

Cross Attention Network for Few-shot Classification

Abstract

- Few-shot classification 의 목표: 주어진 소수의 labeled samples 만으로 unlabeled samples 을 인식하는 것
- unseen classes and low-data problem은 few-shot classification 을 어렵게 만든다.
- 기존의 접근 방식은 labeled and unlabeled samples 에서 독립적으로 feature을 추출하기 때문에 추출된 feature는 충분히 discriminative하지 않다.
- few-shot classification의 challenging problems을 해결하기 위해 새로운 Cross Attention Network 제시
 1. unseen classes 의 문제를 해결하기 위해 **Cross Attention Module** 이 도입된다.
→ 추출된 feature를 더 discriminative 하게 만든다.
 2. low-data problem을 해결하기 위해 **transductive inference algorithm** 이 도입된다.
→ support set 을 augmentation 한다.(making the class features more representative)

few-shot learning

등장 배경

Meta-Learning : Learning-to-learn, 즉 학습을 잘 하는 방법을 학습하는 것에 대한 연구 분야

Meta-Learning의 한 종류로 소량의 데이터(few-shot)만으로도 뛰어난 학습을 하는 모델 만들어보자!

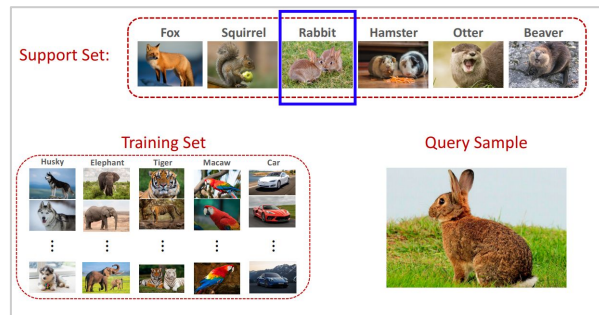
Few-shot learning(N-way K-shot learning)

: 모델이 추론하는 과정에서 소량의 데이터만 보고 추론을 할 경우

- ❑ one-shot: 1장의 데이터만 보고 추론을 할 경우
- ❑ zero-shot: 0장의 데이터만 보고 (즉 Task 조건만 입력하고) 추론을 할 경우

In K-shot learning, we have:

- N = support set의 class의 개수
 - K = support set의 각 class 당 labeled examples의 개수
 - train set
 - Support Set
 - Query Set
- Support Set, Query Set의 label \neq train set의 label

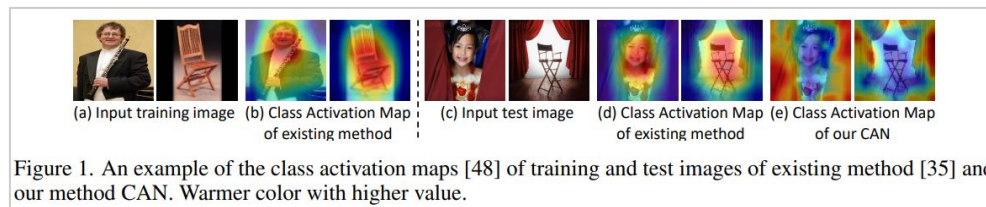
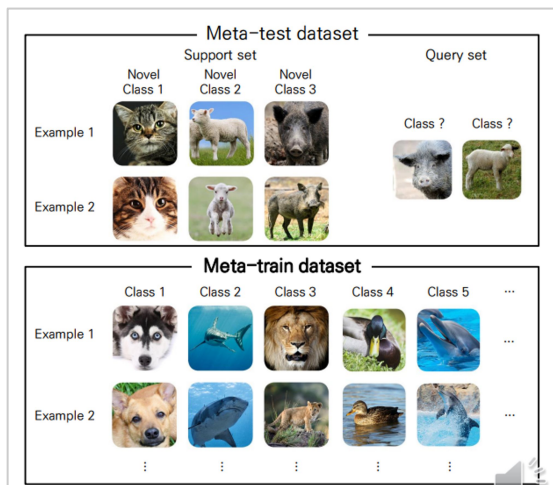


