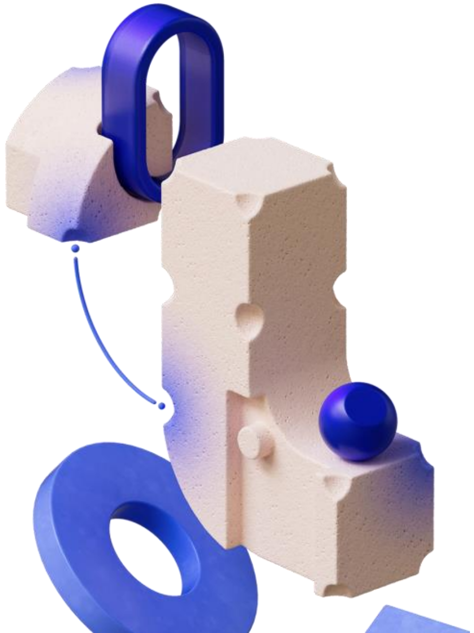
Google DeepMind

Frozen Feature Augmentation for Few-Shot Image Classification



Andreas Bär [+]

TU Braunschweig
Germany

Neil Houlsby

Google DeepMind
Switzerland

Mostafa Dehghani

Google DeepMind
USA

Manoj Kumar [§]

Google DeepMind
Netherlands

[§] Project lead.
[+] Work done during summer internship at Google DeepMind.

Introduction

From Pretraining to Transfer Learning With Frozen Features

Vision models are usually pretrained on **large-datasets**, e.g., ImageNet-21k [Deng+, 2009] or JFT [Zhai+, 2022], and then adapted.

[Deng et al., „ImageNet: A Large-Scale Hierarchical Image Database“, CVPR, June 2009]
[Zhai et al., „Scaling Vision Transformers“, CVPR, June 2022]

Linear probing [Radford+, 2021] is an **effective** method to **transfer** vision models to other tasks [Dehghani+, 2023] using frozen features.

[Radford et al., „Learning Transferable Visual Models From Natural Language Supervision“, ICML, July 2021]
[Dehghani et al., „Scaling Vision Transformers to 22 Billion Parameters“, ICML, July 2023]

Introduction

From Pretraining to Transfer Learning With Frozen Features

Vision models are usually pretrained on **large-datasets**, e.g., ImageNet-21k [Deng+, 2009] or JFT [Zhai+, 2022], and then adapted.

[Deng et al., „ImageNet: A Large-Scale Hierarchical Image Database“, CVPR, June 2009]

[Zhai et al., „Scaling Vision Transformers“, CVPR, June 2022]

Linear probing [Radford+, 2021] is an **effective** method to **transfer** vision models to other tasks [Dehghani+, 2023] using frozen features.

[Radford et al., „Learning Transferable Visual Models From Natural Language Supervision“, ICML, July 2021]

[Dehghani et al., „Scaling Vision Transformers to 22 Billion Parameters“, ICML, July 2023]

Known methods [Radford+, 2021; Dehghani+, 2023; Zhai+, 2023] **do not employ data augmentation** when training on frozen features.

[Zhai et al., „Sigmoid Loss for Language-Image Pretraining“, ICCV, October 2023]

At the same time, **data augmentation is known to be effective** [Cubuk+, 2019; Hendrycks+, 2020; Cubuk+, 2020; Müller & Hutter, 2021].

[Cubuk et al., „AutoAugment: Learning Augmentation Strategies From Data“, CVPR, June 2019]

[Hendrycks et al., „AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty“, ICLR, April 2020]

[Cubuk et al., „RandAugment: Practical Automated Data Augmentation With a Reduced Search Space“, NeurIPS, December 2020]

[Müller and Hutter, „TrivialAugment: Tuning-Free Yet State-of-the-Art Data Augmentation“, ICCV, October 2021]

Introduction

From Pretraining to Transfer Learning With Frozen Features

Vision models are usually pretrained on **large-datasets**, e.g., ImageNet-21k [Deng+, 2009] or JFT [Zhai+, 2022], and then adapted.

[Deng et al., „ImageNet: A Large-Scale Hierarchical Image Database“, CVPR, June 2009]
[Zhai et al., „Scaling Vision Transformers“, CVPR, June 2022]

Linear probing [Radford+, 2021] is an **effective** method to **transfer** vision models to other tasks [Dehghani+, 2023] using frozen features.

[Radford et al., „Learning Transferable Visual Models From Natural Language Supervision“, ICML, July 2021]
[Dehghani et al., „Scaling Vision Transformers to 22 Billion Parameters“, ICML, July 2023]

Known methods [Radford+, 2021; Dehghani+, 2023; Zhai+, 2023] **do not employ data augmentation** when training on frozen features.

