Investigation and novel implementation of *batchboost*: a regularization and stable learning technique

Members: *Piyush Raj Gupta; Gaurav Kumar; Bhushan Chaudhari [19BM6JP31, 19BM6JP40, 19BM6JP59]*

Whenever we talk about training a neural network on a small dataset, there are always speculations and chances of underfitting. It can also overfit if trained even a bit longer. To tackle this problem, usually data-specific strong augmentation is suggested. These augmentations are generally fixed for a set of computer vision tasks, for example, cropping an image is preferred for Classification tasks while increasing lighting and contrast is recommended for image detection tasks. Along with it, stable training with respect to non-tunable hyperparameters (like regularization parameter or weight decay) is also desirable in many cases. Current approaches of handling this issue are *Mixup, SamplePairing* and *BC learning*.

One such recently proposed technique is *batchboost*. This method hypothesizes that mixing many samples rather than just two can significantly increase network's performance. Moreover, it takes on a non-linear mixing technique and proposes a way to create mini-batches of samples. At its core, it advocates about mixing hard-to-classify samples with easy-to-classify samples and hence provides a balance between the different convolution-based features. This method was proposed around January 2020 and was hence apt for investigation by us. In this project, we try to implement and investigate this novel method with respect to image classification tasks on a few standard datasets.

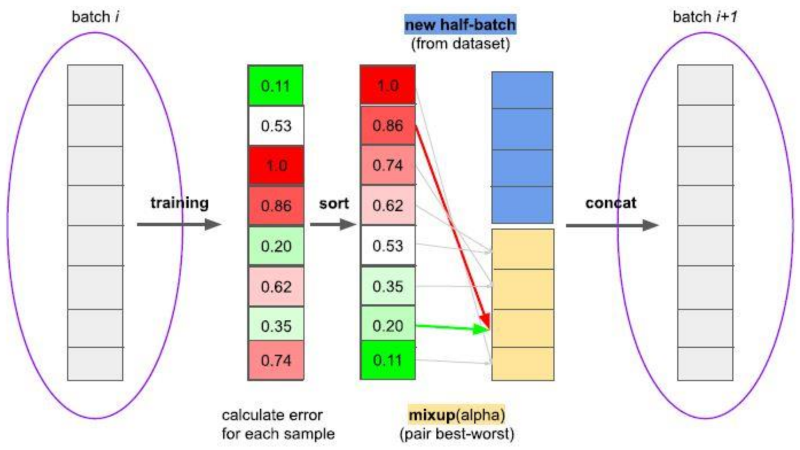

Fig: Explanation of batchboost algorithm

We first start by building vanilla neural network architectures from scratch and try different augmentation and regularization techniques to improve the architecture's performance. In this step, we try building different architectures similar to *AlexNet* and *VGG-16*. Next up, we implement *transfer learning* techniques without any data augmentation to gauze the improvement in performance of vanilla networks over pretrained network of *ImageNet* dataset. In our fourth step, we implement the *batchboost* technique *from scratch* for both the vanilla as well as the pretrained networks. We conclude this experiment by reporting our findings for the CIFAR10 dataset.

Tools and techniques used:

1. **Keras** implementation of vanilla classification architectures (similar to ***AlexNet*** and ***VGG-16***)
2. **Augmentation** of images for **Regularization** in Keras
3. **Transfer Learning** with **ImageNet** dataset's weights
4. **Pytorch** implementation of **batchboost** from scratch on vanilla and pretrained networks
5. Experimentation on the publicly available **CIFAR10** dataset