Rethinking Generalization in Few-Shot Classification

Hiller et al. NeurIPS 2022

MLAI In-depth Seminar

Plan for today

- Few-Shot Learning
- Existing Works
- Problem Define
- Method
- Results
- Conclusion & Discussion

Source:

Few-Shot Learning

- Aims at classifying unlabeled samples (**query** set) into unseen classes given very few labeled samples (**support** set)
- Two main challenges
  - Unseen classes : *non-overlap between training and test classes*
  - Low-data problem : *very few labeled samples for the test unseen classes*

Source:

Few-Shot Learning

Preliminary

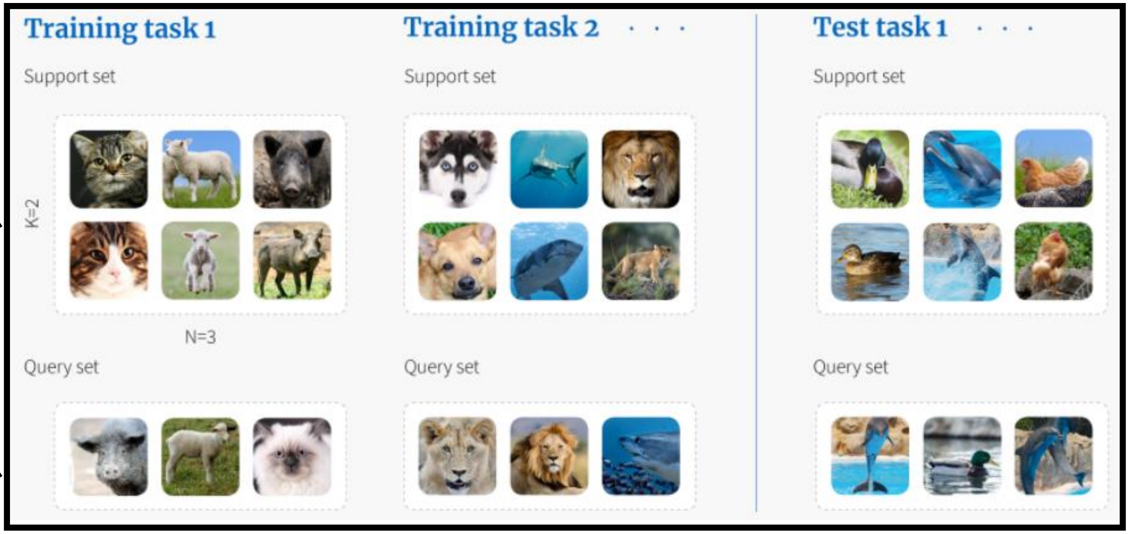
- Aims at classifying unlabeled samples (**query** set) into unseen classes given very few labeled samples (**support** set)
- Two main challenges
  - Unseen classes : *non-overlap between training and test classes*
  - Low-data problem : *very few labeled samples for the test unseen classes*

- Problem Define

< Episodic training and testing >

Generally, Training set contains a large number of classes and correspoding labeled samples.

But, for matching training/test procedures, the dataset is constructed as a series of episodes.



$C_{train} \cap C_{test} = \emptyset$

$C_{train} = \{cat, goat, boar, dog, shark, lion\}$ $C_{test} = \{dug, dolphin, chicken\}$

Source:

Few-Shot Learning

Preliminary

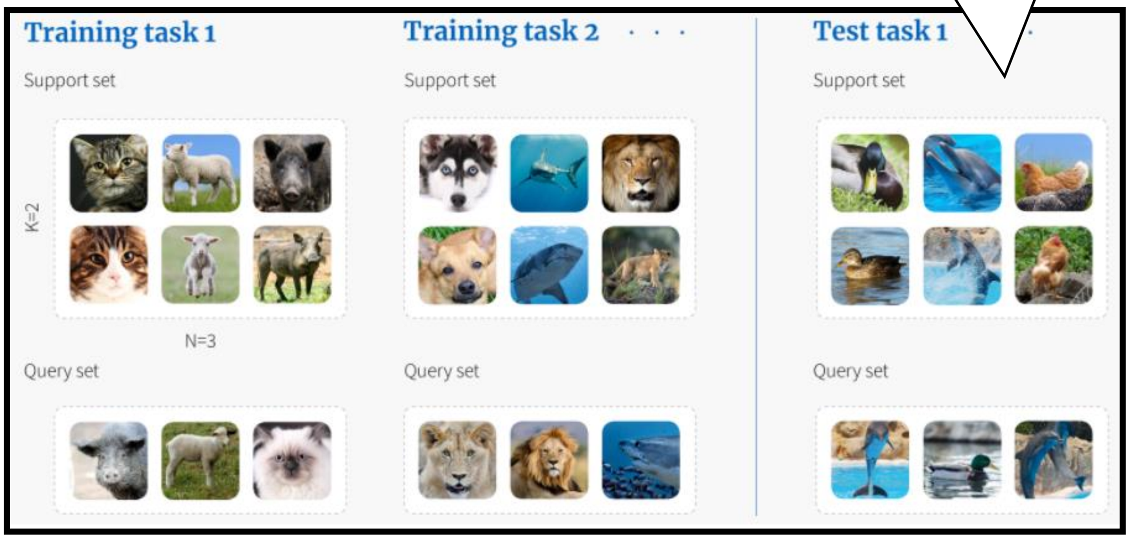
- Aims at classifying unlabeled samples (**query** set) into unseen classes given very few labeled samples (**support** set)

- Two main challenges
  - Unseen classes : *non-overlap between training and test classes*
  - Low-data problem : *very few labeled samples for the test unsee*

Note that, test set also contain *k*-shot labelled data subset (support set).
The performance is measure on the only unlabelled query set.

- Problem Define

< Episodic training and testing >



$C_{train} \cap C_{test} = \emptyset$

$C_{train} = \{cat, goat, boar, dog, shark, lion\}$ $C_{test} = \{dug, dolphin, chicken\}$

Source:

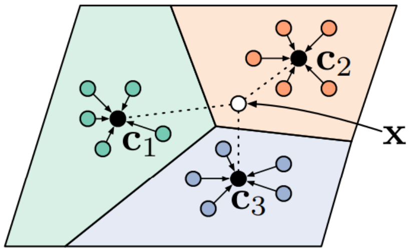
Existing Works

- Metric-based approaches
  - Prototypical Network, Matching Network, Relation Network, …
  - Learn a good feature space where categories can distinguish with each other based on a distance metric, and perform distance-based prediction (nearest neighbor classifier).

$$p_\phi(y = k \mid \mathbf{x}) = \frac{\exp(-d(f_\phi(\mathbf{x}), \mathbf{c}_k))}{\sum_{k'} \exp(-d(f_\phi(\mathbf{x}), \mathbf{c}_{k'}))}$$

$$\mathbf{c}_k = \frac{1}{|S_k|} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_\phi(\mathbf{x}_i)$$

Non-parameteric classifier



- Optimization-based approaches
  - MAML, Reptile, …
  - Learn a good initilaization so that the leaner could rapidly adapt to novel tasks within a few optimization steps.