⇒ Train set을 통해 “구분하는 법을 배우”고, Query image가 들어왔을 때 이 Query image가 Support set 중 어떤 것과 같은 class인지 추론

INTRODUCTION

- Few-shot classification의 목표: 주어진 training set and support set 으로 unlabeled query samples 을 분류하는 것
- traditional classification과 비교했을 때, few-shot classification 는 2개의 main challenges 가 있다.
 - a. **unseen classes**: non-overlap between training and test classes
 - b. **low-data problem**: very few labeled samples for the test unseen classes

⇒ seen classes 로 train 된 모델이 few labeled samples(support set) 만을 가지고도 query set을 unseen classes 로 일반화 할 수 있어야 한다.



→ training set에 없는 object는 feature 추출이 어려움.

INTRODUCTION

- few-shot classification에 대한 feature discriminability를 향상시키기 위해 Cross Attention Network (CAN) 제안

1. unseen class problem을 해결하기 위해 **Cross Attention Module (CAM)** 도입

: class feature map and a query sample feature map이 주어진다면 CAM은 각 feature에 대한 cross attention map 생성하여 target object를 강조한다. (Correlation estimation and meta fusion이 사용)

→ test samples의 target object은 attention을 가질 수 있고 cross attention maps의해 weight 된 features는 더 discriminative 하다.

2. low-data problem을 완화하기 위해 unlabeled query set 전체를 활용하는 **transductive inference algorithm** 도입

: query samples 에 대한 label을 반복적으로 예측하고 pseudo-labeled query samples 을 선택하여 support set 을 augmentation 한다.

→ class 당 더 많은 support samples이 있으면, 획득한 class features 가 더 representative 할 수 있으므로 low-data problem 를 완화한다.

Problem Define.

- **Meta-Learning:** 여러개의 Task 를 동시에 학습 & 각 Task 간의 차이도 학습 (meta-parameter)

→ 전체 학습 이후 소량의 데이터 (few-shot) 으로도 추론 할 수 있는 범용적인 모델 생성

즉, 전체 데이터를 여러 개의 Support set, Query set으로 나누는 과정 필요: **episode training mechanism** 사용

- episode training mechanism

- train

- Query set의 label인 y 의 확률을 높이는 방향으로 parameter θ 를 최적화

$$\theta = \operatorname{argmax}_{\theta} E_{L \sim T} \left[E_{S \sim L, B \sim L} \left[\sum_{(x,y) \in B} \log P_{\theta}(y|x, S) \right] \right]$$

- test

- train에 사용되지 않은 새로운 label 로 분류 성능 확인

- setting

support set: C classes and K labeled samples per class → C-way K-shot problem

$$\text{support set } \mathcal{S} = \{(x_a^s, y_a^s)\}_{a=1}^{n_s} \quad (n_s = C \times K)$$

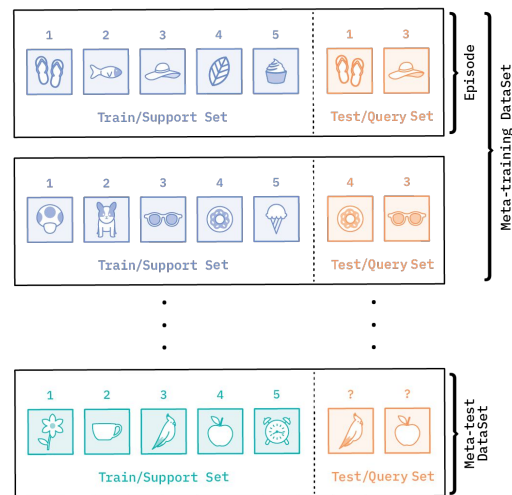
S_k = the support subset of the k-th class

$$\text{query set } \mathcal{Q} = \{(x_b^q, y_b^q)\}_{b=1}^{n_q} \quad (\text{C classes 의 나머지 샘플들의 일부})$$

