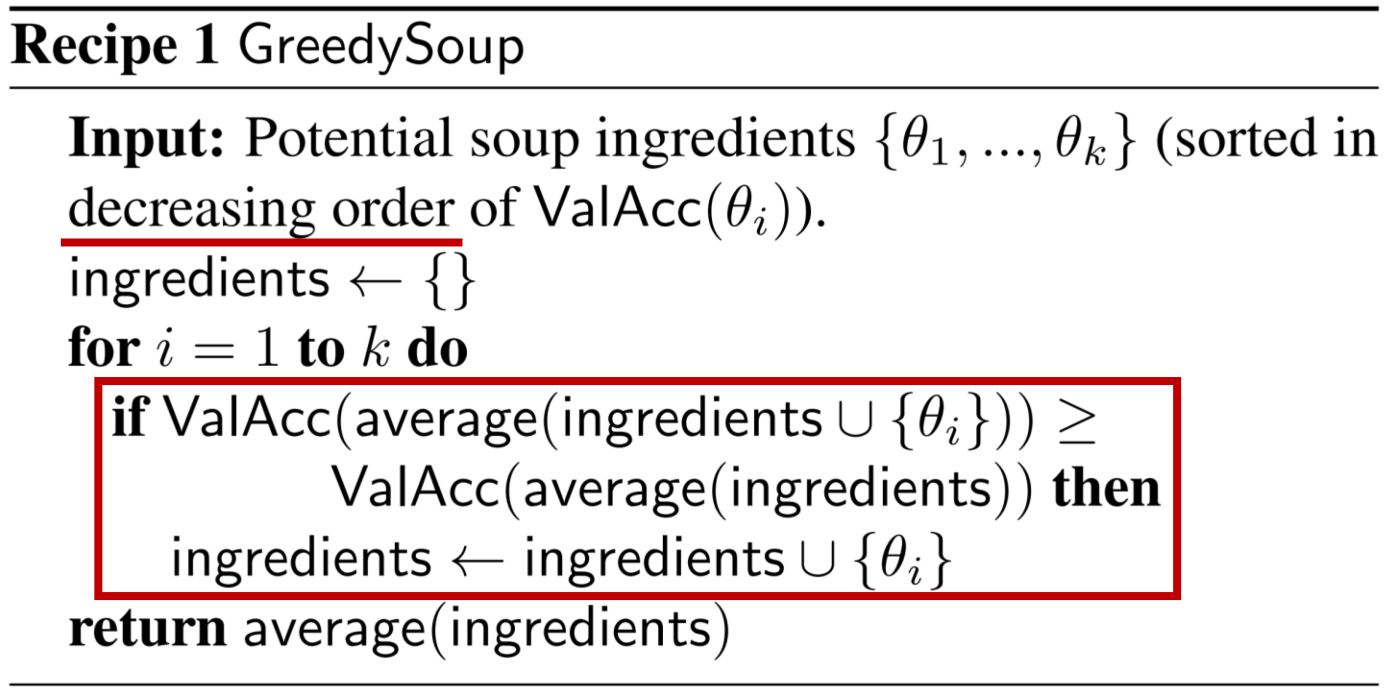


Figure F.1: Analysis of 1D hyperparameter grids, where the average of models at the endpoints often outperforms the best individual model in the grid. In particular, colors and numbers indicate the percentage point improvement obtained by averaging the models on the x and y axis versus taking the best individual model in the range between them. Results are shown for the CLIP ViT-B/32 model fine-tuned on ImagNet.

Advanced Recipe: Greedy Soup

- 가장 성능이 좋았던 모델 순으로 weight를 average 해보고 성능이 상승할 때에만 해당 모델을 저장하는 방법론을 제안함.
- 공정한 비교를 위해서 logit ensemble method에 대해서도 greedy ensemble을 소개함.



