



(a) Individual Contributions

(b) #Parameters

(c) #Exemplars

1 We thank the reviewers for the constructive feedback. **Code** will be made public. Fig. (a, b, c) best viewed in zoom.

2 **Reviewer 2**

3 **R2.1: Difference from PathNet:** Our RPS-Net is inspired by PathNet, yet there are notable differences: 1) *Architecture*:  
4 PathNet adds all modules in a simple feed-forward manner, while RPS-Net adds modules as residual connections to  
5 an identity path ( $M_{skip}$ ) which can always be learnt even when all the modules are saturated (L263). RPS-Net can  
6 therefore adapt to new changes more robustly. 2) *Training setting*: PathNet uses evolutionary algorithm while RPS-Net  
7 employs a random path selection. Since PathNet usually involves only two tasks, an evolutionary algorithm with large  
8 explorations/generations is viable. However, for our case i.e., 10+ tasks, PathNet is not feasible, due to a large number  
9 of possible explorations. See **R3.1** for comparison between random selection and genetic algorithms.

10 **R2.2: Impact of Varying Exemplars:** Fig (c) compares RPS-Net with the best existing method (iCARL) for various  
11 memory budgets of exemplars on CIFAR100. Our proposed RPS-Net consistently performs better across all budgets.

12 **R2.3: Model Size Comparison:** Fig (b) compares total parameters across tasks for Progressive Nets [21], iCARL  
13 [20] and our RPS-Net on CIFAR100. Our model effectively reuses previous parameters, and the model size does not  
14 increase significantly with tasks. After 10 tasks, RPS-Net has 72.26M parameters on average, compared with iCARL  
15 (21.3M) and Progressive Nets (932.84M). In RPS-Net parameters & flops increase logarithmically (L293), while for  
16 Progressive Nets they increase quadratically. We will add a detailed discussion in the final version.

17 **R2.4: Impact of Augmentation:** For a fair comparison, we use *same* augmentation as in iCARL [20].

18 **R2.5: Reproducing Algo. in Sec. 3:** We'll improve the description of Sec. 3, and release our codes for reproducibility.

19 **R2.6: Ablation Study:** Fig (a) studies the impact of progressively integrating individual components of our RPS-  
20 Net: baseline Single Path (37.97%)  $\Rightarrow$  distillation (44.93%)  $\Rightarrow$  controller  $\phi(k, \gamma)$  (51.76%)  $\Rightarrow$  multi-path selection  
21 (58.48%). These results (classification accuracy) on CIFAR100 indicate that the most significant gains come from  
22 controller (+6.83%) and multi-path selection modules (+6.72%), which are our two major contributions. Further, our  
23 two contributions, controller and multi-path selection, provide a combined gain of 13.6% over baseline + distillation.

24 **Reviewer 3**

25 **R3.1: Ablation Analysis:** Impact of different components are shown in Fig (a) on CIFAR100. See **R2.6** for details.

26 **R3.2: Difference from Genetic Algorithms:** We compare our random selection with a genetic algorithm i.e., Binary  
27 Tournament Selection (BTS) for 25 maximum generations, on MNIST with 5 tasks (each of 2 classes), using a simple 2  
28 layer (100 neurons) MLP with  $M = 8$ ,  $J = 1$ . On 5 runs, our proposed random selection achieves an average accuracy  
29 of 96.52% vs BTS gets 96.32%. For same time complexity as ours, BTS has an average accuracy of 71.24% for the  
30 first generation models. For BTS to gain similar performance as our random selection, it needs an average of 10.2  
31 generations ( $> \#$  random paths), hence BTS has more compute complexity. Sophisticated genetic algorithms may beat  
32 random selection, but likely with a high compute cost, which is not suitable for an incremental classifier learning setting  
33 having multiple tasks. Please also see **R2.1**, and L159, L160. We'll elaborate this further in the final version.

34 **R3.3: Network Size and Capacity:** Fig (b) shows the comparison on CIFAR100. Also see **R2.3** for details.

35 **R3.4: Temperature  $t_e = 2$**  is empirically set on a small cross-validation set and *fixed* throughout experiments.

36 **Reviewer 4**

37 **R4.1: Revising the Claims:** We first thank R4 for appreciating technical innovations and contributions of our work.  
38 We also thank R4 for valuable suggestions to improve the manuscript writing. In the final version, we will revisit and  
39 tone down some of our claims as recommended.

40 **R4.2: Error Bars:** For five runs, mean and standard deviation of RPS-Net for tasks 1 to 10 are respectively:  $88.96 \pm 0.27$ ,  
41  $81.07 \pm 0.22$ ,  $77.15 \pm 0.55$ ,  $72.98 \pm 0.56$ ,  $68.69 \pm 0.58$ ,  $66.71 \pm 0.43$ ,  $64.47 \pm 0.34$ ,  $62.06 \pm 0.23$ ,  $60.3 \pm 0.58$ ,  $58.1 \pm 0.49$ .  
42 We will add this in the final version.

43 **R4.3: Backwards Transfer:** We validate Backwards Transfer with BWT metric (See Eq. 3 in GEM [16], larger the  
44 better). After last task, BWT values are  $-0.1462$  (RPS-Net) vs.  $-0.4602$  (iCARL). We will add the results with further  
45 discussion in the final version.

46 **R4.4: Significance Beyond Incremental Classifier Learning:** We thank R4 for the useful insight and will explore  
47 the applicability of our approach to other domains in future.