- metric-learning based method 채택

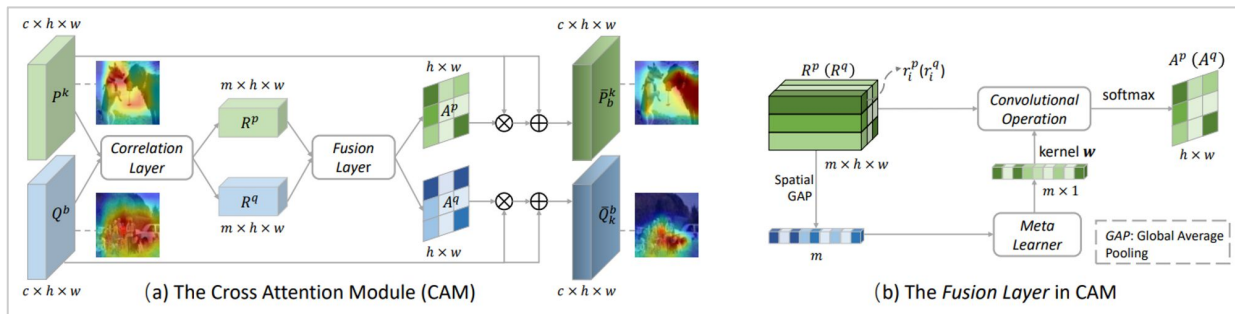
:support and query features 사이의 semantic relevance 를 기반으로 target object를 highlight 한다.

→ support class S_k 와 query sample x_b^q 의 feature representation 과 similarity 를 측정하는 방법 고안



CAM Overview

- cross Attention Module (CAM) 제안
 - **metric-learning** 사용: support class and query sample의 각 pair에 대한 적절한 feature representations을 얻음.
 - class feature and query feature 사이의 semantic relevance를 모델링하여 target objects에 attention 부여



(a) CAM(Cross Attention Module)

(b) CAM에서 Fusion Layer: $R^p(R^q) \in \mathbb{R}^{m \times m}$ 은 더 나은 시각화를 위해 $\mathbb{R}^{m \times h \times w}$ 로 reshape 됨.

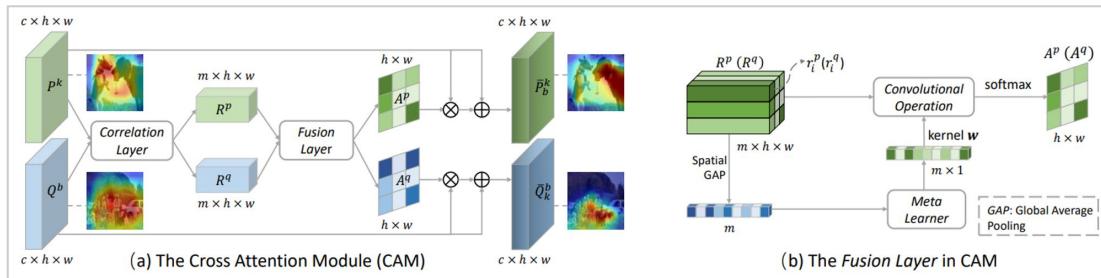
⇒ target object의 regions (coated retriever in the figure)에 적용되는 feature maps 생성

1. **Correlation Layer:** query/class 사이의 semantic relevance 계산 후, correlation map 생성
2. **Meta Fusion Layer:** class and query attention maps 각각 생성
3. **residual attention mechanism:** self attention을 통해 more discriminative feature maps 생성

CAM - Correlation Layer

- Correlation Layer

- metric-learning 사용: support class and query sample의 각 pair에 대한 적절한 feature representations을 얻음.



- input
 - class feature map $P^k \in \mathbb{R}^{c \times h \times w}$: support samples in S^k ($k \in \{1, 2, \dots, C\}$)
 - query feature map $Q^b \in \mathbb{R}^{c \times h \times w}$: query sample x_b^q ($b \in \{1, 2, \dots, n_q\}$)

c, h and w denote the number of channel, height and width of the feature maps respectively.