[Zhai et al., „Sigmoid Loss for Language-Image Pretraining“, ICCV, October 2023]

At the same time, **data augmentation is known to be effective** [Cubuk+, 2019; Hendrycks+, 2020; Cubuk+, 2020; Müller & Hutter, 2021].

[Cubuk et al., „AutoAugment: Learning Augmentation Strategies From Data“, CVPR, June 2019]
[Hendrycks et al., „AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty“, ICLR, April 2020]
[Cubuk et al., „RandAugment: Practical Automated Data Augmentation With a Reduced Search Space“, NeurIPS, December 2020]
[Müller and Hutter, „TrivialAugment: Tuning-Free Yet State-of-the-Art Data Augmentation“, ICCV, October 2021]

In a **data-constrained**, **few-shot setup** we investigate:

Research question:
Can we effectively combine image augmentations and frozen features to train a lightweight model?

Method Description

Training on Top of Frozen Features in Three Steps

H : image height
W : image width
C : # color channels
N: # image patches
D: hidden size
S: # classes
L: # Transformer blocks

Step 1: Select a (frozen) pretrained model and a layer/block to train on top of frozen features.

Step 2: Process images and cache the (frozen) features.

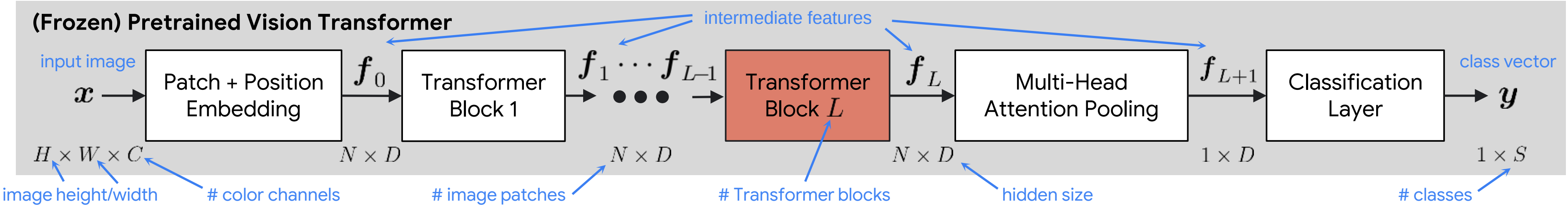
Step 3: Train a (lightweight) model on (augmented) frozen features.

Method Description

Training on Top of Frozen Features in Three Steps

H : image height
W : image width
C : # color channels
N : # image patches
D : hidden size
S : # classes
L : # Transformer blocks

Step 1: Select a (frozen) pretrained model and a layer/block to train on top of frozen features.



Step 2: Process images and cache the (frozen) features.

Step 3: Train a (lightweight) model on (augmented) frozen features.

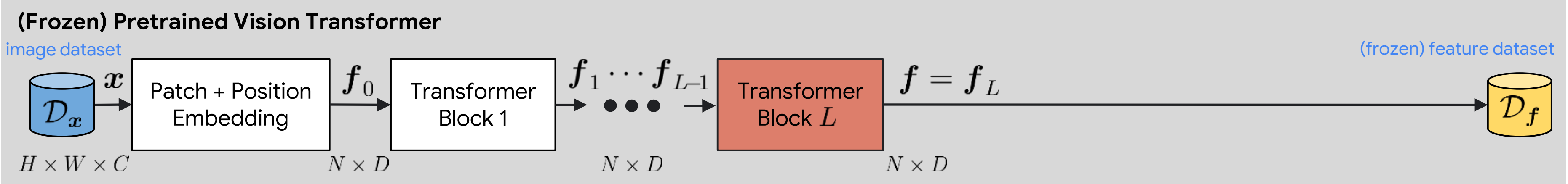
Method Description

Training on Top of Frozen Features in Three Steps

H : image height
W : image width
C : # color channels
N : # image patches
D : hidden size
S : # classes
L : # Transformer blocks

Step 1: Select a (frozen) pretrained model and a layer/block to train on top of frozen features.

Step 2: Process images and cache the (frozen) features.



Step 3: Train a (lightweight) model on (augmented) frozen features.

Method Description

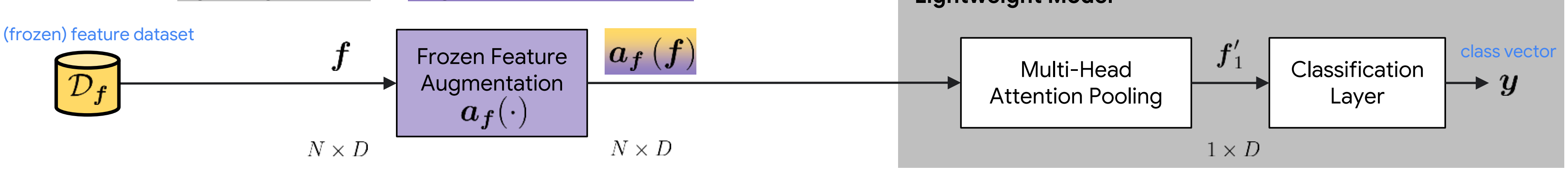
Training on Top of Frozen Features in Three Steps

- H : image height
- W : image width
- C : # color channels
- N : # image patches
- D : hidden size
- S : # classes
- L : # Transformer blocks

Step 1: Select a (frozen) pretrained model and a layer/block to train on top of frozen features.

