We thank the reviewers for the positive comments, highlighting that our *'motivation, explanation, ablation and numerical performance are all quite good'*, with *'consistent improvement over the other algorithms'* on two benchmarks comparing ten different methods, and an *'icing-on-the-cake real medical application'*. Both R2 and R3 point out the contribution of combining global and local feature alignments with episodic training for domain generalization (DG).

R1's main concern regarding the DG problem setting is extensively clarified. We ran additional experiments regarding ResNet-18/50 for R2, and JiGen as baseline for R3 (please see the table at the end). This rebuttal also clarifies all other minor questions. We will add all these in the final version to further strengthen our contribution.

**Response to Reviewer #1**

**Problem setting:** **(i)** DG considers how to learn a model for a *single task* from a number of source domains and test it on unseen domains, in contrast with MAML's assumption of training on a variety of learning tasks for solving new tasks. **(ii)** We also clarify that DG is different from domain adaptation (DA), as DG assumes *no data is available* from the target domain during training (L22–25). **(iii)** In DG, source and target domains correspond to joint distributions $P_k(\mathbf{x}, y)$ and $P_*(\mathbf{x}, y)$ defined over input and label spaces $\mathcal{X} \times \mathcal{Y}$. It assumes there exist domain-invariant patterns (i.e. *semantic features*) in the marginals $P_k(\mathbf{x})$ and $P_*(\mathbf{x})$, which can be extracted to learn an estimate of $P(y \mid \mathbf{x})$ that performs well across seen and unseen domains. **(iv)** Thanks for pointing out the theoretical papers on multi-source to single-target adaptation; we will revise the Sec. 2 accordingly. Our DG definition and experimental setting follow the wide literature [1, 12, 22–25, 27, 31, 32] on this topic, but we agree a more theoretical discussion would be beneficial.

**Design choices:** **(i)** The class-specific average feature $\bar{\mathbf{z}}_c^{(k)}$ is considered as a compact semantic 'concept' of each class. Computing soft labels from the features, rather than averaging final predictions, reflects our goal of explicit regularization in feature space. **(ii)** We found no major theoretical reason to prefer Jensen–Shannon (JS) over symmetrized KL (a.k.a. Jeffreys divergence) in our context. In preliminary experiments, we did try JS but obtained worse empirical results. **(iii)** The 'linear-sized random subset of pairs' (L188) means that we can obtain an efficient unbiased $O(N)$ estimator of the loss by e.g. shuffling and iterating over $(2i-1, 2i)$, $i = 1, \ldots, \lfloor N/2 \rfloor$, rather than enumerating $O(N^2)$ pairs.

**Experiments:** **(i)** All results reported for our method and baselines are the average over 3 runs. Error bars in Table 3 are standard deviation. We will add error bars and statistical significance to Tables 1, 2, and 4. **(ii)** We clip the gradients to prevent them from exploding, because our inner meta-update needs to be implemented with plain gradient descent (not using an off-the-shelf optimizer). This follows the practice of MAML. **(iii)** We chose the margin $\xi$ heuristically, based on preliminary observations of the distances within and between the clusters of class features. **(iv)** Tables 1 and 2 have different columns because not all of those papers reported results on both benchmarks.

**Response to Reviewer #2**

**Engineering issues:** **(i)** We had no difficulty in setting the hyperparameters (e.g. learning rates and loss coefficients). Our heuristic choices worked well and other trials did not show much change. **(ii)** Computing second-order gradients does not excessively slow down training—in MAML [10] (basis of our meta-learning scheme), it is roughly 33% slower than a first-order approximation. **(iii)** Our $\mathcal{L}_{\text{global}}$ can scale to numerous domains, by randomly sampling subsets of meta-train and meta-test domains at each iteration, similarly to how MAML uses mini-batches of tasks. As our datasets have only few domains, we used all of them ($|\mathcal{D}_{\text{tr}}|$=2 and $|\mathcal{D}_{\text{te}}|$=1, with one hold-out test domain).

**ResNet backbone:** Thank you for the suggestion. We now ran experiments with ResNet-18/50 on the PACS benchmark. Our initial results shown in the table (mean $\pm$ std. dev. over 3 runs) are very promising, where our MASF consistently improves over DeepAll baseline. We will add more systematic ResNet experiments in the final version.

**Response to Reviewer #3**

**Local loss:** Note that we employ *either* contrastive *or* triplet loss for local alignment, but not both simultaneously. While contrastive loss has cheaper computational cost, it enforces much tighter constraints than triplet loss. As we argue in L179–182, contrastive loss is a good choice for complex tasks with mild domain shift (e.g. medical image segmentation), and triplet loss is adopted when domains are radically different (e.g. PACS benchmark).

**Additional baseline:** We reproduced the results of JiGen [3] using their released code, and present here preliminary results of using JiGen as the baseline with our proposed $\mathcal{L}_{\text{global}}$ (mean $\pm$ std. dev. over 3 runs). We find that our global alignment is indeed complementary to JiGen's task-agnostic loss. Thanks for the inspiring suggestion. We will further study the generic efficacy of our global and local semantic feature alignments in future work.

| | ResNet-18 | | ResNet-50 | | JiGen as baseline (with AlexNet) | | |
|---|---|---|---|---|---|---|---|
| Domain | DeepAll | MASF (ours) | DeepAll | MASF (ours) | JiGen [3] | Reproduced | $+\mathcal{L}_{\text{global}}$ |
| Art-painting | $77.38 \pm 0.15$ | $80.29 \pm 0.18$ | $81.41 \pm 0.16$ | $82.89 \pm 0.16$ | 67.63 | $67.60 \pm 0.06$ | $68.36 \pm 0.10$ |
| Cartoon | $75.65 \pm 0.11$ | $77.17 \pm 0.08$ | $78.61 \pm 0.17$ | $80.49 \pm 0.21$ | 71.71 | $71.82 \pm 0.17$ | $71.91 \pm 0.11$ |
| Photo | $94.25 \pm 0.09$ | $94.99 \pm 0.09$ | $94.83 \pm 0.06$ | $95.01 \pm 0.10$ | 89.00 | $89.66 \pm 0.12$ | $89.80 \pm 0.09$ |
| Sketch | $69.64 \pm 0.25$ | $71.69 \pm 0.22$ | $69.69 \pm 0.11$ | $72.29 \pm 0.15$ | 65.18 | $65.52 \pm 0.15$ | $66.73 \pm 0.15$ |