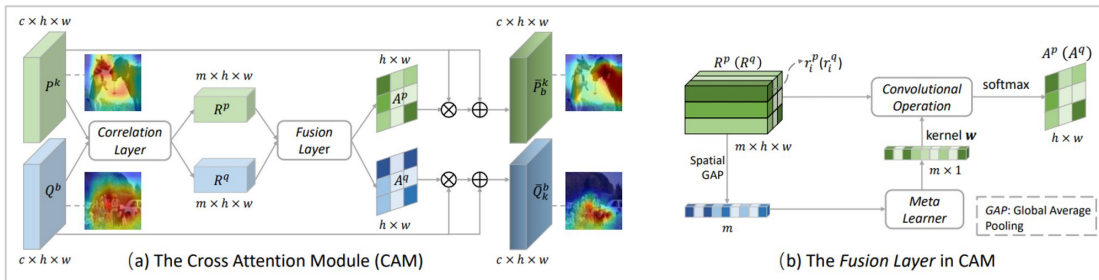
- output

: class feature map P^k and query feature map Q^b 사이의 semantic relevance 계산 후, **correlation map** 생성

CAM - Correlation Layer

- Correlation Layer

- metric-learning 사용: support class and query sample의 각 pair에 대한 적절한 feature representations을 얻음.



- correlation map 생성 절차 \Rightarrow attention score

1. **P와 Q를 $R^{c \times m}$, 즉 $P = [p_1, p_2, \dots, p_m]$ 과 $Q = [q_1, q_2, \dots, q_m]$ 으로 재구성**

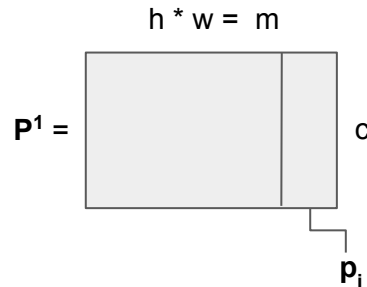
m ($m = h \times w$): 각 feature map의 spatial positions 의 수 = pixel 수

$p_i, q_i \in R^c$: 각각 P와 Q의 i 번째 spatial position에 있는 feature vectors

2. cosine similarity 로 semantic relevance 계산 후, correlation map $R \in R^{m \times m}$ 생성

$$R_{ij} = \left(\frac{p_i}{\|p_i\|_2} \right)^T \left(\frac{q_j}{\|q_j\|_2} \right), \quad i, j = 1, \dots, m.$$

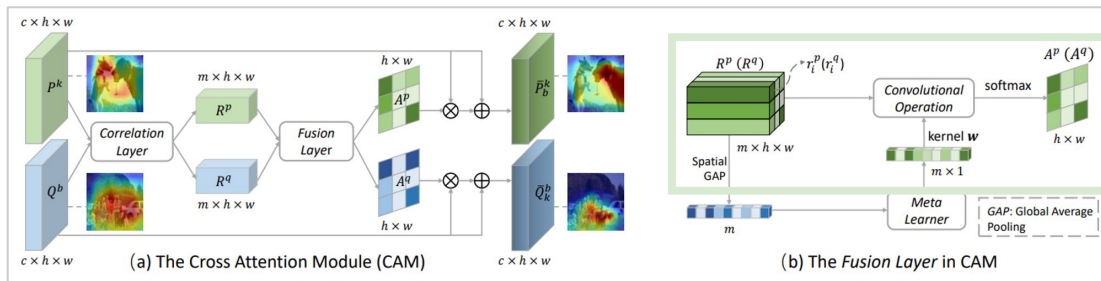
- class correlation map $R^p = R^T = [r_1^p, r_2^p, \dots, r_m^p]$ 과 query correlation map $R^q = R = [r_1^q, r_2^q, \dots, r_m^q]$
 - $r_i^p \in R^m$: local class feature vector p_i 와 모든 query feature vectors $\{q_i\}_{i=1}^m$ 사이의 relevance
 - $r_i^q \in R^m$: local query feature vector q_i 와 모든 class feature vectors $\{p_i\}_{i=1}^m$ 사이의 relevance



CAM - Meta Fusion Layer

- Meta Fusion Layer

- correlation maps $R^p(R^q)$ 을 기반으로 class and query attention maps $A^p(A^q)$ 을 각각 생성



- attention map 생성 절차

- $m \times 1$ kernel, $w \in \mathbb{R}^{m \times 1}$ 의 convolutional operation을 적용해서 R^p 의 각 local correlation vector $\{r_i^p\}$ 를 **attention scalar** 로 융합한다.
- softmax function 을 사용하여 attention scalar를 normalize 하고 i-th position에서 **class attention**을 얻는다.

