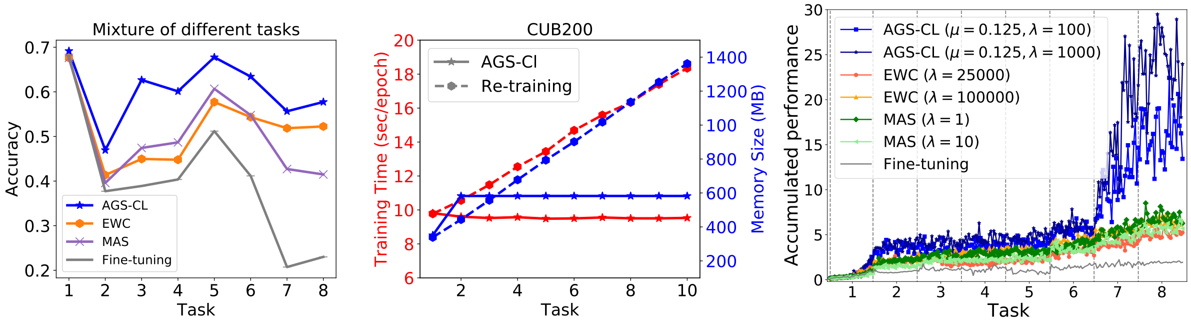

Figure 1: Results on 8 visually different tasks (left), comparison with Re-training (middle), and Atari RL (right).

Mixture of different tasks [R4] We carried out an additional continual learning experiment on eight tasks (as in [33, manuscript]) that consist of vision datasets with *different* domains:{CIFAR-10 / CIFAR-100 / MNIST / SVHN / Fashion-MNIST / Traffic-Signs / FaceScrub / NotMNIST}. Figure 1(left) compares the average accuracy of AGS-CL with two most stable baselines, EWC and MAS, and Fine-tuning. We clearly observe AGS-CL again significantly dominates EWC and MAS, confirming the effectiveness of our approach in a more challenging setting.

Comparison with Re-training [R1] For CUB200, we compared AGS-CL with Re-training, which takes a pre-trained AlexNet and re-trains for each task with the entire training sets observed so far. In terms of accuracy, AGS-CL (82.3%) was just slightly lower than Re-training (86.5%), which is an obvious upper bound. The clear benefit of AGS-CL, however, is shown in Figure 1(middle) in terms of the training time per epoch (red) and memory requirement (blue). We note both AGS-CL and Re-training had 40 training epochs for each task, and for the required memory, Re-training needs to store all the training data observed so far, whereas AGS-CL needs to store one additional AlexNet model and $\{\Omega^t_{n_\ell}\}_{n_\ell \in \mathcal{G}}$. We can clearly observe from the figure that the training time and memory for AGS-CL remain constant, whereas those for Re-training grow linearly (*i.e.*, *very expensive*), as the number of tasks grows. We believe this result clearly responds to **[R1]** and justifies the necessity of continual learning with AlexNet.

Re-initialization [R3,R4] We stress that the **order** of the re-initializations are important, *i.e*, [Zero-init] is always followed by [Rand-init]. (We will re-emphasize this in the final version.) Also, note that the outgoing weights of an unimportant node can be connected to *either* important *or* unimportant nodes in the upper layer. In such a case, our [Zero-init] *nullifies* and *fixes* those connected to the important nodes (Re:**[R3]**), but [Rand-init] re-utilizes those connected to the unimportant nodes so that they can become learnable for the next task (Re:**[R4]**). This is also illustrated in the rightmost network figure in Figure 2 (manuscript); note the activation of the unimportant (gray) node in the second layer can be used for learning the unimportant (gray) node in the third layer for task $t + 1$ (via the gray, solid weight between them that is randomly re-initialized). We will make sure to give a more clarified explanation on the re-init schemes in our final version.

RL [R3,R4] In the RL setting, it is nontrivial to implement SI/RWALK, which require to compute the gradient path integrals, or HAT, which implements hard attention mask for each task. EWC is the only scheme among our baselines that had results for RL, with the limitation mentioned in line 60, and that was the reason why we originally only compared with it. Now, since MAS operates almost similarly as EWC (in terms of the learning process), we also implemented MAS for the Atari RL tasks, and Figure 1(right) shows the results (for a single random seed) together with AGS-CL, EWC, and Fine-tuning. We observe MAS performs almost the same as EWC, and again, AGS-CL convincingly outperforms both. We will add this result as well as the explanation on why we excluded other baselines in the final version.

[R1] ① We respectively disagree that our paper has only incremental contributions. As also clearly listed by **[R4]** and appreciated by **[R3]**, we believe our method combines several different principles in a novel way for continual learning. In our opinion, *"It is somehow . . . regularization."* is quite vague, which makes it hard to make a systematic rebuttal. ② Eq. (5) is a general expression that *defines* the proximal gradient descent. In AGS-CL, the proximal operator in Eq.(5) can be applied independently for each node, and Eq.(5) becomes Eq.(7). This is also clearly mentioned in line 170∼173, and we hope this helps to resolve the confusion. ③ Regarding the re-initialization, we cannot see why using an *intuition* would be problematic. Numerous algorithms for deep learning, *e.g.*, dropout, batch normalization, attention mechanism, or weight initialization, are based on sound intuitions with little theoretical explanation. They are shown to be very effective, typically via extensive experiments, which we believe is also done in our paper as well (*e.g.*, Figure 5(b)(c)).

[R3] ① Our method can be also applied to the online continual learning setting. However, since the PGD leads to the sparsification or freezing of weights after a couple of epochs, the performance could be limited for the 1 epoch setting.

[R4] ① We mentioned in line 45∼46 that PGD was used as a tool to elegantly optimize our loss function and is described in details in Section 3.3. We will make sure to more clearly motivate PGD in the final version. ② Thanks for the comments. We will write a clear 1-paragraph summary to provide readers with a better overview of our method.