Web Based Image Data Classification: An Overview of MLP-Mixer and CIFAR10 Classification

Oğuzhan ERCAN
Yıldız Technical University
Computer Engineering Department
oguzhanercancs@gmail.com

***Abstract*— MLP-Mixer made a lot of noise when it was released to replace CNNs, but there were no new studies on it like VITs. In this work, I worked on CIFAR10 dataset. The paper is all about exploring MLP-Mixer from different perspectives experimentally .**

***Keywords*— MLP-Mixer, CNN, Vision Transformers, CIFAR10, SGD, ADAM, Data Augmentation**

I. Introduction

In the last few years, many alternatives to CNNs[1] have emerged in the field of computer vision. The most prominent model among these alternatives was actually the VITs[2], which evolved from transformers for NLP. VITs showed state of the art results. MLP-Mixer[3] has been introduced against the ever-complicating deep learning models as a competitor to the Transformers that come up with the attention title and the main solution of computer vision; CNNs . In the MLP-Mixer article, the authors revealed that the model had a higher accuracy than both CNNs and VITs. At the same time, this model has lower complexity in terms of time and space.

Although the CIFAR10[4] dataset is a dataset consisting of 10 classes and seems easy to classify, its image size is 32x32, and when we examine it according to other datasets, the following result emerges: the distribution of the samples is different from each other in the data distribution of a single class; that the distributions are similar in the samples between the classes. It is seen that this dataset does not reach high success results in deep learning applications. It is true that there are some models that do well in CIFAR10, but the hyperparameters of these models are tuned to perform well on this dataset. In this study, the results obtained when MLP-Mixer is trained on CIFAR10 dataset using various different methods were examined.

The best method I have chosen as a result of experimental studies gives a higher success rate compared to other studies. In the following sections, we will examine CIFAR10, future works and the effects of the following situations on the model success metric: patch size, learning rate and learning rate scheduler, batch size, optimizer, momentum.

II. Dataset: CIFAR10

A. Overview

CIFAR10 was collected by Alex Krishevsky and his team to train manufacturer models and was published in 2009. CIFAR10 has 10 classes and they are: airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. There are 1,000 images of each class.When the structure of 1000 samples is examined, it is insufficient. For this reason, I used various data augmentation methods during model training and continued the training with 5000 samples for each class.

B. Challenges

As I mentioned before, the small size of the samples in the data causes the features to be extracted to have low meaning. Compared to other datasets, the similarity rates are low even if the samples are from the same classes. The similarity between samples belonging to different classes is higher compared to other datasets. Using MLP-Mixer turns into a difficult task, especially for low resolution images. Because MLP-Mixer up-samples the samples due to its operation. Up-sampled results may contain unnecessary information due to little information in the source.

III. MLP-Mixer

MLP-Mixer is a competitive but conceptually and technically simple alternative, that does not use convolutions or self-attention. Instead, Mixer'sarchitecture is based entirely on multi-layer perceptrons (MLPs) that are repeatedly applied across either spatial locations or feature channels. Mixer relies only on basic matrix multiplication routines, changes to data layout (reshapes and transpositions), and scalar nonlinearities.

Figure 1 shows the architecture that proposed. It basically takes an image divided to patches as an input, linearly projects them to vectors, applies layer normalization. Then transposes the result. The transposed result goes to the linear layer. This allows the model to extract the features between patches. This operation corresponds to global attention in transformers. Then transposes again and applies layer normalization to the result and the result goes to the linear layer again. This allows the model to extract the features between channels. This

operation corresponds to self-attention in transformers. The operations mentioned above is called as Mixer layer. This process is continued with more than one mixer layer. Global average pooling is applied to the output. Output goes to linear layers.

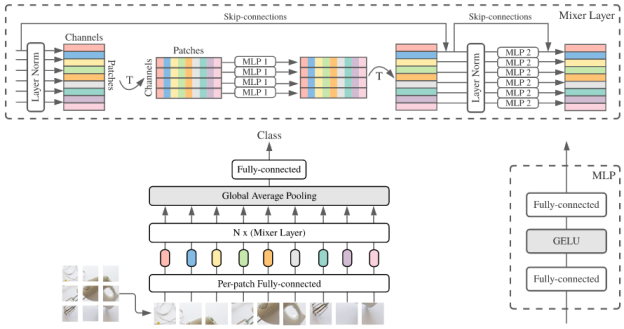

Figure 1

In the following sections, we will examine the effects of patch size, learning rate and learning rate scheduler, batch size, optimizer and momentum on model accuracy.