Source:

Existing Works

- Metric-based approaches
  - Prototypical Network, Matching Network, Relation Network, …
  - Learn a good feature space where categories can distinguish with each other based on a distance metric, and perform distance-based prediction (nearest neighbor classifier).
- Optimization-based approaches
  - MAML, Reptile, …
  - Learn a good initilaization so that the leaner could rapidly adapt to novel tasks within a few optimization steps.

Algorithm 1 Model-Agnostic Meta-Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
randomly initialize θ
while not done **do**
  Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
  for all $\mathcal{T}_i$ **do**
    Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ with respect to K examples
    Compute adapted parameters with gradient descent: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
  end for
  Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$
end while

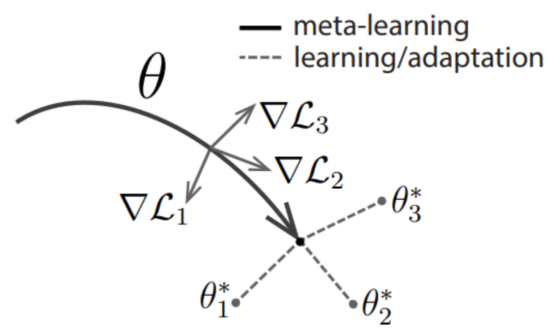


Figure 1. Diagram of our model-agnostic meta-learning algorithm (MAML), which optimizes for a representation θ that can quickly adapt to new tasks.

Source:

Problem Define

Alleviate supervision collapse for few-shot generalization

- Single image-level annotations can not depict the complex real-world scenes. They **only describe a small subset of an image's content**.
  - Neural Network representations lose any information that is not necessary for performing the training task.
    - ✓ However, this information may be necessary for transfer to new tasks or domain.
    - ✓ **Supervision collapse**
  - This might be problematic,
    - ✓ especially when the training and test time classes are differ.
    - ✓ When a test image containing multiple objects

Source:

Problem Define

Alleviate supervision collapse for few-shot generalization

- Supervision collapse

  - The extracted feature may attend to the objects from **seen** classes (person, chair) which have large number of labeled samples in the training set, while ignore the target object from **unseen** class (curtain).

Source: Cross Attention Network for Few-shot Classification, NeurIPS 2019

Problem Define

Alleviate supervision collapse for few-shot generalization

- Supervision collapse

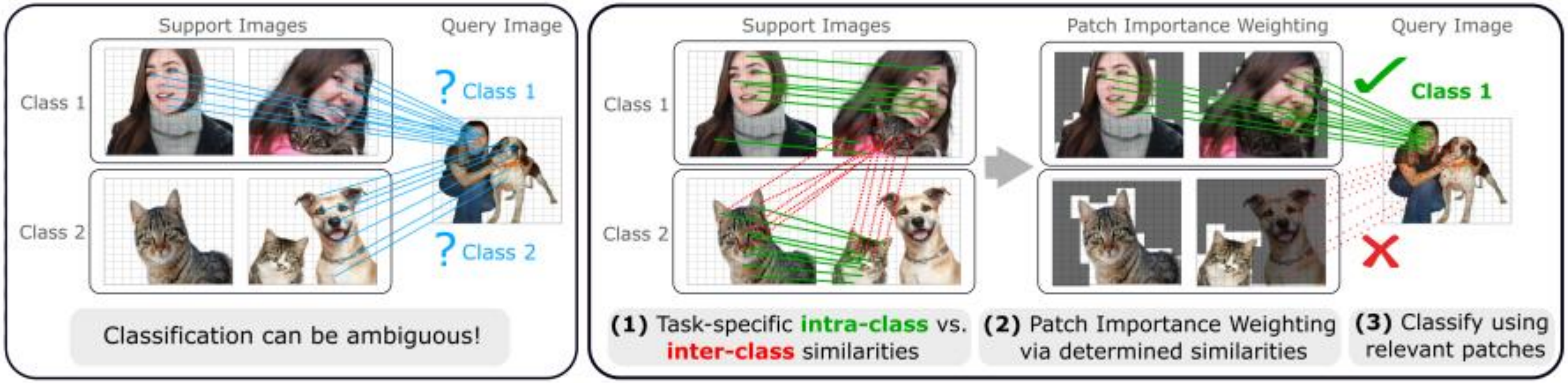
- The errors within that wrong class often have widely different appearance.
- Author's interpretation: the network picks up on image patterns during training that allow images of each class to be tightly grouped in the feature space, minimizing other ways that the image might be similar to other classes in preparation for a confident classification.

Source: CrossTransformers: spatially-aware few-shot transfer, NeurIPS 2020

Problem Define

Alleviate supervision collapse for few-shot generalization

- Supervision collapse



  - Labels assigned to real-world images with multiple entities only correctly describe a subset of the depicted content, leading to ambiguous classification results.

Source:

Method

- Summary of method
  - Leverage self-supervised pretraining (MIM) for FSL
  - Patch-level similarity (local correspondence) based classification (with ViT)
  - Token importance re-weighting for better classification

Source:

Method

Few-shot classification with Transformers Using Reweighted Embedding similarity

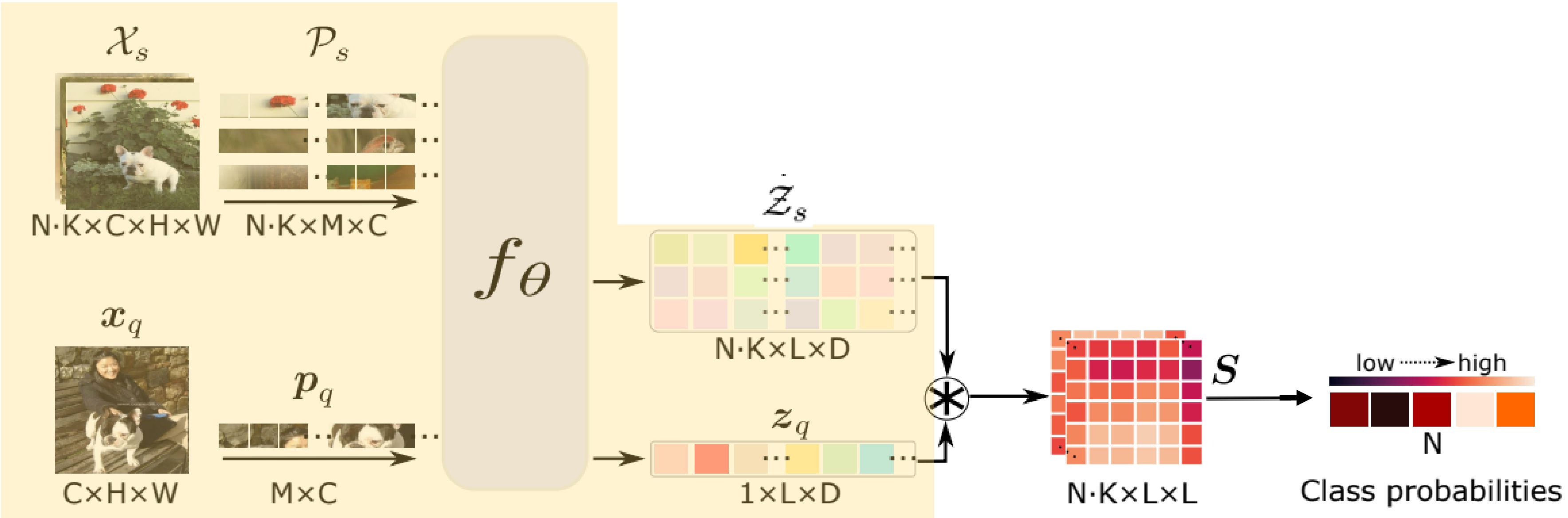
- Problem formulation
  - N-way K-shot classification with episodic training and testing $C_{train} \cap C_{test} = \emptyset$
  - An episode (train) is composed of a
    - ✓ Support set $\mathcal{X}_s = \{(x_s^{nk}, y_s^{nk}) | n = 1, \dots, N; k = 1, \dots, K; y_s^{nk} \in C_{train}\}$,
      where x_s^{nk} denotes the k-th sample of class n with label y_s^{nk}.
    - ✓ Query set $\mathcal{X}_q = \{(x_q^n, y_q^n) | n = 1, \dots, N\}$,
      where x_q^n denotes a query sample of class n with label y_q^n.
      (assume one query sample per class for easy understanding)

Source:

Method

Few-shot classification with Transformers Using Reweighted Embedding similarity

- Overview



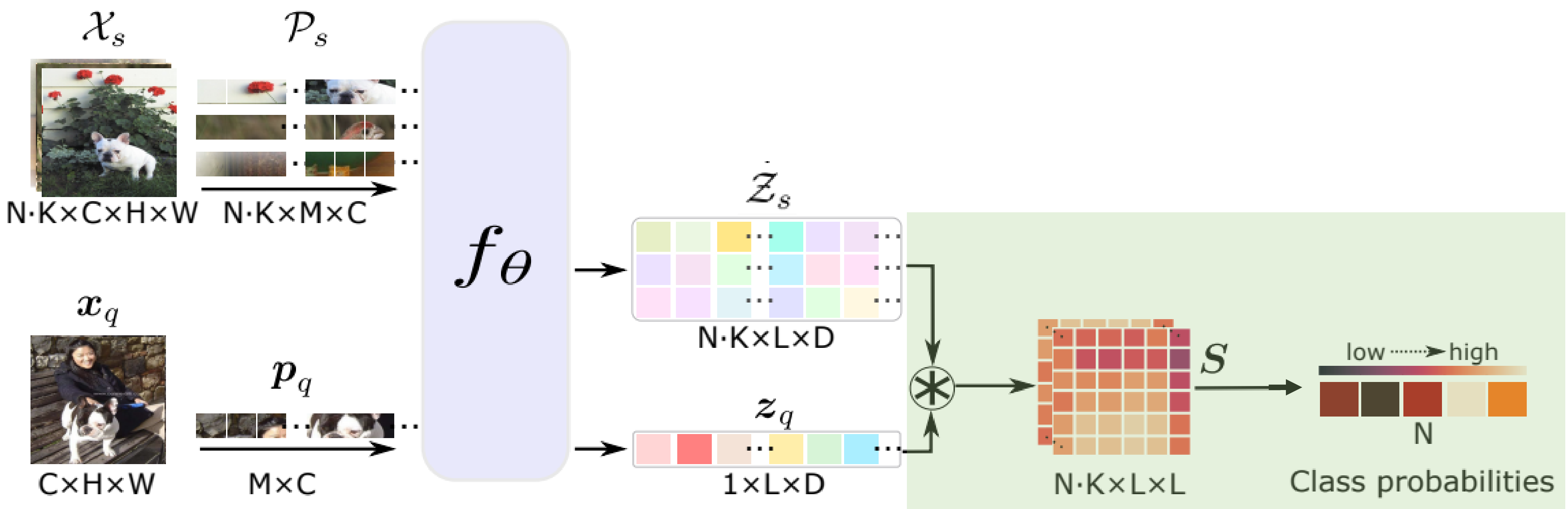
Extract support set and query embeddings from MIM pre-trained ViT backbone $f_\theta(\cdot)$

Source:

Method

Few-shot classification with Transformers Using Reweighted Embedding similarity

- Overview



Actually, we can directly derive the class probabilities from patch-wise similarity map

