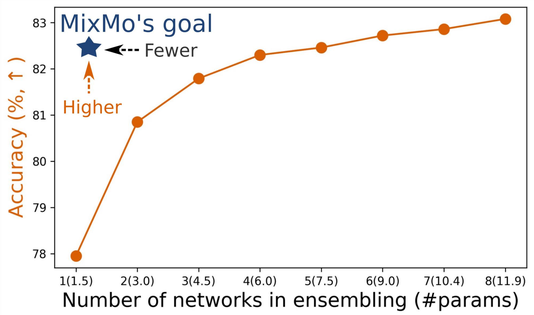
MixMo: Mixing Multiple Inputs for Multiple Outputs via Deep Subnetworks

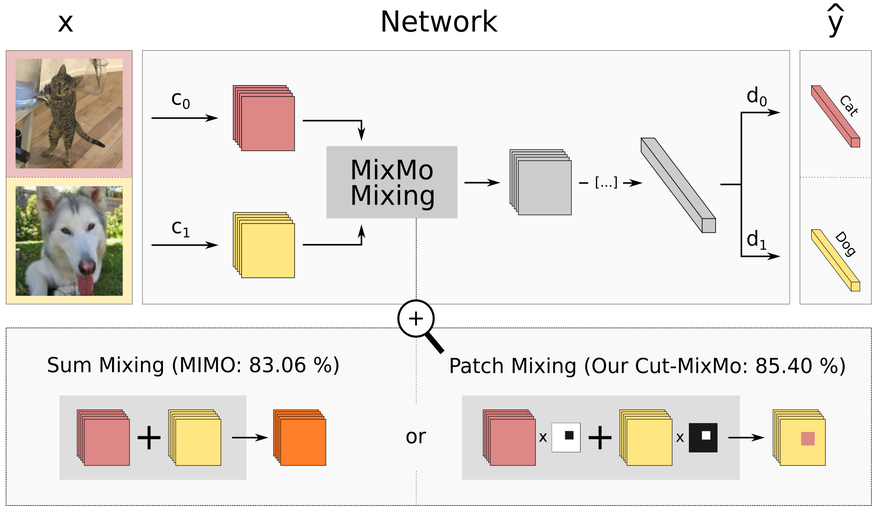
Alexandre Ramé[†*], Rémy Sun[*] and Matthieu Cord

Github: alexrame/mixmo-pytorch

1 Towards cheaper ensembling



2 Learning different subnetworks for ensembling by mixing multiple inputs



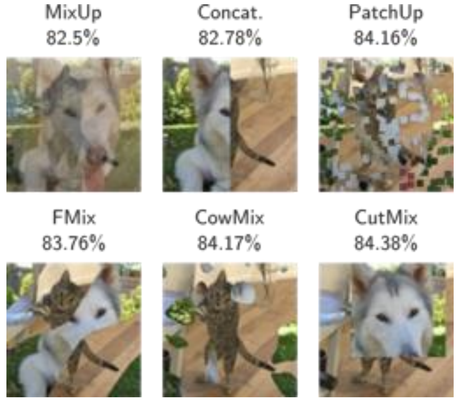
MixMo keeps computational costs fixed by finding subnetworks within a large base network :

1. 2 encoders embed 2 pictures (*e.g.,* a cat and a dog) into a shared feature space, where they are mixed.
2. A core network processes the mixed representation.
3. 2 classifiers each predict the label of one input (*e.g.,* the label cat and the label dog).

At inference, we consider two copies of the same input and ensemble the two predictions like MIMO [1].

3 Masking improves MixMo mixing

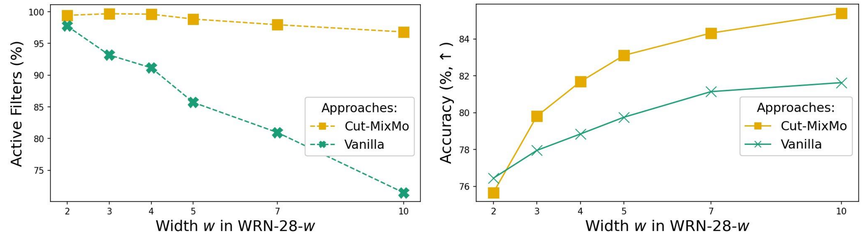
Our mixing block can use many methods:



Binary masking methods (*e.g.* CutMix [2]) perform better in our mixing block than linear interpolations (*e.g.* MixUp).

4 Leveraging over-parametrization in wide networks yields state-of-the-art performances

Approach	#Params	WRN-28-10 CIFAR100	WRN-28-10 CIFAR10	ResNet-18-3 TinyImageNet
Vanilla	1.0	81.63	96.34	65.78
CutMix	1.0	84.05	97.23	68.95
Deep Ens.	2.0	83.17	96.67	68.38
MIMO	1.002	83.06	96.74	68.48
Cut-MixMo	1.002	**85.40**	**97.51**	**70.24**



While wide over-parametrized models often fail to use all their filters, MixMo models fully leverage theirs to obtain state of the art results.

References

[1] MIMO: Havasi *et al.* Training independent subnetworks for robust prediction. ICLR 2021

[2] CutMix: Yun *et al.* Regularization strategy to train strong classifiers with localizable features. CVPR 2019.