Step 2: Process images and cache the (frozen) features.

Step 3: Train a (lightweight) model on (augmented) frozen features.



Method Description

From Image Augmentations to Frozen Feature Augmentations (FroFAs)

Step 0: Feature reshaping from 2D ($\boldsymbol{f}$) to 3D ($\boldsymbol{f}^*$). - *optional* -

Reminder: $\boldsymbol{x} \in \mathbb{I}^{H \times W \times C}$ # color channels ... *with* $\mathbb{I} = [0, 1]$ *or* $\mathbb{I} = \{0, 1, 2, ..., 255\}$

image height/width

image value range

image patches

hidden size

$\boldsymbol{f} = (\boldsymbol{f}_{n,d}) \in \mathbb{R}^{N \times D}$ ... *reshaped to* ... $\boldsymbol{f}^* = (\boldsymbol{f}^*_{n_1,n_2,d}) \in \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times D}$ ... *with* ... $\boldsymbol{f}^*_d = \boldsymbol{f}^*_{:,:,d} \in \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times 1}$

patch index

channel index

2D patch indices

2D-ordered patches of a channel

Method Description

From Image Augmentations to Frozen Feature Augmentations (FroFAs)

Step 0: Feature reshaping from 2D ($\boldsymbol{f}$) to 3D ($\boldsymbol{f}^*$). *- optional -*

Reminder: $\boldsymbol{x} \in \mathbb{I}^{H \times W \times C}$ # color channels ... *with* $\mathbb{I} = [0, 1]$ *or* $\mathbb{I} = \{0, 1, 2, ..., 255\}$

image height/width image value range

image patches hidden size

$\boldsymbol{f} = (f_{n,d}) \in \mathbb{R}^{N \times D}$ *... reshaped to ...* $\boldsymbol{f}^* = (f^*_{n_1,n_2,d}) \in \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times D}$ *... with ...* $\boldsymbol{f}^*_d = f^*_{:,:,d} \in \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times 1}$

patch index channel index 2D patch indices 2D-ordered patches of a channel

Step 1: Address the channel dimensionality mismatch between $\boldsymbol{x}$ and $\boldsymbol{f}$ or $\boldsymbol{f}^*$.

Method Description

From Image Augmentations to Frozen Feature Augmentations (FroFAs)

Step 0: Feature reshaping from 2D ($\boldsymbol{f}$) to 3D ($\boldsymbol{f}^*$). *- optional -*

Reminder: $\boldsymbol{x} \in \mathbb{I}^{H \times W \times C}$ # color channels ... *with* $\mathbb{I} = [0, 1]$ *or* $\mathbb{I} = \{0, 1, 2, ..., 255\}$

image height/width

image value range

image patches

hidden size

$\boldsymbol{f} = (f_{n,d}) \in \mathbb{R}^{N \times D}$ *... reshaped to ...* $\boldsymbol{f}^* = (f^*_{n_1,n_2,d}) \in \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times D}$ *... with ...* $\boldsymbol{f}^*_d = f^*_{:,:,d} \in \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times 1}$

patch index

channel index

2D patch indices

2D-ordered patches of a channel

Step 1: Address the channel dimensionality mismatch between $\boldsymbol{x}$ and $\boldsymbol{f}$ or $\boldsymbol{f}^*$.

- **Most image augmentations** can be applied channel-wise, e.g., brightness adjustments.
- We **re-use these augmentations** in the feature space and ignore image augmentations relying on three color channels, e.g., color jitter.

Method Description

From Image Augmentations to Frozen Feature Augmentations (FroFAs)

Step 0: Feature reshaping from 2D ($\boldsymbol{f}$) to 3D ($\boldsymbol{f}^*$). *- optional -*

Reminder: $\boldsymbol{x} \in \mathbb{I}^{H \times W \times C}$ # color channels ... *with* $\mathbb{I} = [0, 1]$ *or* $\mathbb{I} = \{0, 1, 2, ..., 255\}$

image height/width

image value range

image patches

hidden size

$\boldsymbol{f} = (f_{n,d}) \in \mathbb{R}^{N \times D}$ *... reshaped to ...* $\boldsymbol{f}^* = (f^*_{n_1,n_2,d}) \in \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times D}$ *... with ...* $\boldsymbol{f}^*_d = f^*_{:,:,d} \in \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times 1}$

patch index

channel index

2D patch indices

2D-ordered patches of a channel

Step 1: Address the channel dimensionality mismatch between $\boldsymbol{x}$ and $\boldsymbol{f}$ or $\boldsymbol{f}^*$.