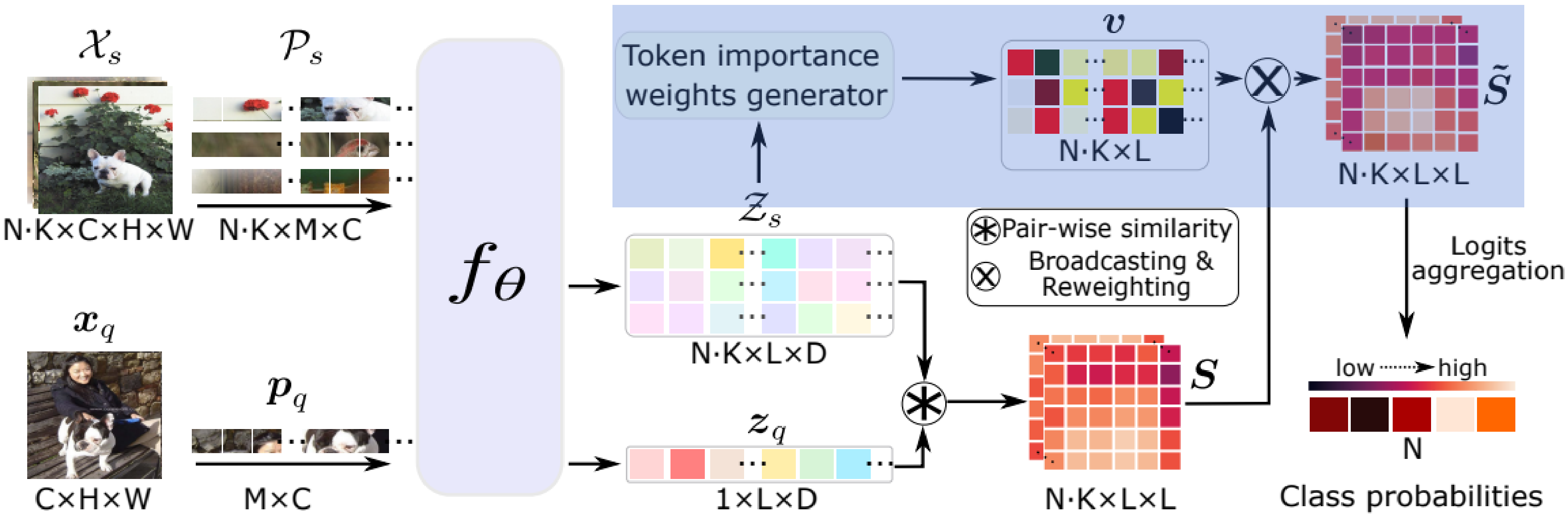
Source:

Method

Few-shot classification with Transformers Using Reweighted Embedding similarity

- Overview

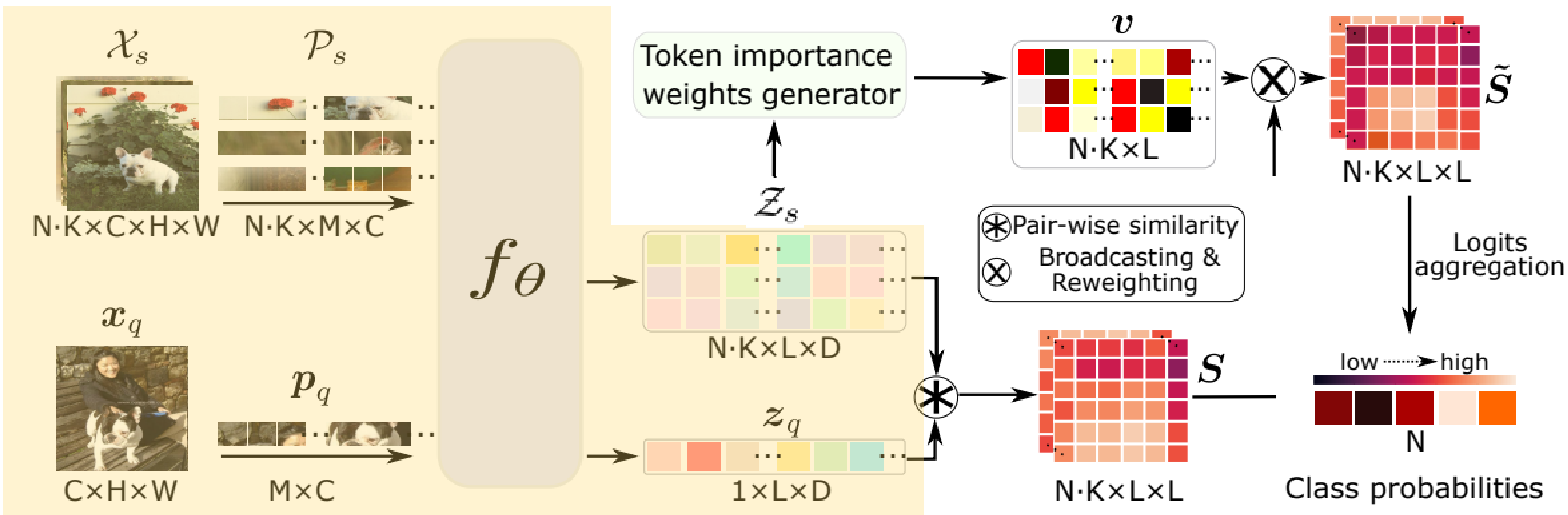
Authors devise task-specific token importance weights generating module for re-weighting the similarity map



Method

Few-shot classification with Transformers Using Reweighted Embedding similarity

- In-detail



In ViT, L=M,
In Swim , L<M

1. Split each input image $x \in \mathbb{R}^{H\times W\times C}$ into a sequence of $M = (H \cdot W)/P^2$ patches $\boldsymbol{p} = \{p^i\}_{i=1}^{M}$, with each patch $p^i \in \mathbb{R}^{P^2\times C}$