Recipe 1 GreedySoup

```
Input: Potential soup ingredients $\{\theta_1, ..., \theta_k\}$ (sorted in decreasing order of ValAcc($\theta_i$)).
ingredients $\leftarrow \{\}$
for $i = 1$ to $k$ do
    if ValAcc(average(ingredients $\cup \{\theta_i\}$)) $\geq$
                ValAcc(average(ingredients)) then
        ingredients $\leftarrow$ ingredients $\cup \{\theta_i\}$
return average(ingredients)
```

절대 best individual model보다
성능이 하락하지 않는 방법론

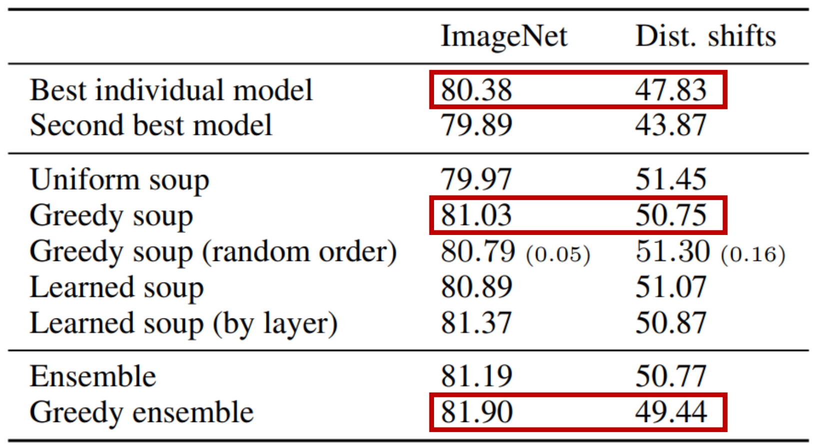
Advanced Recipe: Learned Soup

- Learned soup: 이전 까지는 전체 모델을 동일한 비율로 average 했지만, learnable parameter를 이용하여 섞는 비율을 조절하는 방법론.
  - Learned soup (by layer): layer 별로 learnable parameter를 두어 mixing.
  - Additional training cost가 필요함.

$$\arg\min_{\alpha\in\mathbb{R}^k,\, \beta\in\mathbb{R}} \sum_{j=1}^{n} \ell\left(\beta\cdot f\left(x_j, \sum_{i=1}^{k}\alpha_i\theta_i\right), y_j\right)$$

α: mixing coefficients
β: temperature scaling parameter

Table 3: Ablation on multiple methods from Table 2 and their variants when when fine-tuning CLIP ViT-B/32 with the random hyperparameter search described in Section 3.3.1. For "Greedy soup (random order)", we try three random model orders when running the greedy soup procedure (by default, models are sorted by decreasing held-out val accuracy). The "Learned soup" and its variants are descried in Appendix I. The *best* in *best individual model* refers to ImageNet accuracy.



	ImageNet	Dist. shifts
Best individual model	80.38	47.83
Second best model	79.89	43.87
Uniform soup	79.97	51.45
Greedy soup	81.03	50.75
Greedy soup (random order)	80.79 (0.05)	51.30 (0.16)
Learned soup	80.89	51.07
Learned soup (by layer)	81.37	50.87
Ensemble	81.19	50.77
Greedy ensemble	81.90	49.44

Ensemble은 추가 inference time이 필요

Ensemble Comparison

- Ensemble performance와 Model soups performance는 correlated되어 있음.
  - Large learning rate이 섞이지 않았을 때.
- ImageNet에서는 Ensemble이 더 높은 성능을 보이고 distribution shifts에서는 Model soups가 더 높은 성능을 보여줌.