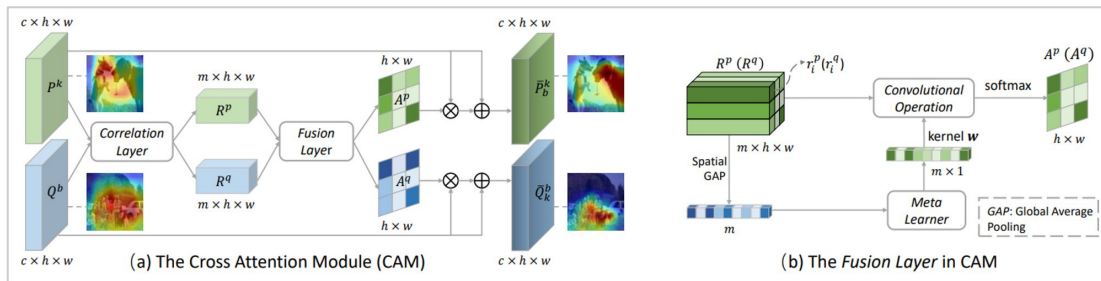
$$A_i^p = \frac{\exp((w^T r_i^p)/\tau)}{\sum_{j=1}^{h \times w} \exp((w^T r_j^p)/\tau)}, \quad \tau: \text{temperature hyperparameter}$$

- class attention map 은 A^p 를 matrix in $\mathbb{R}^{h \times w}$ 로 reshape함으로써 얻어진다.
- residual attention mechanism을 사용하는데, 여기서 초기 feature maps P and Q 는 요소적으로 (elementwisely) $1 + A^p$ 와 $1 + A^q$ 로 weight가 부여되어 보다 discriminative feature maps $P^- \in \mathbb{R}^{c \times h \times w}$ 와 $Q^- \in \mathbb{R}^{c \times h \times w}$ 를 각각 형성한다.

CAM - Meta Fusion Layer

- Meta Fusion Layer

- correlation maps $R^p(R^q)$ 을 기반으로 class and query attention maps $A^p(A^q)$ 을 각각 생성



- kernel w
: local class feature p_i 와 all local query features $\{q_j\}_{j=1}^m$ 사이의 correlations 을 i-th position에서 attention scalar 로 aggregate한다.
- $R^p \rightarrow$ global average pooling (GAP) operation(i.e., row-wise averaging)을 적용 \rightarrow 평균 class correlation vector 추출 \rightarrow meta-learning 에 입력하여 kernel $w \in \mathbb{R}^m$ 을 생성

$$w = W_2(\sigma(W_1(GAP(R^p))),$$

- $W_1 \in \mathbb{R}^{\frac{m}{r} \times m}$ and $W_2 \in \mathbb{R}^{m \times \frac{m}{r}}$: parameters of the meta-learner
- r : reduction ratio
- σ : ReLU function

Cross Attention Network

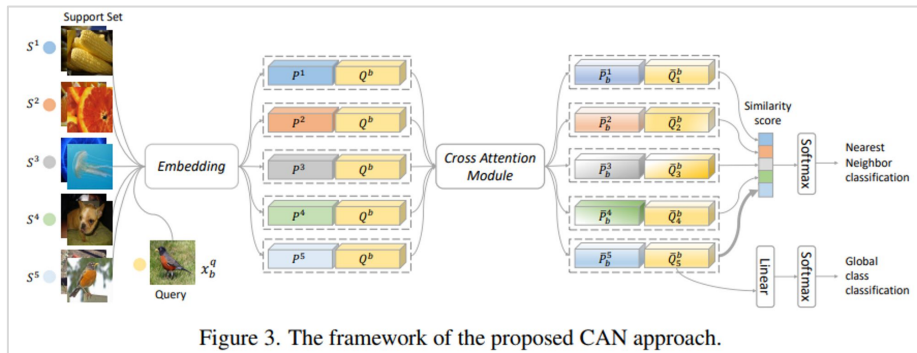


Figure 3. The framework of the proposed CAN approach.