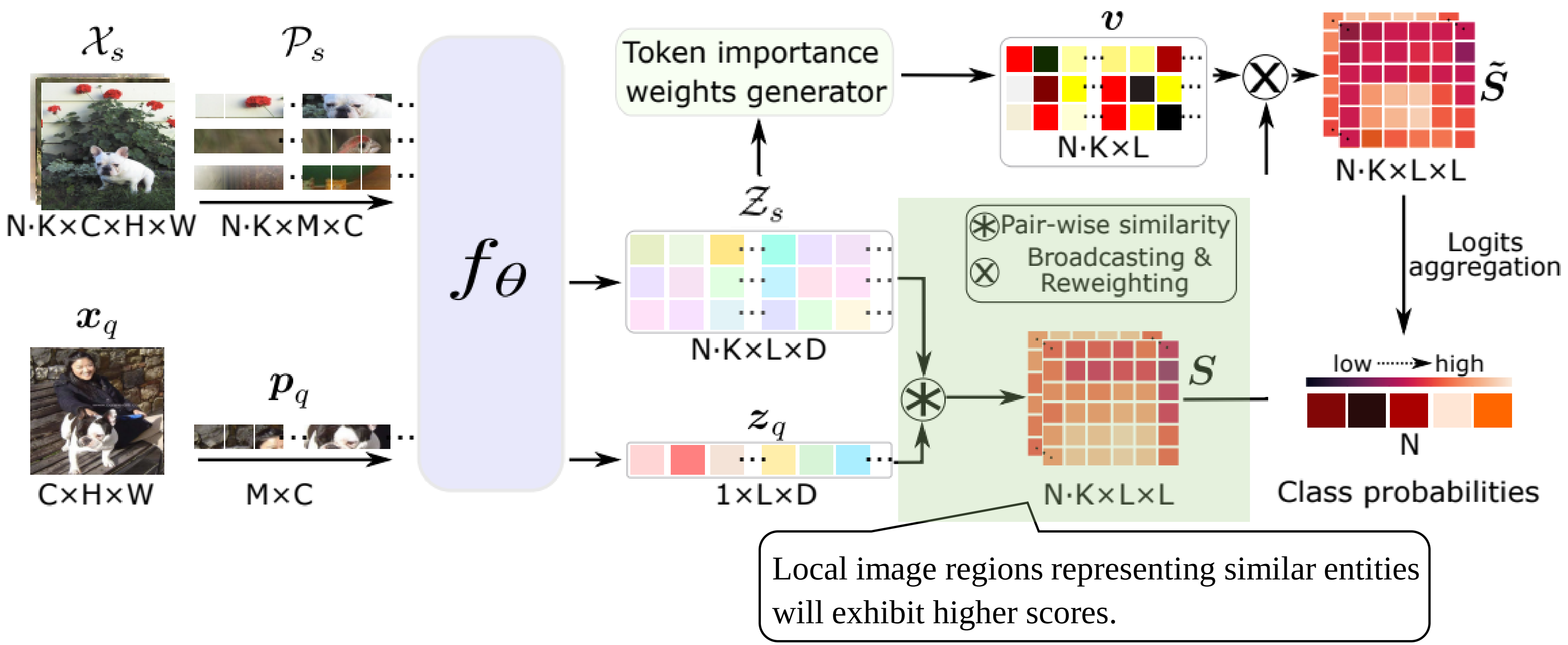
2. Fed those tokens into ViT encoder and obtain:
$Z_s = f_\theta(P_s)$ with $Z_s = \{\boldsymbol{z}_s^{nk} | n = 1, \ldots, N, k = 1, \ldots, K\}$, $\boldsymbol{z}_s^{nk} = \{z_s^{nkl} | l = 1, \ldots, L; z_s^{nkl} \in \mathbb{R}^D\}$
$z_q = f_\theta(\boldsymbol{p}_q)$ with $z_q = \{z_q^l | l = 1, \ldots, L; z_q^l \in \mathbb{R}^D\}$

Source:

Method

Few-shot classification with Transformers Using Reweighted Embedding similarity

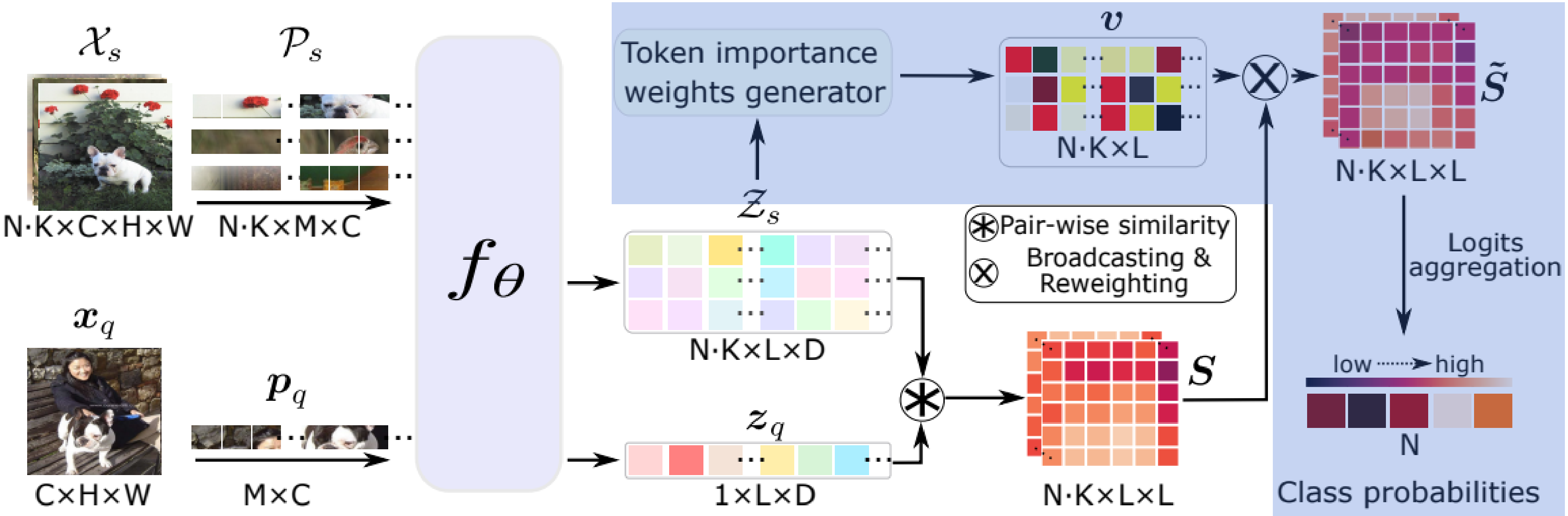
- In-detail



3. Based on patch embedding, pair-wise patch simiarlity matrix $\boldsymbol{S}$ is obtained by $s_{nk}^{l_s,l_q} = sim(z_s^{nkl_s}, z_q^{l_q})$, where $l_s = 1, \dots, L$ and $l_q = 1, \dots, L$

Source:

- In-detail



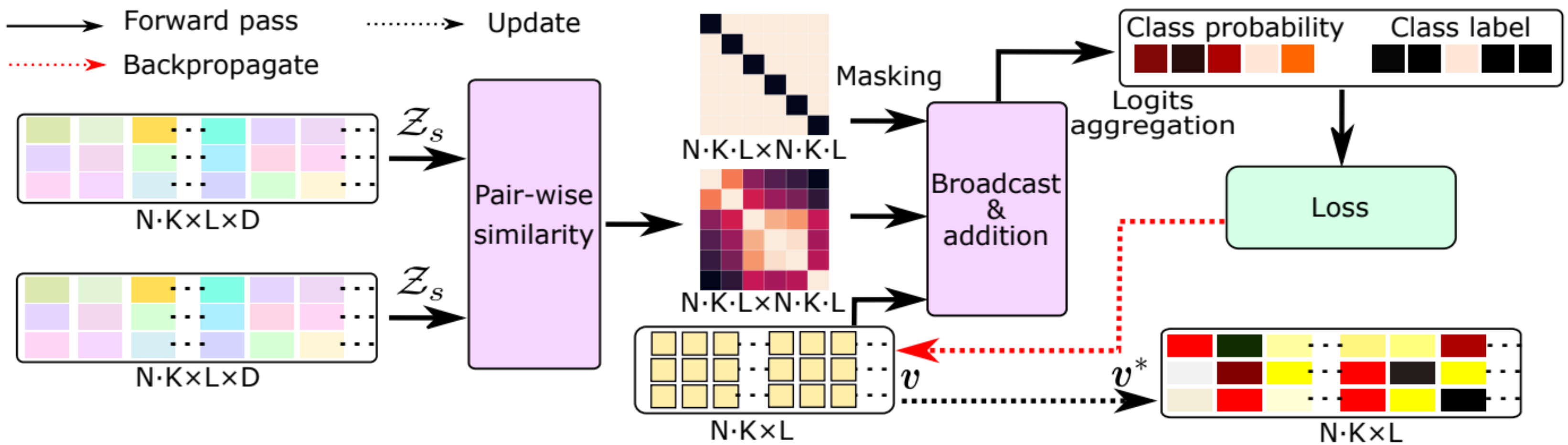
4. Task-specific token importance weights $\boldsymbol{v} \in \mathbb{R}^{N \cdot K \cdot L \times 1}$ is inferred via online optimization (inner loop) => will be described in next slide.