- **Most image augmentations** can be applied channel-wise, e.g., brightness adjustments.
- We **re-use these augmentations** in the feature space and ignore image augmentations relying on three color channels, e.g., color jitter.

Step 2: Address the value range mismatch between $\boldsymbol{x}$ and $\boldsymbol{f}$ or $\boldsymbol{f}^*$.

Method Description

From Image Augmentations to Frozen Feature Augmentations (FroFAs)

Step 0: Feature reshaping from 2D ($\boldsymbol{f}$) to 3D ($\boldsymbol{f}^*$). *- optional -*

Reminder: $\boldsymbol{x} \in \mathbb{I}^{H \times W \times C}$ ← # color channels ... *with* $\mathbb{I} = [0, 1]$ *or* $\mathbb{I} = \{0, 1, 2, ..., 255\}$

image height/width

image value range

image patches

hidden size

$\boldsymbol{f} = (f_{n,d}) \in \mathbb{R}^{N \times D}$ *... reshaped to ...* $\boldsymbol{f}^* = (f^*_{n_1, n_2, d}) \in \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times D}$ *... with ...* $\boldsymbol{f}^*_d = \boldsymbol{f}^*_{:,:,d} \in \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times 1}$

patch index

channel index

2D patch indices

2D-ordered patches of a channel

Step 1: Address the channel dimensionality mismatch between $\boldsymbol{x}$ and $\boldsymbol{f}$ or $\boldsymbol{f}^*$.

- **Most image augmentations** can be applied channel-wise, e.g., brightness adjustments.
- We **re-use these augmentations** in the feature space and ignore image augmentations relying on three color channels, e.g., color jitter.

Step 2: Address the value range mismatch between $\boldsymbol{x}$ and $\boldsymbol{f}$ or $\boldsymbol{f}^*$.

- We define feature-to-image/image-to feature transformations applied simultaneously on D_t channels.

$\boldsymbol{t}_{\boldsymbol{f} \to \boldsymbol{x}} : \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times D_t} \to \mathbb{I}^{\sqrt{N} \times \sqrt{N} \times D_t}$ $\quad$ $\boldsymbol{t}_{\boldsymbol{f} \leftarrow \boldsymbol{x}} : \mathbb{I}^{\sqrt{N} \times \sqrt{N} \times D_t} \to \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times D_t}$

We perform a linear scaling:

$$\boldsymbol{x}_{\boldsymbol{f}} = \boldsymbol{t}_{\boldsymbol{f} \to \boldsymbol{x}}(\boldsymbol{f}^*) = \frac{\boldsymbol{f}^* - f_{\min}}{f_{\max} - f_{\min}}$$

Method Description

From Image Augmentations to Frozen Feature Augmentations (FroFAs)

Step 0: Feature reshaping from 2D (f) to 3D (f^*). *- optional -*

Reminder: $x \in \mathbb{I}^{H \times W \times C}$ # color channels ... *with* $\mathbb{I} = [0, 1]$ *or* $\mathbb{I} = \{0, 1, 2, ..., 255\}$

image height/width image value range

image patches hidden size

$f = (f_{n,d}) \in \mathbb{R}^{N \times D}$ ... *reshaped to* ... $f^* = (f^*_{n_1,n_2,d}) \in \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times D}$ ... *with* ... $f^*_d = f^*_{:,:,d} \in \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times 1}$

patch index channel index 2D patch indices 2D-ordered patches of a channel

Step 1: Address the channel dimensionality mismatch between x and f or f^*.

- **Most image augmentations** can be applied channel-wise, e.g., brightness adjustments.
- We **re-use these augmentations** in the feature space and ignore image augmentations relying on three color channels, e.g., color jitter.

Step 2: Address the value range mismatch between x and f or f^*.

- We define feature-to-image/image-to feature transformations applied simultaneously on D_t channels.

$t_{f \to x} : \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times D_t} \to \mathbb{I}^{\sqrt{N} \times \sqrt{N} \times D_t}$ $\quad t_{f \leftarrow x} : \mathbb{I}^{\sqrt{N} \times \sqrt{N} \times D_t} \to \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times D_t}$

We perform a linear scaling:

$$x_f = t_{f \to x}(f^*) = \frac{f^* - f_{\min}}{f_{\max} - f_{\min}}$$

Step 3: Putting all together → **FroFA (a_f)**:

$a_f = t_{f \leftarrow x} \circ a_x \circ t_{f \to x}$ ... *with* $a_x : \mathbb{I}^{\sqrt{N} \times \sqrt{N} \times D_a} \to \mathbb{I}^{\sqrt{N} \times \sqrt{N} \times D_a}$

image augmentation

Method Description

From Image Augmentations to Frozen Feature Augmentations (FroFAs)