- embedding module E**: input image $x \rightarrow$ feature map $E(x) \in \mathbb{R}^{c \times h \times w}$ 으로 매핑
 - several cascaded convolutional layers 으로 구성
 - class feature : embedding space 내에 support set 의 평균으로 정의
 - input: support set S , a query sample x_b^q
 - output: class feature map $P^k = \frac{1}{|S^k|} \sum_{x_a^s \in S^k} \hat{E}(x_a^s)$, query feature map $Q^b = E(x_b^q)$ 생성
- cross attention module**: feature map \rightarrow attention map
 - input: 각 feature maps(P^k, Q^b) pair
 - output: classification 을 위해 more discriminative feature pairs(P_b^k, Q_k^b) 을 출력
- classification module**
 - input: 각 attention maps(P_b^k, Q_k^b) pair
 - output
 - Nearest Neighbor classification: query samples \rightarrow C support classes 로 분류
 - Global class classification: query samples \rightarrow training set의 label로 분류

Model Training via Optimization

⇒ CAN은 query samples of training set 에 대한 classification loss 최소화함으로써 훈련
classification module : nearest neighbor and a global classifier으로 구성

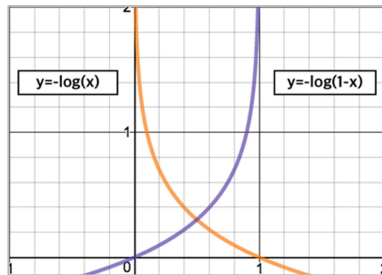
1. **nearest neighbor classifier** : pre-defined similarity measure(cosine) 을 기반으로 **query samples** → **C support classes** 로 분류
ex. i번째 position의 각 local query feature q_i^b 에 대해 nearest neighbor classifier 는 C support classes 에 대한 softmax-like label distribution 생성 ⇒ q_i^b 를 k번째 class로 예측할 확률

$$p(y = k | q_i^b) = \frac{\exp(-d((\bar{Q}_k^b)_i, \text{GAP}(\bar{P}_b^k)))}{\sum_{j=1}^C \exp(-d((\bar{Q}_j^b)_i, \text{GAP}(\bar{P}_b^j)))},$$

- $(\bar{Q}_k^b)_i$: \bar{Q}_k^b 의 i-th spatial position의 the feature vector
- GAP: mean class feature 을 얻기 위한 the global average pooling operation
- d: cosine distance, CAM에 의해 생성된 feature space에서 계산

loss

$$L_1 = - \sum_{b=1}^{n_q} \sum_{i=1}^m \log p(y = y_b^q | q_i^b).$$



Model Training via Optimization

⇒ CAN은 query samples of training set 에 대한 classification loss 최소화함으로써 훈련
classification module : nearest neighbor and a global classifier으로 구성

1. **global classifier** : fully connected layer 와 softmax를 통해 사용 가능한 모든 train class 중에서 각 query sample을 분류
training set 에 전체 l classes 가 있다고 가정.

local query feature q_i^b 에 대한 classification probability vector $z_i^b \in \mathbb{R}^l$ 은 $z_i^b = \text{softmax}(W_c(\bar{Q}_{y_b^q}^b)_i)$

global classification loss

$$L_2 = - \sum_{b=1}^{n_q} \sum_{i=1}^m \log \left((z_i^b)_{l_b^q} \right)$$


- $W_c \in \mathbb{R}^{l \times c}$: fully connected layer의 weight
 - $l_b^q \in \{1, 2, \dots, l\}$: x_b^q 의 true global class
-
- 전체 classification loss: $L = \lambda L_1 + L_2$ 로 정의

Transductive Inference

few-shot classification task 에서 각 class에는 labeled samples 이 많이 없기 때문에 class feature는 true class distribution 를 나타낼 수 없다.