5. Reweighting similarity map as $\tilde{S} = S + [v \cdot 1^{1 \times L}]$ and aggregate the information as below

$$\boxed{\hat{\boldsymbol{y}}_q = \text{softmax}\left(\{\hat{y}_q^n\}_{n=1}^{N}\right) = \text{softmax}\left(\left\{\log \sum_{k=1}^{K} \sum_{l_q=1}^{L} \sum_{l_s=1}^{L} \exp\left(\tilde{s}_{nk}^{l_s,l_q} / \tau_S\right)\right\}_{n=1}^{N}\right)}$$

Source:

- In-detail: inner loop token importance weight generator.



- Support set is coppied as Z_s (original) and Z_{sq} (pseudo-query)
- Obtrain the similarity matrix $S_s \in \mathbb{R}^{N\cdot K\cdot L\times N\cdot K\cdot L}$
- Reweighted similarity matrix is computed as $\widetilde{S_s} = S_s + [v^0 + 1^{1\times N\cdot K\cdot L}]$
- Based on reweighted similarity matrix, perform classification using support set labels

$$\arg\min_{\boldsymbol{v}} \sum_{n=1}^{N}\sum_{k=1}^{K} \mathcal{L}_{\text{CE}}\left(\boldsymbol{y}_s^{nk},\ \hat{\boldsymbol{y}}_s^{nk}(\boldsymbol{v})\right)$$

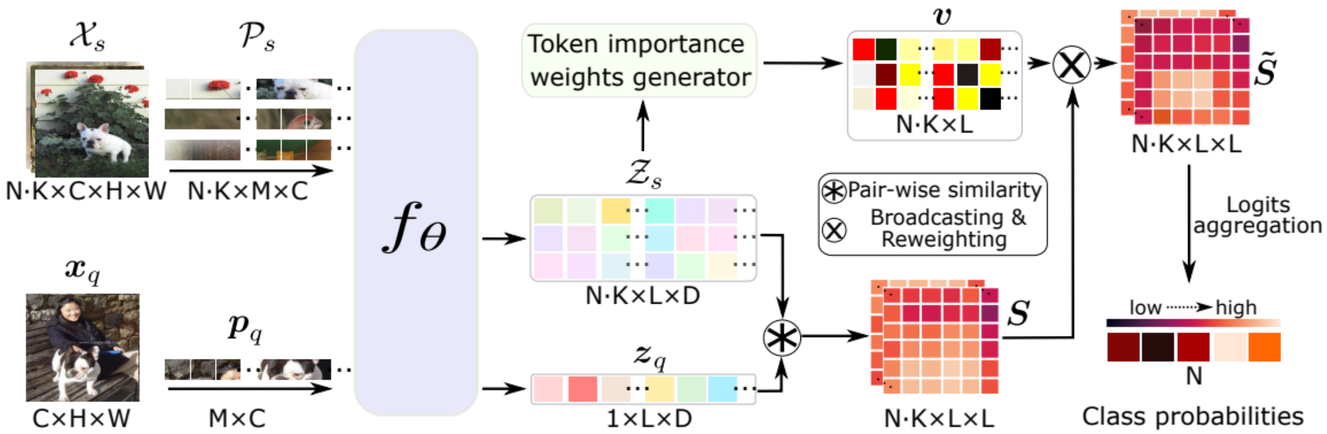
Same with previous equation, but N*K times rep.

Source:

Method

Few-shot classification with Transformers Using Reweighted Embedding similarity

- Train & Inferemce procedure



1. Self-supervised pretraining for f_θ
   ✓ With Masked Image Modeling (ibot) objective
2. Meta fine-tuning for f_θ and v
   ✓ With patch-wise similarity based classification objective

- Note, v is additionally updated in inference time with labelled support set.

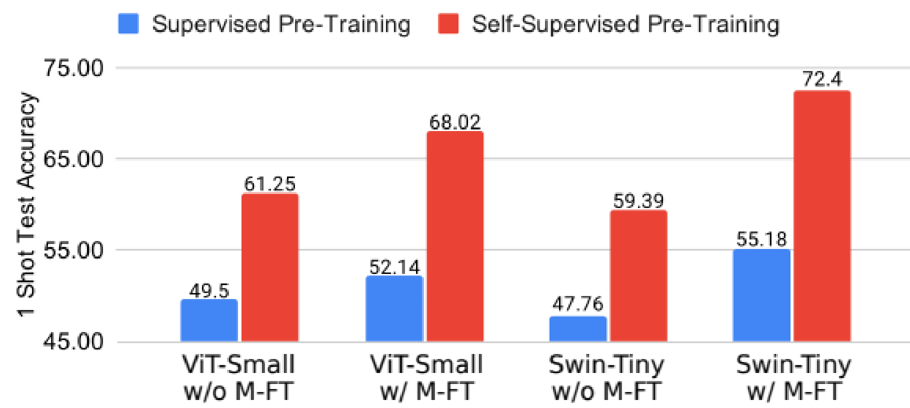
Source:

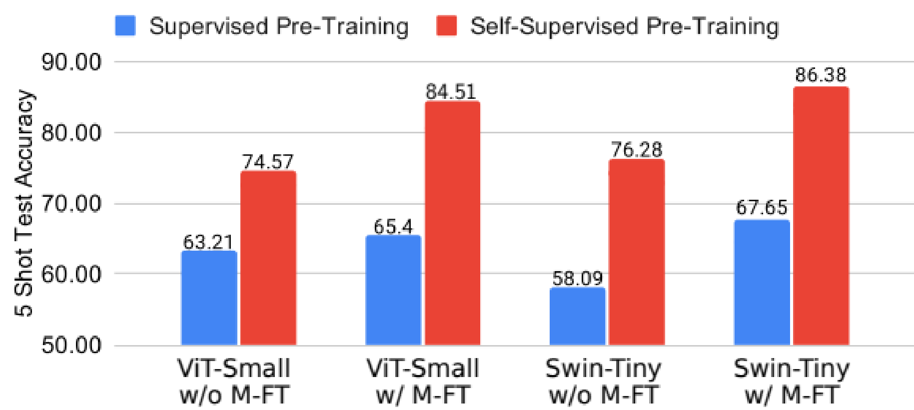
Results

Few-shot classification with standard benchmarks

- Dataset
  - miniImageNet
  - tieredImageNet
  - CIFAR-FS
  - FC-100
- Architecture
  - ViT-small
  - Swin-tiny

<Training strategy comparison>



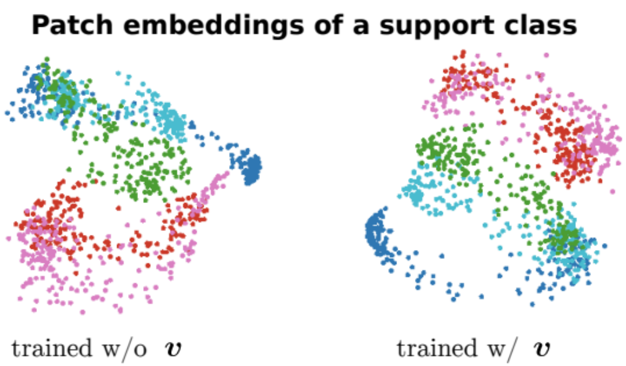


Messeage 1. supervised PT < SSL PT
Messeage 2. meta fine-tuning helps FSL largely. Especially for SSL PT backbone.

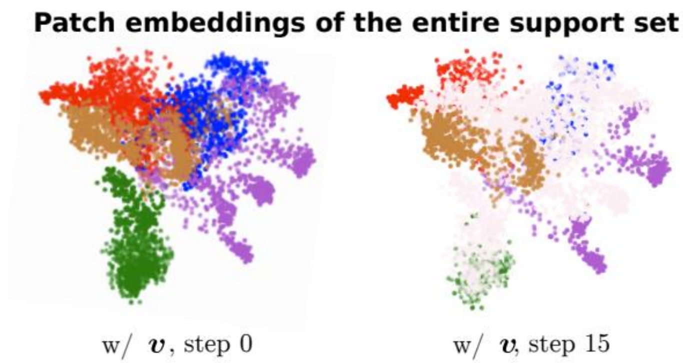
Source:

Few-shot classification with standard benchmarks

- Learned embedding analysis



<same class different instances>



<different classes different instances>

Messeage : the embeddings retain the instance information and separation is improved when using token importance reweighting.

I think their argument is inconsistent…

Messeage : token reweighting make the embedding space more disentangled between intra-class samples as well as inter-class samples.

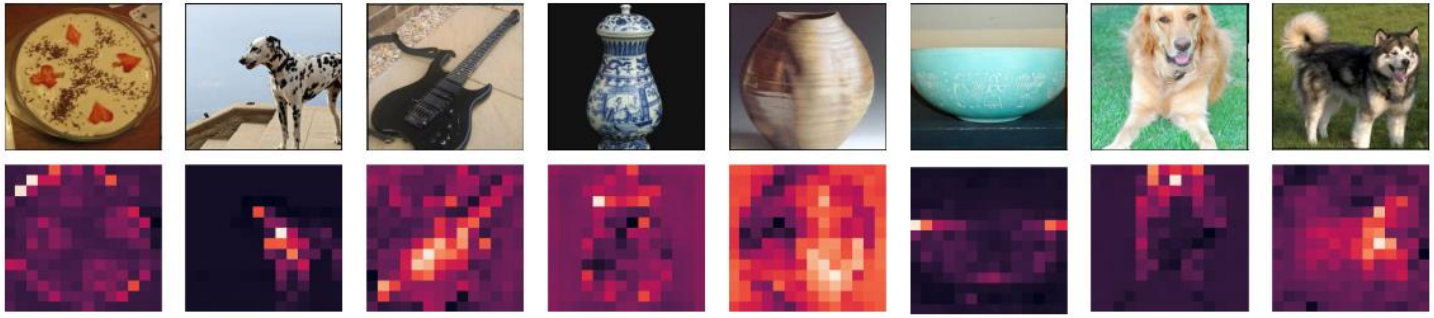
lack of explanation for visualization technique

Source:

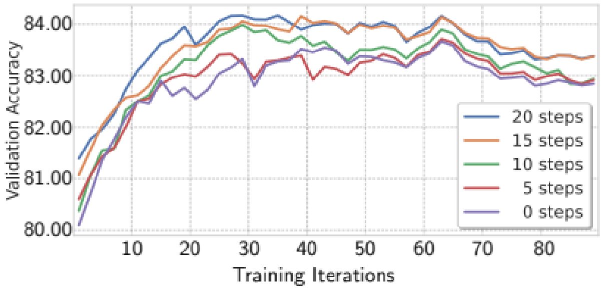
Results

Few-shot classification with standard benchmarks

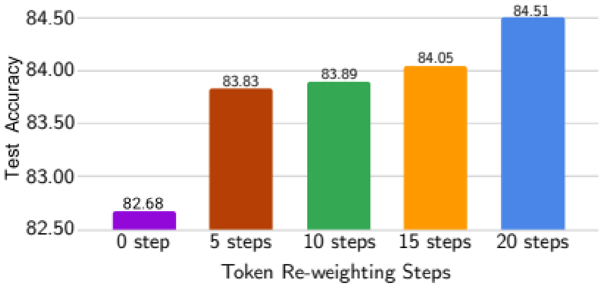
- Analysis on token importance re-weighting mechanism



Messeage : FewTRUE select characteristic regions of the depicted objects, and to exclude unimportant or out-of-task information.



(a)



(b)

Messeage : increasing inner loop steps up to 20 aligns with increased performance, both during validation and testing.