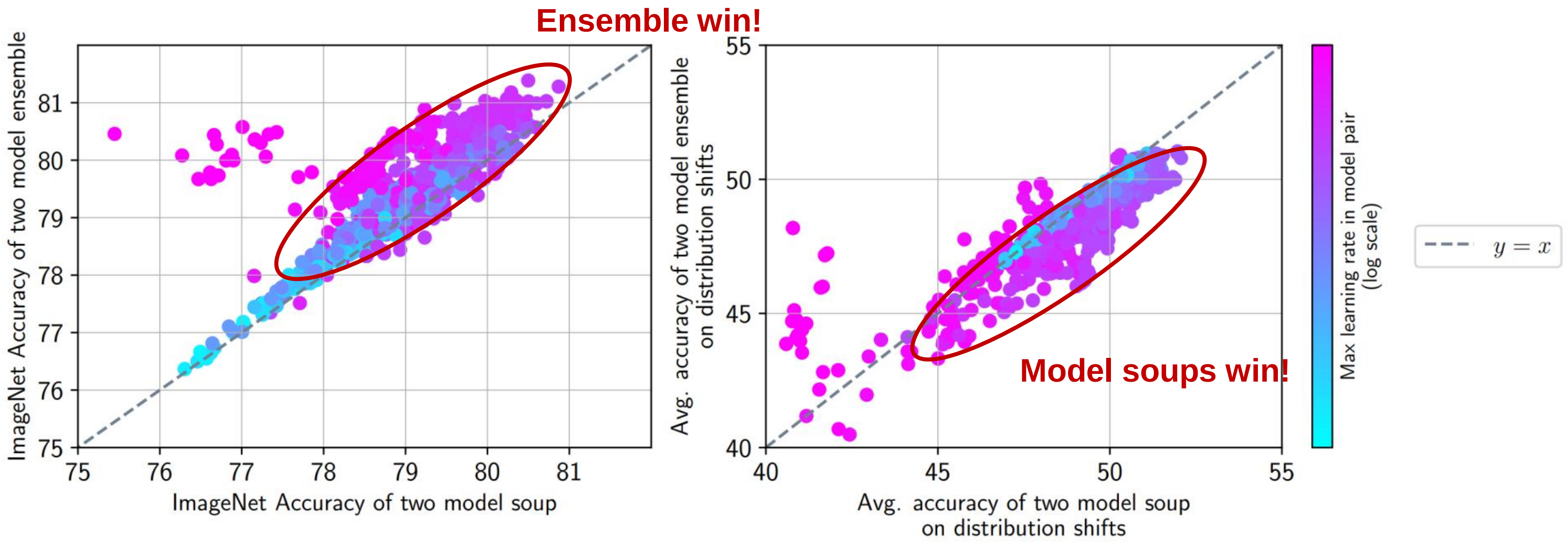


Figure 4: Ensemble performance is correlated with model soup performance. Each point on the scatter plot is a model pair with different hyperparameters. The x-axis is the accuracy when the weights of the two models are averaged (i.e., the two model soup) while the y-axis is the accuracy of the two model ensemble. Ensembles often perform slightly better than soups on ImageNet (left) while the reverse is true on the distribution shifts (right). Each model pair consists of two random greed diamonds from Figure 1.

Experiments: ImageNet and Distribution Shift

- ReaL and ObjectNet 데이터셋 에서만 test accuracy기준으로 Greedy soup보다 좋은 성능의 individual model이 존재하고 이 두 모델은 서로 다른 모델임.
  - 즉, Model soups을 이용하면 기존에 존재하던 모델보다 더 좋은 모델이 새로 만들어진 것임.