⇒ 이러한 문제를 완화하기 위해 unlabeled query samples 을 활용하여 class feature 를 풍부하게 하는 transductive inference algorithm 제안

$$\hat{y}_b^q = \arg \min_k d \left(\text{GAP}(\bar{Q}_k^b), \text{GAP}(\bar{P}_b^k) \right)$$

1. 위의 식을 통해 가장 가까운 mean class feature 을 찾아 unlabeled query samples $\{x_b^q\}_{b=1}^{n_q}$ 의 label $\{\hat{y}_b^q\}_{b=1}^{n_q}$ 을 예측
2. query sample $x_b^q \leftrightarrow$ nearest class neighbor c_b^q 사이의 cosine distance 를 사용하여 label confidence 기준을 정의한다.
 $c_b^q = \min_k d(\text{GAP}(Q_k^b), \text{GAP}(P_k^b))$, : higher the confidence of the predicted label $\{\hat{y}_b^q\}$

Based on this criterion,

3. candidate set $D = \{(x_b^q, \hat{y}_b^q) | s_b = 1, x_b^q \in Q\}$ 생성
 $s_b \in \{0, 1\}$: query sample x_b^q 에 대한 selection indicator
4. $s_b \in \{0, 1\}^{n_q}$ 는 top t개의 신뢰할 수 있는 query samples $s = \arg \min_{||s||_0=t} \sum_{b=1}^{n_q} s_b c_b^q$ 에 의해 결정
5. support set S 와 함께 candidate set D 를 사용하여 more representative class feature map $(P^k)^*$ 생성

$$(P^k)^* = \frac{1}{|S^k| + |D^k|} \left(\sum_{x_a^s \in S^k} E(x_a^s) + \sum_{x_b^q \in D^k} E(x_b^q) \right)$$

$$D^k = \{(x_b^q, \hat{y}_b^q) | x_b^q \in D, \hat{y}_b^q = k\}$$

Experiments

Datasets.

1. ILSVRC-12의 subset인 **minilImageNet** 사용
 - 클래스당 600개의 이미지가 있는 100개의 클래스가 포함
 - standard split: 64 classes for training, 16 for validation and 20 for testing
2. ILSVRC-12의 훨씬 큰 하위 집합인 **tieredImageNet** 사용
 - 34개의 카테고리 와 총 608개의 클래스가 포함
 - split: 20 categories (351 classes) for training, 6 categories (97 classes) for validation, and 8 categories (160 classes) for testing

Experimental setting.

- **5-way 1-shot / 5-way 5-shot settings**
- episode 구성: C classes and each class includes K support samples 포함, 6 and 15 query samples are used for training and inference respectively.
- When inference: test set 에서 랜덤하게 2000 episodes 추출
→ 2000 episodes 에 대한 average accuracy 와 해당 95% confidence interval 을 보고

Experiments

Table 1. Comparison to state-of-the-arts with 95% confidence intervals on 5-way classification on minilImageNet and tieredImageNet datasets.