Source:

Results

Few-shot classification with standard benchmarks

- SOTA on benchmarks

Model	Backbone	≈ # Params	*mini*ImageNet 1-shot	*mini*ImageNet 5-shot	*tiered*ImageNet 1-shot	*tiered*ImageNet 5-shot
ProtoNet [41]	ResNet-12	12.4 M	62.29±0.33	79.46±0.48	68.25±0.23	84.01±0.56
FEAT [53]	ResNet-12	12.4 M	66.78±0.20	82.05±0.14	70.80±0.23	84.79±0.16
DeepEMD [54]	ResNet-12	12.4 M	65.91±0.82	82.41±0.56	71.16±0.87	86.03±0.58
IEPT [56]	ResNet-12	12.4 M	67.05±0.44	82.90±0.30	72.24±0.50	86.73±0.34
MELR [12]	ResNet-12	12.4 M	67.40±0.43	83.40±0.28	72.14±0.51	87.01±0.35
FRN [49]	ResNet-12	12.4 M	66.45±0.19	82.83±0.13	72.06±0.22	86.89±0.14
CG [58]	ResNet-12	12.4 M	67.02±0.20	82.32±0.14	71.66±0.23	85.50±0.15
DMF [52]	ResNet-12	12.4 M	67.76±0.46	82.71±0.31	71.89±0.52	85.96±0.35
InfoPatch [25]	ResNet-12	12.4 M	67.67±0.45	82.44±0.31	-	-
BML [60]	ResNet-12	12.4 M	67.04±0.63	83.63±0.29	68.99±0.50	85.49±0.34
CNL [58]	ResNet-12	12.4 M	67.96±0.98	83.36±0.51	73.42±0.95	87.72±0.75
Meta-NVG [55]	ResNet-12	12.4 M	67.14±0.80	83.82±0.51	74.58±0.88	86.73±0.61
PAL [29]	ResNet-12	12.4 M	69.37±0.64	84.40±0.44	72.25±0.72	86.95±0.47
COSOC [28]	ResNet-12	12.4 M	69.28±0.49	85.16±0.42	73.57±0.43	87.57±0.10
Meta DeepBDC [51]	ResNet-12	12.4 M	67.34±0.43	84.46±0.28	72.34±0.49	87.31±0.32
LEO [39]	WRN-28-10	36.5 M	61.76±0.08	77.59±0.12	66.33±0.05	81.44±0.09
CC+rot [15]	WRN-28-10	36.5 M	62.93±0.45	79.87±0.33	70.53±0.51	84.98±0.36
FEAT [53]	WRN-28-10	36.5 M	65.10±0.20	81.11±0.14	70.41±0.23	84.38±0.16
PSST [8]	WRN-28-10	36.5 M	64.16±0.44	80.64±0.32	-	-
MetaQDA [57]	WRN-28-10	36.5 M	67.83±0.64	84.28±0.69	74.33±0.65	89.56±0.79
OM [36]	WRN-28-10	36.5 M	66.78±0.30	85.29±0.41	71.54±0.29	87.79±0.46
FewTURE (ours)	ViT-Small	22 M	68.02±0.88	84.51±0.53	72.96±0.92	86.43±0.67
FewTURE (ours)	Swin-Tiny	29 M	**72.40±0.78**	**86.38±0.49**	**76.32±0.87**	**89.96±0.55**

Model	Backbone	≈ # Params	CIFAR-FS 1-shot	CIFAR-FS 5-shot	FC100 1-shot	FC100 5-shot
ProtoNet [41]	ResNet-12	12.4 M	-	-	41.54±0.76	57.08±0.76
MetaOpt [21]	ResNet-12	12.4 M	72.00±0.70	84.20±0.50	41.10±0.60	55.50±0.60
MABAS [20]	ResNet-12	12.4 M	73.51±0.92	85.65±0.65	42.31±0.75	58.16±0.78
RFS [45]	ResNet-12	12.4 M	73.90±0.80	86.90±0.50	44.60±0.70	60.90±0.60
BML [60]	ResNet-12	12.4 M	73.45±0.47	88.04±0.33	-	-
CG [14]	ResNet-12	12.4 M	73.00±0.70	85.80±0.50	-	-
Meta-NVG [55]	ResNet-12	12.4 M	74.63±0.91	86.45±0.59	46.40±0.81	61.33±0.71
RENet [19]	ResNet-12	12.4 M	74.51±0.46	86.60±0.32	-	-
TPMN [50]	ResNet-12	12.4 M	75.50±0.90	87.20±0.60	46.93±0.71	63.26±0.74
MixFSL [1]	ResNet-12	12.4 M	-	-	44.89±0.63	60.70±0.60
CC+rot [15]	WRN-28-10	36.5 M	73.62±0.31	86.05±0.22	-	-
PSST [8]	WRN-28-10	36.5 M	77.02±0.38	88.45±0.35	-	-
Meta-QDA [57]	WRN-28-10	36.5 M	75.83±0.88	88.79±0.75	-	-
FewTURE (ours)	ViT-Small	22 M	76.10±0.88	86.14±0.64	46.20±0.79	63.14±0.73
FewTURE (ours)	Swin-Tiny	29 M	**77.76±0.81**	**88.90±0.59**	**47.68±0.78**	**63.81±0.75**

Powerful, but, different backbone architectures can be justified?

Source:

Results

Few-shot classification with standard benchmarks

- Additional experiments
  - Token pruning

Table 3: Pruning the number of tokens. Test accuracy for 5-way 5-shot on *mini*ImageNet [48].

# tokens	Test Acc.
100%	84.05 ± 0.53
75%	83.15 ± 0.57
50%	83.81 ± 0.59
25%	81.79 ± 0.57
10%	81.05 ± 0.62

Pruning via attention map value thresholding

  - Ablation on classifier

Table 4: Changing the classifier. Test accuracy for 5-way 5-shot on *mini*ImageNet [48].

Classifier	Test Acc.
Prototyp. w/ Euclid. Dist.	82.80 ± 0.59
Prototyp. w/ Cosine. Dist.	79.90 ± 0.65
Linear (optimized online)	82.37 ± 0.57
FewTURE (0 rew. steps)	82.68 ± 0.55
FewTURE (15 rew. steps)	$\mathbf{84.05 \pm 0.53}$

Comparison with
- Other non-parameter classifier (proto)
- learnable parameteric classifier (linear)

Source:

Conclusion & Discussion

- Summary
  - Using ViT with SSL pre-training (MIM) for FSL for the first-time.
    - ✓ (To mitigate supervision collapse)
  - Propose a new patch-wise simiarlity based non-parametric classifier
  - Propose online token importance re-weighting module

- Discussion
  - The “supervision collapse” and “multiple entities” problem is not convincing…
    - ✓ Authors gave just illustration.
  - The experiments are only performed on standard single-object-centric dataset..
    - ✓ Few-shot variants of ImageNet / CIFAR100.
  - The idea underlying methodology is not completely new.
    - ✓ Consideration on local correspondence is already addressed by CAN (Hou et al. 2019)
    - ✓ SSL objective is also used in previous work CTX (Doersch et al. 2020)

Source: