We thank the reviewers for their positive comments on the novelty and performance improvement. We will release the code for paper reproduction and facilitating using and building upon this method as **R1** and **R3** suggested.

R1: ***Qualitative comparisons via visualization to other meta-learners, especially to task-adaptive meta-learners.***

A: We compare the visualization results of CAN to other meta-learners, Relation Network (RN)[33], MAML [6] and TADAM [23]. As shown in Fig. 1 (a), the features of RN usually contain non-target objects since it lacks an explicit mechanism for feature adaptation. MAML performs gradient-based adaptation, which makes the model merely learn some conspicuous discriminative features in the support images without deeping into the intrinsic characteristic of the target objects. As shown in Fig. 1 (b), MAML attends to *ship* for the *groenendael* support image to better distinguish it from the *golden retriever* category, resulting in a confusing location and misclassification of the *groenendael* category. TADAM performs task-dependent adaptation and applies the **same** adaptive parameters to all query images of a task, thus it is difficult to locate different target objects for different categories. As shown in Fig. 1 (c), TADAM mistakenly attends to the *dog* for *worm fence* query image. In contrast, CAN processes the query samples with **different** adaptive parameters, which allows it to focus on the different target objects for different categories shown in Fig. 1 (d). We will add these qualitative comparisons into the main text in the final version.

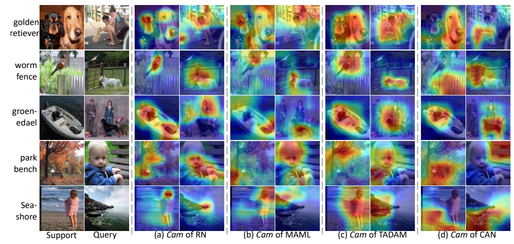

Figure 1. Class activation mapping (*Cam*) visualization on a 5-way 1-shot task with 1 query sample per class.

R1: ***Ablation that modifies a standard Prototypical Network to use the proposed feature-wise distance metric.***

A: Following your suggestion, we compare the standard Prototypical Network (PN) with Prototypical Network using feature-wise distance metric (PN-F) on miniImageNet. PN-F only brings a marginal improvement while CAN significantly outperforms it (PN/PN-F/CAN: 1-shot accuracy: 61.30/61.94/**63.95**, 5-shot accuracy: 76.70/76.91/**79.44**). The results further verify that the significant improvement of CAN to PN is due to the proposed cross attention module.

R1: ***Apply the proposed joint training schema to other commonly-used meta-learners.***

A: We try another two meta-learners, Matching Network (MN) [36] and Relation Network (RN) [33], to further verify the effectiveness of the proposed joint learning schema . We re-implement MN and RN with ResNet12 as backbone on miniImageNet. As shown in Tab. 1, our joint training schema (-JT) significantly improves the performance with respect to different meta-learners.

Table 1. Performance with joint training.

models	MN	MN-JT	RN	RN-JT
1-shot	55.29	**59.14**	51.25	**54.29**
5-shot	67.74	**73.81**	64.45	**67.58**

R1: ***Experiment on a dataset of cluttered scenes for few-shot classification.***

A: Following your suggestion, we use a more cluttered dataset, a scene recognition dataset miniPlaces365 [1]. A scene image usually contains multiple objects, while not all the objects are related to this scene. Therefore, it requires the models to accurately locate the target objects for correct classification. We compare CAN to MN, RN and PN with the same backbone and joint-training schema on miniPlace365. CAN achieves more gain, with an improvement up to 6% (MN/RN/PN/CAN: 1-shot: 48.16/44.52/48.34/**54.44**). The results demonstrate that CAN is more efficient on cluttered scenes. For qualitative analysis, we compare the visualization results in Fig 2. As can be seen, other methods usually highlight non-target objects, while CAN can attend to the targets among multiple objects of the input images.

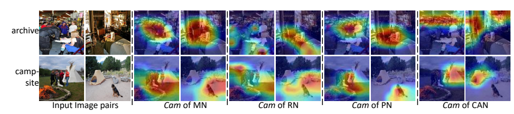

Figure 2. *Cam* visualization on input pairs from same class.

R2: ***Compare time complexity to other methods.***

A: (***i***) Our cross attention module (CAM) only increases marginal time cost. The cross-correlation maps between a query image and all support images can be simply worked out by one matrix multiplication, which is lightweight when it is used in high-level, sub-sampled feature maps. To illustrate the extra cost of CAM, we compare the time cost of the backbone for feature extraction and CAM for cross-correlation estimation in a 5-way 1-shot task. The backbone takes 0.041s for a query data, while CAM only takes 0.002s, equivalent to only $\sim$4% relative time increase over the backbone. (***ii***) Tab. 2 further compares the time cost of our method to others. Some methods [36,31,33,13,6] use a 4-layer *Conv* as the backbone thus take relatively lower time cost. Even though, our CAN is still comparable even superior to these methods in term of time cost, with a performance improvement up to 10%. The others use the same backbone as CAN, but require following up modules such as model update per task [32,12], gradient-based parameter generation [19], or expensive condition generation [23], which all incur more time overhead than CAM. Overall, Tab. 2 shows that CAN outperforms other methods without excessive overhead. We will report the time complexity of different methods in the comparison table (Tab1 in the main paper) in the final version.

Table 2. Time overhead of different methods. All times are reported per query data in a 5-way 1-shot task on one NVIDIA 1080Ti GPU.

model	MN[36]	PN[31]	RN[33]	DN4[13]	MAML[6]	MTC[32]	MetaOptNet[12]	adaNet[19]	TADAM[23]	CAN
test time (s)	0.021	0.018	0.033	0.049	0.103	2.02	0.096	1.371	0.079	0.044

R3: ***Novelty of transductive method and Release the code for reproduction.***

A: Thank you for your positive comments on the writing and performance improvement. (***i***) For the second contribution, the transductive method, we are the first to explore the idea that incorporates the unlabeled **query data** to **refine prototypes** in a meta-learning setup. We demonstrate it is effective to alleviat the *low-data* problem on transductive few-shot setting, which outperforms prior work [15] by a large margin, up to 8% improvement. (***ii***) All implementation details of CAN are given in the 'Experiment Setup' section. We will release the code and trained models for paper reproduction. In addition, we will submit the code with the camera-ready version once this paper is accepted.

[1] We randomly select 100 classes with 600 images per class from the training set of Places365 to form miniPlace365. The classes are divided into 60 classes for training, 20 classes for validation and 20 classes for testing. The input images size is 84×84.