Step 0: Feature reshaping from 2D (f) to 3D (f^*). *- optional -*

Reminder: $x \in \mathbb{I}^{H \times W \times C}$ ← # color channels ... *with* $\mathbb{I} = [0, 1]$ *or* $\mathbb{I} = \{0, 1, 2, ..., 255\}$

image height/width

image value range

image patches

hidden size

$f = (f_{n,d}) \in \mathbb{R}^{N \times D}$ *... reshaped to ...* $f^* = (f^*_{n_1,n_2,d}) \in \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times D}$ *... with ...* $f^*_d = f^*_{:,:,d} \in \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times 1}$

patch index

channel index

2D patch indices

2D-ordered patches of a channel

Step 1: Address the channel dimensionality mismatch between x and f or f^*.

- **Most image augmentations** can be applied channel-wise, e.g., brightness adjustments.
- We **re-use these augmentations** in the feature space and ignore image augmentations relying on three color channels, e.g., color jitter.

Step 2: Address the value range mismatch between x and f or f^*.

- We define feature-to-image/image-to feature transformations applied simultaneously on D_t channels.

$t_{f \to x} : \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times D_t} \to \mathbb{I}^{\sqrt{N} \times \sqrt{N} \times D_t}$ $\quad t_{f \leftarrow x} : \mathbb{I}^{\sqrt{N} \times \sqrt{N} \times D_t} \to \mathbb{R}^{\sqrt{N} \times \sqrt{N} \times D_t}$

We perform a linear scaling:

$$x_f = t_{f \to x}(f^*) = \frac{f^* - f_{\min}}{f_{\max} - f_{\min}}$$

Step 3: Putting all together → **FroFA** (a_f):

$a_f = t_{f \leftarrow x} \circ a_x \circ t_{f \to x}$ *... with* $\underset{\text{image augmentation}}{a_x} : \mathbb{I}^{\sqrt{N} \times \sqrt{N} \times D_a} \to \mathbb{I}^{\sqrt{N} \times \sqrt{N} \times D_a}$

By default, we have the same mapping and randomness across channels.

We tested **channel-wise interpretations**:

- Channel-wise randomness
- Channel-wise mapping/randomness

Experimental Setup

Network Architectures, Baseline Models, Pretraining/Few-Shot Transfer Datasets, Data Augmentation

Network architectures:

ViT Model	# Params (M)
Ti/16	5.5
B/16	86.0
L/16	303.0

Pure **image classification** pretraining:

Dataset	# Images
JFT-3B	~ 3 billion
ImageNet-21k	~ 14 million

Vision-language pretraining:

Dataset	# Images-text pairs
WebLI	~ 31 billion

Few-shot learning (1-, 5-, 10-, 25-shot):

Dataset	# Classes
ImageNet1k	1000
CIFAR10	10
CIFAR100	100
DMLab	6
DTD	47
Resisc45	45
SUN397	397
SVHN	10

Experimental Setup

Network Architectures, Baseline Models, Pretraining/Few-Shot Transfer Datasets, Data Augmentation

Network architectures:

ViT Model	# Params (M)
Ti/16	5.5
B/16	86.0
L/16	303.0

Pure **image classification** pretraining:

Dataset	# Images
JFT-3B	~ 3 billion
ImageNet-21k	~ 14 million

Vision-language pretraining:

Dataset	# Images-text pairs
WebLI	~ 31 billion

Few-shot learning (1-, 5-, 10-, 25-shot):

Dataset	# Classes
ImageNet1k	1000
CIFAR10	10
CIFAR100	100
DMLab	6
DTD	47
Resisc45	45
SUN397	397
SVHN	10

Baseline models:

Baseline	Input size	Regularization
MAP^{wd}	$N \times D$	weight decay
Linear probe	$1 \times D$	L2 regularization

our lightweight model, but without FroFA

penultimate output of the pretrained model

Data augmentation

Other
- JPEG
- mixup

Crop & Drop
- crop
- resized crop
- inception crop
- patch dropout

Stylistic
- brightness
- contrast
- equalize
- invert
- posterize
- sharpness
- solarize

Geometric
- rotate
- shear-x,
- shear-y,
- translate-x,
- translate-y

Experimental Setup

Network Architectures, Baseline Models, Pretraining/Few-Shot Transfer Datasets, Data Augmentation

Network architectures:

ViT Model	# Params (M)
Ti/16	5.5
B/16	86.0
L/16	303.0

Pure **image classification** pretraining:

Dataset	# Images
JFT-3B	~ 3 billion
ImageNet-21k	~ 14 million

Vision-language pretraining:

Dataset	# Images-text pairs
WebLI	~ 31 billion

Few-shot learning (1-, 5-, 10-, 25-shot):

Dataset	# Classes
ImageNet1k	1000
CIFAR10	10
CIFAR100	100
DMLab	6
DTD	47
Resisc45	45
SUN397	397
SVHN	10