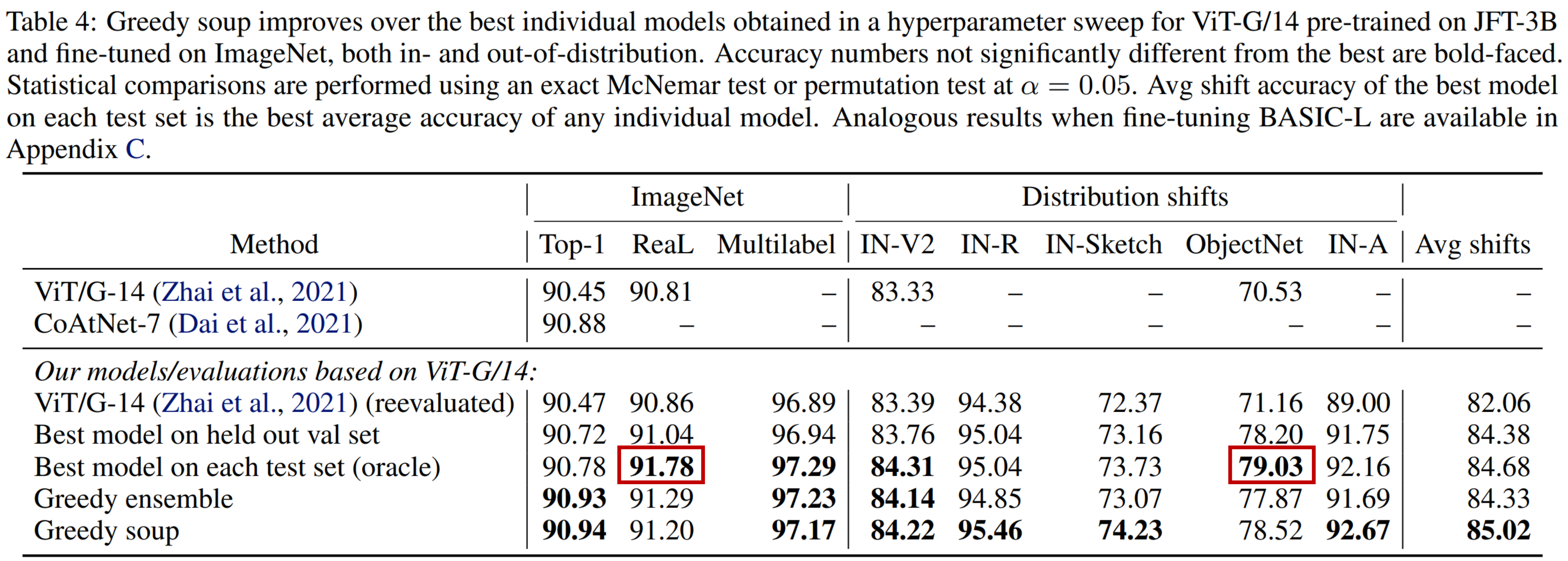


Table 4: Greedy soup improves over the best individual models obtained in a hyperparameter sweep for ViT-G/14 pre-trained on JFT-3B and fine-tuned on ImageNet, both in- and out-of-distribution. Accuracy numbers not significantly different from the best are bold-faced. Statistical comparisons are performed using an exact McNemar test or permutation test at $\alpha = 0.05$. Avg shift accuracy of the best model on each test set is the best average accuracy of any individual model. Analogous results when fine-tuning BASIC-L are available in Appendix C.

	ImageNet			Distribution shifts					
Method	Top-1	ReaL	Multilabel	IN-V2	IN-R	IN-Sketch	ObjectNet	IN-A	Avg shifts
ViT/G-14 (Zhai et al., 2021)	90.45	90.81	–	83.33	–	–	70.53	–	–
CoAtNet-7 (Dai et al., 2021)	90.88	–	–	–	–	–	–	–	–
Our models/evaluations based on ViT-G/14:									
ViT/G-14 (Zhai et al., 2021) (reevaluated)	90.47	90.86	96.89	83.39	94.38	72.37	71.16	89.00	82.06
Best model on held out val set	90.72	91.04	96.94	83.76	95.04	73.16	78.20	91.75	84.38
Best model on each test set (oracle)	90.78	**91.78**	**97.29**	**84.31**	95.04	73.73	**79.03**	92.16	84.68
Greedy ensemble	**90.93**	91.29	**97.23**	**84.14**	94.85	73.07	77.87	91.69	84.33
Greedy soup	**90.94**	91.20	**97.17**	**84.22**	**95.46**	**74.23**	78.52	**92.67**	**85.02**

58 ViT-G/14 models fine-tuned on ImageNet
Varying learning rate, decay schedule, loss function, minimum crop size, RandAugment, mixup, and CutMix.

Experiments: Additional Fine-tuning Dataset

- WILDS-FMoW, WILDS-iWildCam, CIFAR-10에 대해서도 ImageNet과 같은 방식의 실험을 진행했을 때, 마찬가지로 더 좋은 성능의 모델을 찾아낼 수 있었음.
  - Model soups는 ImageNet 외의 fine-tuning dataset에 대해서도 general하게 잘 작동하는 방법론임.

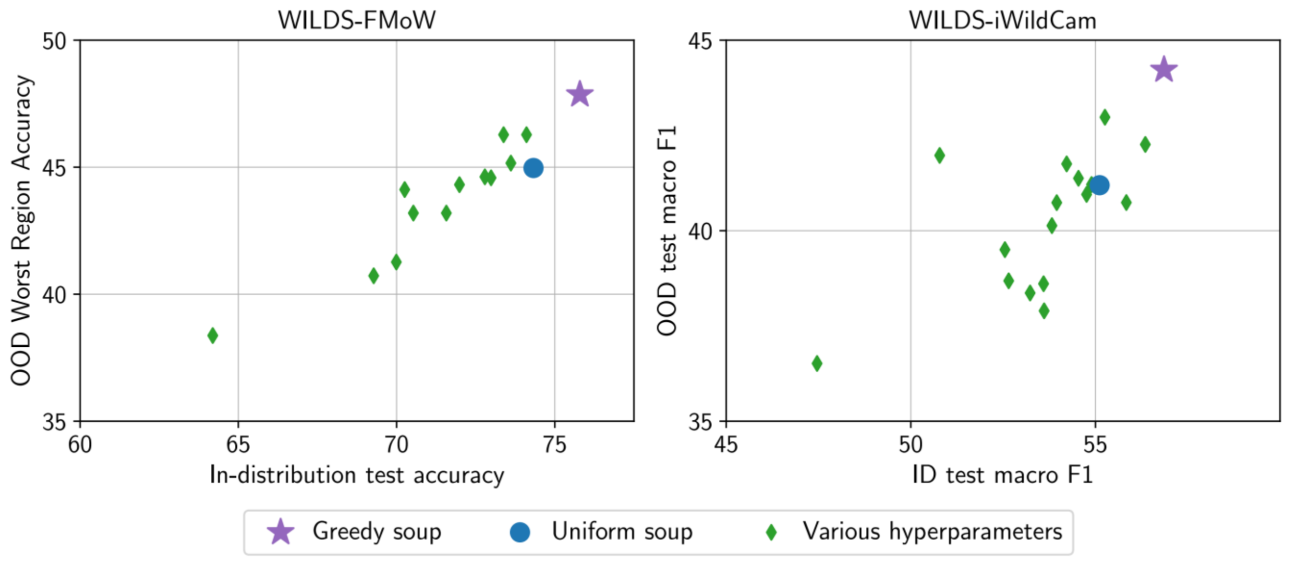


Figure G.1: Model soups improve accuracy when fine-tuning on the diverse classification tasks WILDS-FMoW (Koh et al., 2021; Christie et al., 2018) and WILDS-iWildCam (Koh et al., 2021; Beery et al., 2021). Results are shown for the CLIP ViT-L/14 model and a random hyperparameter search over learning rate, weight-decay, iterations, data augmentation, mixup, and label smoothing.

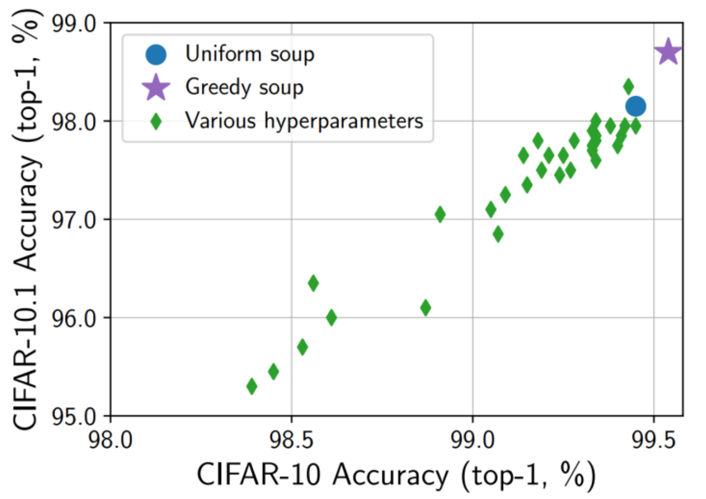


Figure G.2: Fine-tuning a CLIP ViT-L model on CIFAR-10 (Krizhevsky et al., 2009) with the random hyperparameter search described in Section J.2.1. The y-axis displays accuracy on CIFAR-10.1 (Recht et al., 2019), a reproduction of CIFAR-10 with a distribution shift.

Experiments: Additional Pre-training Dataset

- 본 논문의 대부분의 실험들은 fine-tuning dataset과 heterogeneous인 large-scale dataset에서 pre-training된 큰 모델에 대해서 진행되었음.
- ImageNet-22k pretrained ViT-B/32의 Greedy soup은 ImageNet에서의 성능 향상을 보여주었으나, **CLIP and ALIGN을 fine-tuning했을 때 보다 성능의 증가폭이 미미하다.** Limitation

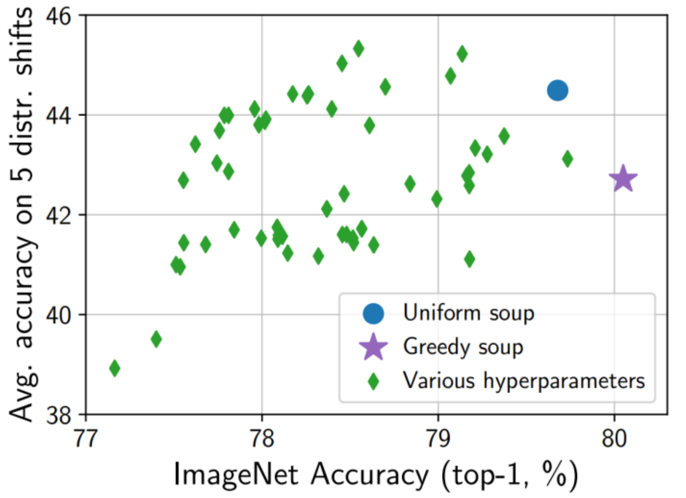


Figure G.3: Fine-tuning on ImageNet, using a ViT-B/32 (Dosovitskiy et al., 2021) pre-trained on ImageNet-22k (Deng et al., 2009).

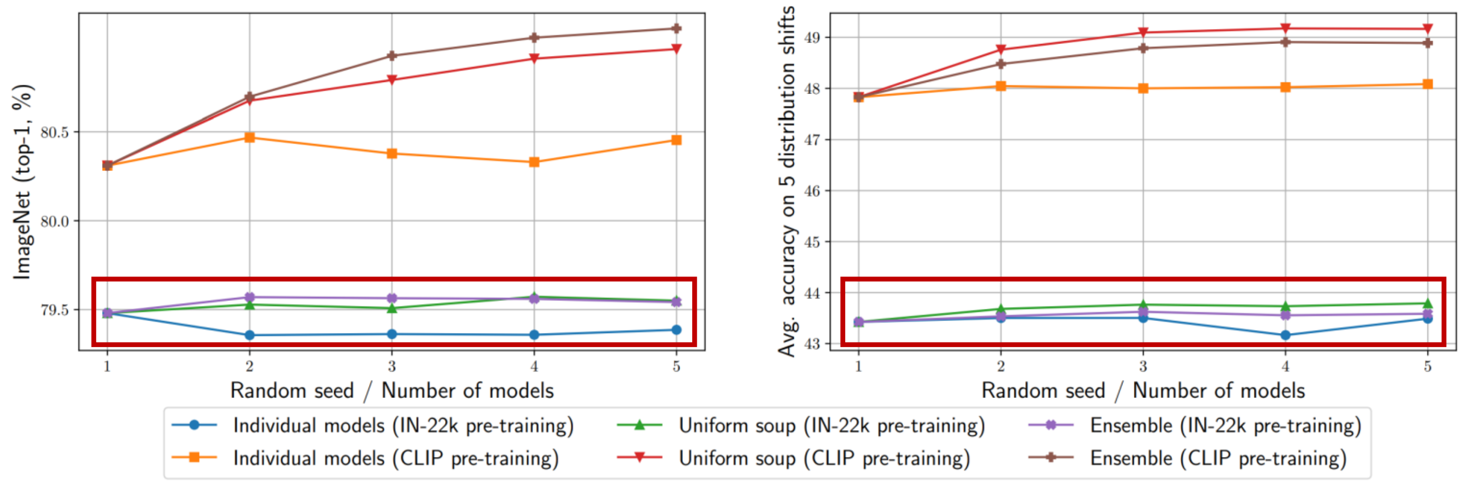


Figure G.4: For a CLIP and ImageNet-22k pre-trained ViT-B/32 model, we use the best hyperparameters found by the hyperparameter sweep to fine-tune multiple times, varying only the random seed. For an experimental budget of $1 \leq k \leq 5$ models, we show: (i) the individual model with random seed k, (ii) the model soup composed of models with random seeds 1 through k, and (iii) the ensemble composed of models with random seeds 1 through k.

Experiments: Fine-tune BERT and T5 on GLUE Benchmark

- Image classification만큼의 결과를 보여주지는 못하였지만, Greedy soup은 BERT와 T5에 대해서도 best individual model보다 높은 성능을 보여주었다.

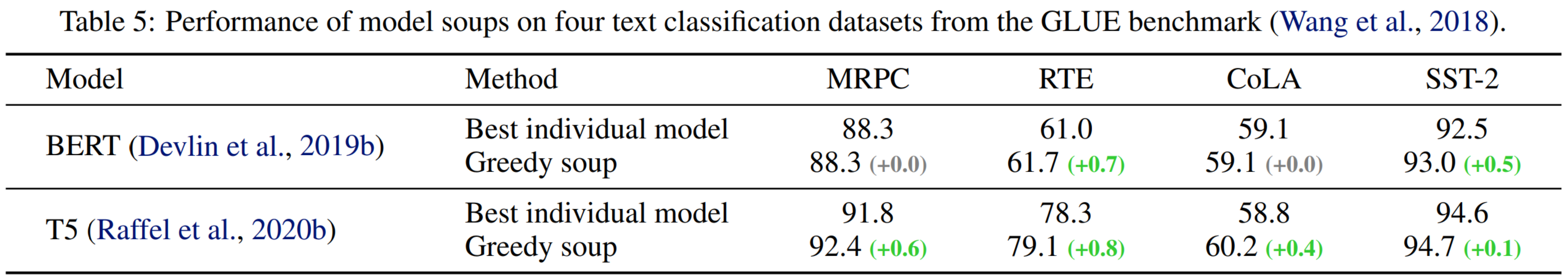


Table 5: Performance of model soups on four text classification datasets from the GLUE benchmark (Wang et al., 2018).

Model	Method	MRPC	RTE	CoLA	SST-2
BERT (Devlin et al., 2019b)	Best individual model	88.3	61.0	59.1	92.5
	Greedy soup	88.3 (+0.0)	61.7 (+0.7)	59.1 (+0.0)	93.0 (+0.5)
T5 (Raffel et al., 2020b)	Best individual model	91.8	78.3	58.8	94.6
	Greedy soup	92.4 (+0.6)	79.1 (+0.8)	60.2 (+0.4)	94.7 (+0.1)

Fine-tune 32 models for each dataset
Varying learning rate, batch size, number of epochs and random seed.

Analytically Comparing Soups to Ensembles

- 두 모델 θ_0, θ_1에 대한 soup을 생각해보자.
  - $\theta_\alpha = (1-\alpha)\cdot\theta_0 + \alpha\cdot\theta_1$
  - $f_\alpha^{\text{ens}}(x) = (1-\alpha)\cdot f(x;\theta_0) + \alpha\cdot f(x;\theta_1)$
  - $\text{err}_\alpha \coloneqq \mathbb{E}_{x,y}\mathbf{1}\left[\arg\max_i f_i(x;\theta_\alpha) \neq y\right]$: θ_α의 에러율
- err_α을 미분가능한 형태로 나타내기 위해 cross-entropy loss와 β-calibration을 이용함.
  - $\mathcal{L}_\alpha^{\text{soup}} = \mathbb{E}_{x,y}[\ell(\beta f(x;\theta_\alpha),y)]$, $\mathcal{L}_\alpha^{\text{ens}} = \mathbb{E}_{x,y}[\ell(\beta f_\alpha^{\text{ens}}(x),y)]$

- **Logit-level ensembles**에 대한 많은 연구를 통해 "$\text{err}_\alpha^{\text{ens}}$ is often strictly below $\min\{\text{err}_0, \text{err}_1\}$"로 알려져 있다.

- 따라서 우리는 $\text{err}_\alpha \approx \text{err}_\alpha^{\text{ens}}$ 임을 보여서 Model soups 또한 ensemble처럼 성능 향상을 얻어낼 수 있다는 것을 증명할 것이다.

Analytically Comparing Soups to Ensembles

- 본 연구에서는 ensemble과 Model soups간의 loss차이가 다음과 같이 근사 된다는 것을 보였다:

$$\mathcal{L}_\alpha^{\text{soup}} - \mathcal{L}_\alpha^{\text{ens}} \approx \frac{\alpha(1-\alpha)}{2}\left(\underbrace{-\frac{\mathrm{d}^2}{\mathrm{d}\alpha^2}\mathcal{L}_\alpha^{\text{soup}}}_{\text{Convexity of the soup loss}} + \beta^2 \mathbb{E}_x\left[\mathrm{Var}_{\underbrace{Y\sim p_{\text{sftmx}}(\beta f(x;\theta_\alpha))}_{\text{Soup produces confident predictions}}}\overbrace{\left[f(x;\theta_1) - f(x;\theta_0)\right]}^{\text{Endpoints are similar}}\right]\right)$$

- Assumption) the logits are not too far from linear.
- See Appendix K.3 for a detailed derivation.

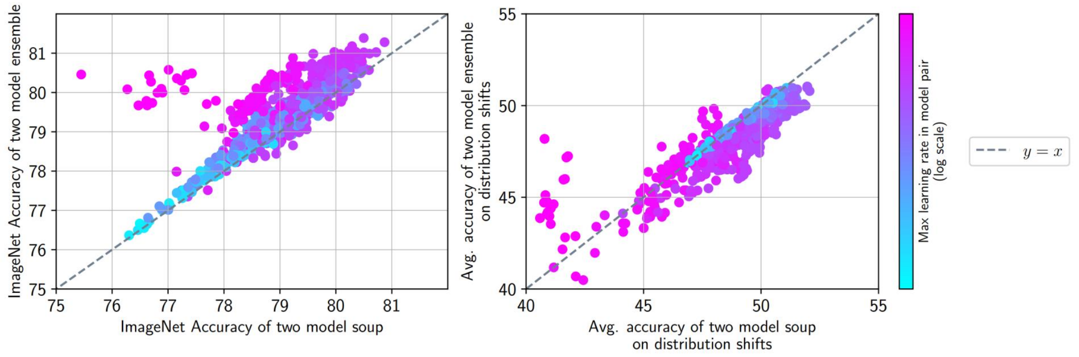


Figure 4: Ensemble performance is correlated with model soup performance. Each point on the scatter plot is a model pair with different hyperparameters. The x-axis is the accuracy when the weights of the two models are averaged (i.e., the two model soup) while the y-axis is the accuracy of the two model ensemble. Ensembles often perform slightly better than soups on ImageNet (left) while the reverse is true on the distribution shifts (right). Each model pair consists of two random greed diamonds from Figure 1.

Empirically comparing

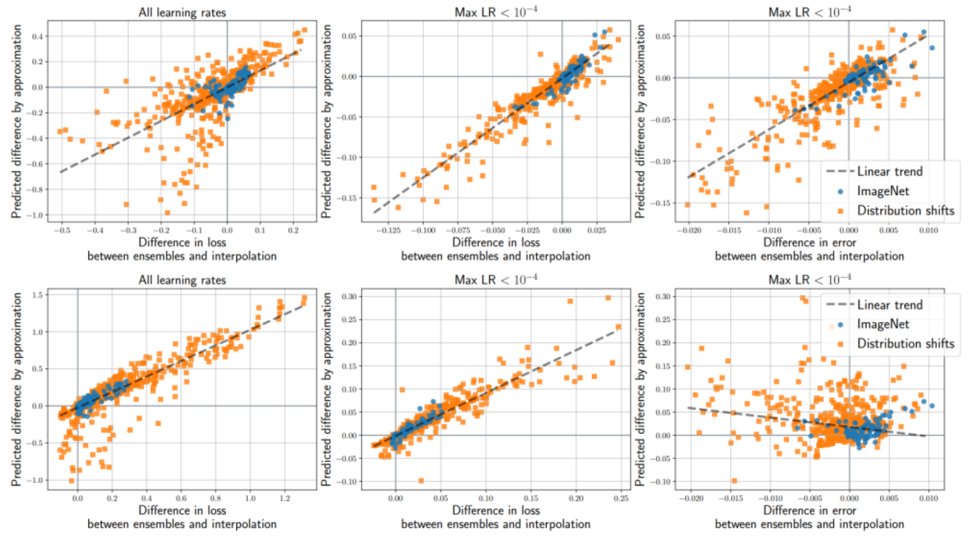


Figure K.1: Validation of the analytical approximation (1) for the performance difference of a 2-model soup and ensemble. Each marker on the scatter plots represent a different choice of endpoint models (θ_0, θ_1) and interpolation weight α. In every scatter plot, the vertical axis shows the true performance difference between the soup and ensemble (in loss for the **left** and **center** panes, and error for the **right** pane), where a positive value indicates the ensemble is better. The horizontal axis shows our approximation for the loss difference. The **top** row shows results with inverse temperature β chosen to calibrate the soup, and the **bottom** row shows results for β fixed to 1.

Validation of the analytical approximation

Conclusion

- Weight를 average하는 것은 성능 향상의 기법으로 자주 사용되었다.
- 동일한 optimization trajectory에서의 weight ensemble과 pre-training and fine-tuning에서의 weight ensemble은 비슷하게 작동한다.
- WiSE-FT를 이용하여 CLIP의 robustness를 유지할 수 있다.
- Greedy soup은 best individual model보다 높은 성능을 기록하였다.
  - with no extra training and no extra compute during inference.
- 단순한 Uniform soup도 각각의 모델이 높은 성능을 기록할 때, best individual model보다 높은 성능을 기록할 수 있었다.
  - 본 실험의 경우 high learning rate을 제외한 uniform soup의 결과를 통해 관측할 수 있었다.
- 하지만, Model soups의 성능 향상은 large, heterogeneous pre-trained model에서 주로 관측되었고, ImageNet-22k pre-training에서만 하더라도 marginal한 성능 향상을 보여주었다.

Guo, Chuan, et al. "On calibration of modern neural networks." *ICML*. 2017.