A. *Patch Size*

Patch size is one of the most important hyper parameters of MLP-Mixer. Patch size is suggested as 16 in the original paper, but the examples presented in the original paper are 224x224. The sample sizes for the data set used in this study are 32x32. If we use 16x16 patches, the number of patches we will get will be 4. In this case, the part of the model that works in the logic of global attention will work with 4 samples and will suffer from the low input values. In the experiments performed for patch size, it was determined that the most appropriate value is 8.

B. *Learning Rate and Learning Rate Scheduler*

Learning rate and learning rate scheduler affect model performance a lot. As will be examined in the following sections, although the graph of the error function is differentiable, it is found that the 2nd degree derivative is constantly changing. In this case, the use of learning rate scheduler negatively affects the success performance of the model. The learning rate value was chosen large compared to other studies. The reason for this is that the 2nd order derivative is constantly changing, and local minimums are frequently encountered.

C. *Optimizer and Momentum*

Adam was chosen as the optimizer. Momentum can be applied during optimization with Adam optimizer, allowing to prevent blockage at local minimums.

D. *Batch Size*

Experimental studies on Batch size have shown that 8 is the optimum value. The fact that the samples in the dataset were not large allowed the batch size to be selected large, but it did not create a big performance difference.

IV. Implementation

Pytorch library is used in implementation. Pytorch library has surpassed Tensorflow and Keras in academia in the last 2 years. Among the deep learning libraries, Pytorch, which is the most optimized deep learning library after Mx-Net and K-Net, is by far the most successful deep learning library compared to other deep learning libraries by bringing together the academy-product potential-Python environment.

The project has 2 different implementations, these implementations can be used for 3 different purposes. The first of these is used to train and test models using the console. The second use is model training, testing and analysis visualization with the interface designed on Jupyter Notebook. The third use is that it allows the written modules to be transferred to your own project as a package.

V. Results

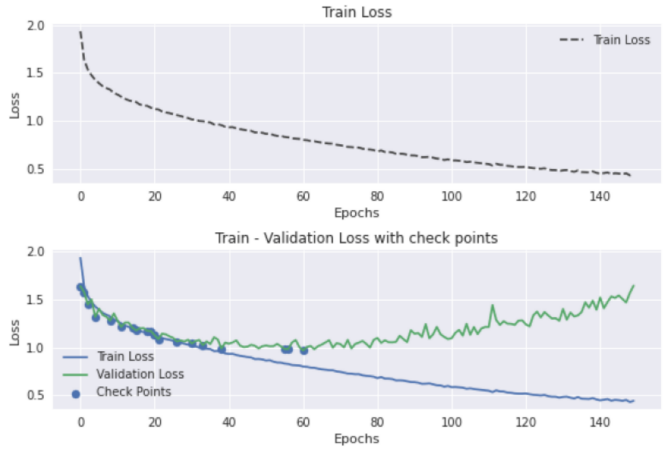

Figure 2

When we examine the graph created after the training process, the results are promising. When we examine Figure 2, we can see that the error continues to decrease in the train dataset while it increases after a certain point in the validation dataset. This situation exactly reflects the ideal error function graph taught in the books. Increasing the number of samples by 5 times with data augmentation methods and not taking into account the increase in the Kulback-Leibler divergence between the original data and the augmented data at this stage caused a change in the data distribution. Since the separation of the data is completely random, it is expected that the train and validation data will be similar to each other, but when the graph is examined, it is considered that the character of the validation loss does not resemble the character of the train

loss. Studies to be carried out on datasets that do not need data augmentation will reveal what caused this situation.

Acknowledgment

There are several implementations, you can reach them via https://github.com/Oguzhanercan/MLP-Mixer, , https://www.kaggle.com/oguzhanercan/mlp-mixer-cifar10-classification

References

[1] Yann Lecun, Patrick Haffner, Lean Bottou, Yoshua Bengio, *Object Recognition with Gradient-Based Learning*, 1999.

[2] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE, 2020

[3] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, Alexey Dosovitskiy, MLP-Mixer, An All Architecture for Vision, 2021

[4] Alex Krizhevsky, Learning Multiple Layers of Features from Tiny Images, 2009