model		Embedding	IT(s)	miniImageNet		tieredImageNet	
				1-shot	5-shot	1-shot	5-shot
O	MAML [7]	ConvNet	0.103	48.70 \pm 0.84	55.31 \pm 0.73	51.67 \pm 1.81	70.30 \pm 1.75
	MTL [36]	ResNet-12	2.020	61.20 \pm 1.80	75.50 \pm 0.80	-	-
	LEO [33]	WRN-28	-	61.76 \pm 0.08	77.59 \pm 0.12	66.33 \pm 0.05	81.44 \pm 0.09
	MetaOpt [14]	ResNet-12	0.096	62.64 \pm 0.62	78.63 \pm 0.46	65.99 \pm 0.72	81.56 \pm 0.53
P	MetaNet [20]	ConvNet	-	49.21 \pm 0.96	-	-	-
	MM-Net [3]	ConvNet	-	53.37 \pm 0.48	66.97 \pm 0.35	-	-
	adaNet [21]	ResNet-12	1.371	56.88 \pm 0.62	71.94 \pm 0.57	-	-
M	MN [40]	ConvNet	0.021	43.44 \pm 0.77	60.60 \pm 0.71	-	-
	PN [35]	ConvNet	0.018	49.42 \pm 0.78	68.20 \pm 0.66	53.31 \pm 0.89	72.69 \pm 0.74
	RN [37]	ConvNet	0.033	50.44 \pm 0.82	65.32 \pm 0.70	54.48 \pm 0.93	71.32 \pm 0.78
	DN4 [15]	ConvNet	0.049	51.24 \pm 0.74	71.02 \pm 0.64	-	-
	TADAM [26]	ResNet-12	0.079	58.50 \pm 0.30	76.70 \pm 0.30	-	-
	Our CAN	ResNet-12	0.044	63.85 \pm 0.48	79.44 \pm 0.34	69.89 \pm 0.51	84.23 \pm 0.37
T	TPN [17]	ResNet-12	-	59.46	75.65	-	-
	Our CAN+T	ResNet-12	-	67.19 \pm 0.55	80.64 \pm 0.35	73.21 \pm 0.58	84.93 \pm 0.38

- IT: **Inference Time per query data** in a 5-way 1-shot task on one NVIDIA 1080Ti GPU.
- CAN+T: transductive inference 가 있는 CAN
- **optimization-based (O), parameter-generating (P), metric-learning (M) and transductive methods (T)**

Experiments

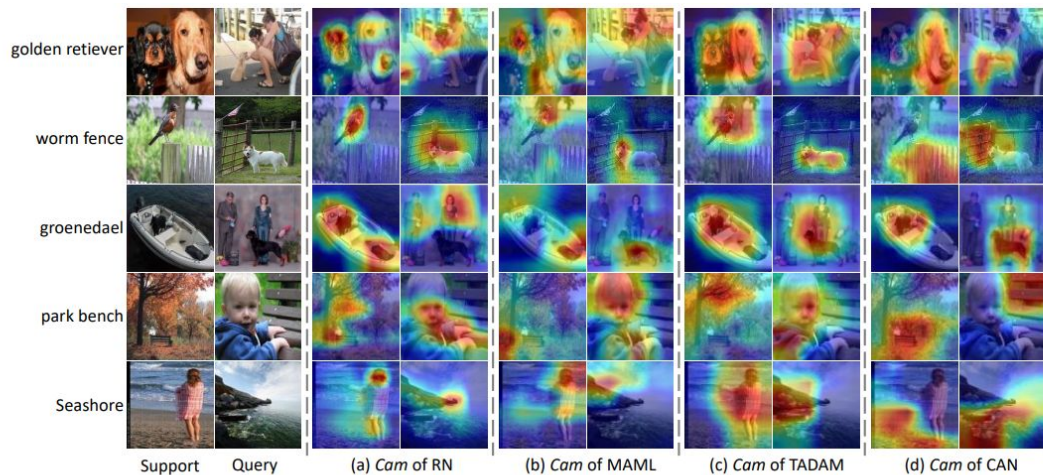


Figure 4. Class activation mapping (*Cam*) visualization on a 5-way 1-shot task with 1 query sample per class.

To qualitatively evaluate the proposed cross attention mechanism,

- we compare the class activation maps [48] visualization results of CAN to other **meta-learners**, RN, MAML and TADAM.

Conclusion

⇒ we proposed a **cross attention network** for few-shot classification.

1. a cross tention module: class and query features 사이의 semantic relevance를 모델링하기 위해 설계 → relevant regions localize 하고 more discriminative features 를 생성할 수 있다.
2. the low-data problem을 완화하기 위한 transductive inference algorithm 제안
→ It utilizes the unlabeled query samples to enrich the class features to be more representative.
3. Extensive experiment: our method is far simpler and more efficient than recent few-shot meta-learning approaches, and produces state-of-the-art results.