Baseline models:

Baseline	Input size	Regularization
MAP^{wd}	$N \times D$	weight decay
Linear probe	$1 \times D$	L2 regularization

our lightweight model, but without FroFA

penultimate output of the pretrained model

Data augmentation

Stylistic
- brightness
- contrast
- equalize
- invert
- posterize
- sharpness
- solarize

Other
- JPEG
- mixup

Crop & Drop
- crop
- resized crop
- inception crop
- patch dropout

Geometric
- rotate
- shear-x,
- shear-y,
- translate-x,
- translate-y

TL;DR

We test **three** ViT models, **three** pretraining datasets, **eight** few-shot datasets, **eighteen** data augmentations ...
... and compare to **two** baselines.

Experimental Setup

Network Architectures, Baseline Models, Pretraining/Few-Shot Transfer Datasets, Data Augmentation

Network architectures:

ViT Model	# Params (M)
Ti/16	5.5
B/16	86.0
L/16	303.0

Pure **image classification** pretraining:

Dataset	# Images
JFT-3B	~ 3 billion
ImageNet-21k	~ 14 million

Vision-language pretraining:

Dataset	# Images-text pairs
WebLI	~ 31 billion

Few-shot learning (1-, 5-, 10-, 25-shot):

Dataset	# Classes
ImageNet1k	1000
CIFAR10	10
CIFAR100	100
DMLab	6
DTD	47
Resisc45	45
SUN397	397
SVHN	10

Baseline models:

Baseline	Input size	Regularization
MAP[wd]	$N \times D$	weight decay
Linear probe	$1 \times D$	L2 regularization

our lightweight model, but without FroFA

penultimate output of the pretrained model

Data augmentation

Stylistic
- brightness
- contrast
- equalize
- invert
- posterize
- sharpness
- solarize

Other
- JPEG
- mixup

Crop & Drop
- crop
- resized crop
- inception crop
- patch dropout

Geometric
- rotate
- shear-x,
- shear-y,
- translate-x,
- translate-y

TL;DR

We test **three** ViT models, **three** pretraining datasets, **eight** few-shot datasets, **eighteen** data augmentations ...

... and compare to **two** baselines.

In the following, we focus on a subset of our results.

Experimental Results (Highlights)

L/16 Pretrained on JFT-3B or WebLI and Transferred to Seven Few-Shot Datasets Using a Channel Variant of Brightness FroFA

We compute the average gains across **7 few-shot datasets (SUN397, ...)**, excluding ImageNet1k.

top-1 accuracy [%]

Pretraining scheme	Method	Shots 1	5	10	25
JFT-3B	**MAP^{wd}**				
	Linear probe				
	MAP^{wd} + FroFA (**ours**)				

top-1 accuracy [%]

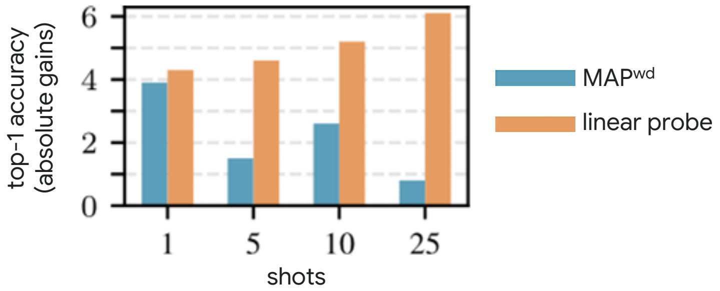
Pretraining scheme	Method	Shots 1	5	10	25
WebLI + SigLIP	**MAP^{wd}**				
	Linear probe				
	MAP^{wd} + FroFA (**ours**)				

Experimental Results (Highlights)

L/16 Pretrained on JFT-3B or WebLI and Transferred to Seven Few-Shot Datasets Using a Channel Variant of Brightness FroFA

We compute the average gains across **7 few-shot datasets (SUN397, …)**, excluding ImageNet1k.

top-1 accuracy [%]		Shots			
Pretraining scheme	**Method**	1	5	10	25
JFT-3B	**MAP^{wd}**	49.5	65.8	68.3	74.1
	Linear probe	49.1	62.7	65.7	68.8
	MAP^{wd} + FroFA (**ours**)	**53.4**	**67.3**	**70.9**	**74.9**



top-1 accuracy [%]		Shots			
Pretraining scheme	**Method**	1	5	10	25
WebLI + SigLIP	**MAP^{wd}**				
	Linear probe				
	MAP^{wd} + FroFA (**ours**)				

Experimental Results (Highlights)

L/16 Pretrained on JFT-3B or WebLI and Transferred to Seven Few-Shot Datasets Using a Channel Variant of Brightness FroFA

