Author response for "Incremental Few-Shot Learning with Attention Attractor Networks"

We thank all reviewers for their time and insightful comments. We address individual comments below.

To R1 on the normalization constant: Thank you for pointing this out. In our experiments we actually used the same normalization constant (same softmaxes) for both support set and query set, both having W_a. Predictions for the base classes are simply ignored when calculating the loss for the support set. We will fix the paper to clarify this.

To R1 on *mini-* vs. *tiered-*ImageNet: *tiered*-ImageNet contains many more images, so this allows the network to be pretrained with a better feature representation, and we are also able to fit it with a larger and deeper network (ResNet-18 vs. ResNet-10). The challenge in tiered ImageNet lies in the split on higher level class categories, so there is a domain shift in training, validation and testing.

To R1 on computation overhead: Thank you for asking. We are not directly inverting the matrix but computing the inverse matrix vector product. Second, we are using truncated RBP, which computes a low-rank approximation of the matrix inverse, i.e. doing Jacobian transpose vector product for a fixed number of steps (see Line 15 in Alg. 1). In the experiments, we used 20 steps, so this has the same time complexity as truncated BPTT for 20 steps, while RBP saves memory by not having to store intermediate activations. We will add these clarifications in the paper.

To R1 on code release: We will release the code that can reproduce the experiments if the paper gets accepted. The release is expected to happen before the conference.

To R2 on catastrophic forgetting literature: Thank you for providing the references. We will cite them and include them in our discussion of related work. [1] uses the diagonal approximation of the Fisher matrix to prevent the parameter from drifting too far. Since our backbone network is frozen, this wouldn't be an issue in our setting, however, it could be complementary to our method if we choose to finetune the backbone network as well. [2] stores a few data points of old tasks in the "Coreset" so that they can estimate the variational distribution, whereas in our case, the regularizer is learned through meta-learning.

To R2 on disjoint classes & data augmentation: Data augmentation is only applied during pretraining, and for simplicity not applied during meta-learning. To ensure that in meta-learning novel classes do not overlap with base classes, we mask out the base classes that are used in the few-shot episode.

To R2 on θ_E in Figure 1: Thanks for the suggestion. We will modify Figure 1 to match with the notations. The meta parameters θ_E is enclosed in the teal part of the figure, which is learned during meta-learning stage, i.e. for mini ImageNet, there are 64 training classes, and the query set will be 59+5 classes, where there are 5 "fake" novel classes.

To R2 on other notation issues: Thanks for pointing them out. We will adopt your suggestions and change the notation to ensure consistency.

To R2 on Δ_a and Δ_b: These two measure how much drop in accuracy is caused by jointly classifying base and novel classes. We first record the accuracy by classifying base classes alone (i.e. standard classification problem), and novel classes alone (i.e. standard few-shot problem), and then observe the accuracy of base and novel classes in a mixed query set. Lastly, we take the difference of these two accuracies to be the drop in performance (Δ_a and Δ_b).

To R3 on motivation of attention mechanism: The motivation is to assess the similarity between the novel classes and the base classes, so that during the learning of novel classes, the system learns to differentiate from the base classes and prevent interference. The attention vector is used to retrieve an attractor to regularize the episodic training.

To R3 on clarity: Could you comment on the specific places that you feel are not clearly written? We will revise these sections in the next version.

To R3 on OptimizerStep: Line 12-17 is the part of the algorithm that computes the gradients of the meta-parameters θ_E. "OptimizerStep" is an external function that performs optimization given some gradient direction (e.g. gradient descent, momentum, Adam, etc.).

To R3 on T-BPTT: The reason why T-BPTT fails is that it only optimizes for a short horizon (e.g. 20-100 steps of gradient descent). When we train the episodic objective till convergence, T-BPTT is no longer trained for that, so the accuracy goes down for longer iterations.