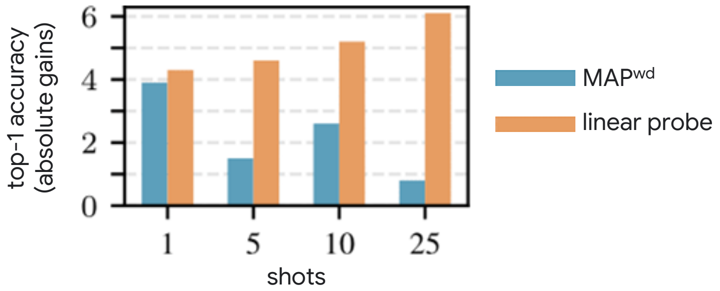
We compute the average gains across **7 few-shot datasets (SUN397, …)**, excluding ImageNet1k.

top-1 accuracy [%]

Pretraining scheme	**Method**	**Shots** 1	5	10	25
JFT-3B	**MAP^{wd}**	49.5	65.8	68.3	74.1
	Linear probe	49.1	62.7	65.7	68.8
	MAP^{wd} + FroFA (**ours**)	**53.4**	**67.3**	**70.9**	**74.9**

top-1 accuracy [%]

Pretraining scheme	**Method**	**Shots** 1	5	10	25
WebLI + SigLIP	**MAP^{wd}**				
	Linear probe				
	MAP^{wd} + FroFA (**ours**)				



JFT-3B pretraining:

- On average, we **improve** upon both baselines (MAP^{wd}, linear probe).
- The **gains** for MAP^{wd} **diminish** with **higher shots**, but for linear probe **increase** with **higher shots**.

Experimental Results (Highlights)

L/16 Pretrained on JFT-3B or WebLI and Transferred to Seven Few-Shot Datasets Using a Channel Variant of Brightness FroFA

We compute the average gains across **7 few-shot datasets (SUN397, ...)**, excluding ImageNet1k.

top-1 accuracy [%]

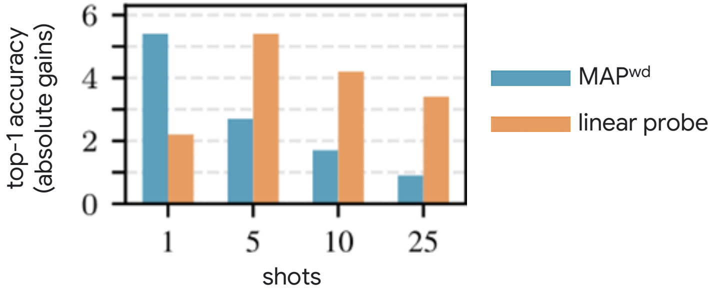
Pretraining scheme	Method	Shots: 1	5	10	25
JFT-3B	MAP^{wd}	49.5	65.8	68.3	74.1
	Linear probe	49.1	62.7	65.7	68.8
	MAP^{wd} + FroFA (**ours**)	**53.4**	**67.3**	**70.9**	**74.9**

JFT-3B pretraining:

- On average, we **improve** upon both baselines (MAP^{wd}, linear probe).
- The **gains** for MAP^{wd} **diminish** with **higher shots**, but for linear probe **increase** with **higher shots**.

top-1 accuracy [%]

Pretraining scheme	Method	Shots: 1	5	10	25
WebLI + SigLIP	MAP^{wd}	45.9	67.7	71.8	75.1
	Linear probe	49.1	65.0	69.3	72.6
	MAP^{wd} + FroFA (**ours**)	**51.3**	**70.4**	**73.5**	**76.0**



Experimental Results (Highlights)

L/16 Pretrained on JFT-3B or WebLI and Transferred to Seven Few-Shot Datasets Using a Channel Variant of Brightness FroFA

We compute the average gains across **7 few-shot datasets (SUN397, ...)**, excluding ImageNet1k.

top-1 accuracy [%]

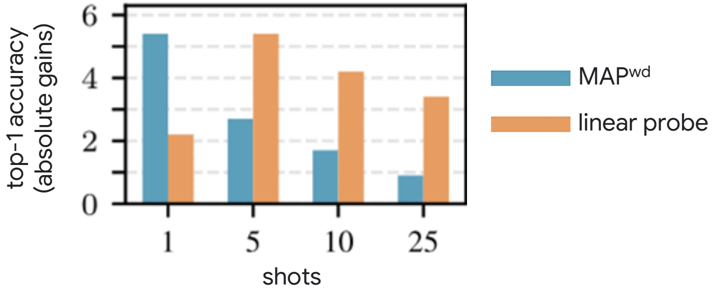
Pretraining scheme	Method	Shots 1	5	10	25
JFT-3B	MAP^{wd}	49.5	65.8	68.3	74.1
	Linear probe	49.1	62.7	65.7	68.8
	MAP^{wd} + FroFA (**ours**)	**53.4**	**67.3**	**70.9**	**74.9**

JFT-3B pretraining:

- On average, we **improve** upon both baselines (MAP^{wd}, linear probe).
- The **gains** for MAP^{wd} **diminish** with **higher shots**, but for linear probe **increase** with **higher shots**.

top-1 accuracy [%]

Pretraining scheme	Method	Shots 1	5	10	25
WebLI + SigLIP	MAP^{wd}	45.9	67.7	71.8	75.1
	Linear probe	49.1	65.0	69.3	72.6
	MAP^{wd} + FroFA (**ours**)	**51.3**	**70.4**	**73.5**	**76.0**



WebLI vision-language pretraining (based on SigLIP [Zhai+, 2023]):

[Zhai et al., „Sigmoid Loss for Language-Image Pretraining“, ICCV, October 2023]

- On average, we **improve** upon both baselines (MAP^{wd}, **linear probe**).
- The **gains** for both MAP^{wd} and **linear probe diminish** with **higher shots**.

Experimental Results (Highlights)

L/16 Pretrained on JFT-3B or WebLI and Transferred to Seven Few-Shot Datasets Using a Channel Variant of Brightness FroFA

We compute the average gains across **7 few-shot datasets (SUN397, ...)**, excluding ImageNet1k.

top-1 accuracy [%]

Pretraining scheme	Method	Shots 1	5	10	25
JFT	MAP^{wd}	[illegible]	[illegible]	[illegible]	[illegible]

top-1 accuracy [%]

Pretraining scheme	Method	Shots 1	5	10	25
	MAP^{wd}	[illegible]	[illegible]	[illegible]	75.1
					72.6
					76.0

Initial Research question:

Can we effectively combine image augmentations and frozen features to train a lightweight model?

JFT-3B pretraining:

- On average, we **improve** upon both baselines (MAP^{wd}, linear probe).
- The **gains** for MAP^{wd} **diminish** with **higher shots**, but for linear probe **increase** with **higher shots**.

WebLI vision-language pretraining (based on SigLIP [Zhai+, 2023]):

[Zhai et al., „Sigmoid Loss for Language-Image Pretraining", ICCV, October 2023]

- On average, we **improve** upon both baselines (MAP^{wd}, linear probe).
- The **gains** for both MAP^{wd} and linear probe **diminish** with **higher shots**.

Experimental Results (Highlights)

L/16 Pretrained on JFT-3B or WebLI and Transferred to Seven Few-Shot Datasets Using a Channel Variant of Brightness FroFA

We compute the average gains across **7 few-shot datasets (SUN397, …)**, excluding ImageNet1k.

top-1 accuracy [%]

Pretraining scheme	Method	Shots 1	5	10	25
	MAP^{wd}				
JFT					

top-1 accuracy [%]

Pretraining scheme	Method	Shots 1	5	10	25
	MAP^{wd}				75.1
					72.6
					76.0

Initial Research question:

Can we effectively combine image augmentations and frozen features to train a lightweight model?

Answer:

Yes! Our method **transfers across pretraining** and **few-shot transfer setups!**

JFT-3B pretraining:

- On average, we **improve** upon both baselines (MAP^{wd}, linear probe).
- The **gains** for MAP^{wd} **diminish** with **higher shots**, but for linear probe **increase** with **higher shots.**

WebLI vision-language pretraining (based on SigLIP [Zhai+, 2023]):

[Zhai et al., „Sigmoid Loss for Language-Image Pretraining", ICCV, October 2023]

- On average, we **improve** upon both baselines (MAP^{wd}, linear probe).
- The **gains** for both MAP^{wd} and linear probe **diminish** with **higher shots.**

Conclusion

We investigated 18 frozen feature augmentations (FroFAs) for few-shot transfer learning for image classification.

We performed ablations along three axes: model size, pretraining, and transfer few-shot dataset.

Our main takeaways:

(a) Shown in this talk:

- Our best FroFA **excels** on **smaller few-shot datasets**.
- Our best FroFA **transfers** to **vision-language pretrained models**.

Conclusion

We investigated 18 frozen feature augmentations (FroFAs) for few-shot transfer learning for image classification.

We performed ablations along three axes: model size, pretraining, and transfer few-shot dataset.

Our main takeaways:

(a) Shown in this talk:

- Our best FroFA **excels** on **smaller few-shot datasets**.
- Our best FroFA **transfers** to **vision-language pretrained models**.

(b) Not shown in this talk:

- Training with **stylistic FroFAs** give **large improvements** upon a representative baseline across all shots.
- **Channel variants** can yield to **further improvements**.
- **Sequential protocols** seem **promising** for future works, given our simple proof of concept.

For more details, checkout our paper or let‘s have a chat at the conference!

Thank you.

Link to our paper:

SCAN ME

Andreas Bär
TU Braunschweig

Neil Houlsby
Google DeepMind

Mostafa Dehghani
Google DeepMind

Manoj Kumar
Google DeepMind