DATA AUGMENTATION WITH FRACTIONAL LINEAR TRANSFORMATIONS

Dariya Mamyrbek. Department of Mathematical Sciences, Purdue University Fort Wayne.

1 INTRODUCTION

Machine learning is having large impact in most research fields and, in particular, deep learning and neural networks have shown a great results in the analysis of images. To achieve the best results, deep learning models have to be trained on large data sets. However, data is often costly [WP03] or cannot be collected for the populations of interest [BSK]. Neural networks tend to have poor generalization capability when trained on small samples [CIT]. There are several ways to handle this problem, including lack of data including building a simple model with fewer parameters, transfer learning, synthetic data, and data augmentation [SK19]. Data augmentation which is a set of techniques to artificially increase the amount of data by generating new data points from existing data [SK19], is a valuable tool to solve the problem of limited data. It has been used in many different context, including image classification, object detection, and instance segmentation [SK19].

Researchers have used many data augmentation techniques, including basic image manipulations such as geometric transformations, mixing images, kernel filters, etc. and deep learning approaches such as adversarial training, GAN data augmentation [SK19]. Nonlinear transformations such as conformal transformations [BSK] have proven to be useful to increase the sample size of training data and improve generalizbility of algorithms. In this work, we will explore bilinear or fractional linear transformation. This transformation can be considered as combinations of the transformations of translation, rotation, stretching and inversion, and how data augmetation based on these individual transformations can improve algorithms accuracy of a neural networks on benchmark datasets, such as CIFAR 10.

2 METHODS

2.1 DATA SOURCES AND SOFTWARE

The input images we will use in our data augmentation will be taken from the CIFAR-10 dataset. The CIFAR-10 is a collection of images collected by groups at MIT and NYU that are commonly used to train machine learning and computer vision algorithms [KH+09]. It is a popular dataset used for machine learning research which consists of 10 classes of images (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck) with 6000 images per class. We will gather a subset of the CIFAR-10 images and train the neural network to classify them. We will then apply Mobius transformations on the images to have a larger collection of photos. The remaining images from CIFAR-10 will be used to test the accuracy of the neural network. The analysis will be performed using Python and Tensorflow library. We are using PyTorch for neural network architecture.

2.2 CONFORMAL TRANSFORMATIONS

Among the most important affine transformations are the conformal transformations: translation, rotation, and scaling. Mobius transformations are a type of conformal mapping, too. A conformal mapping is a mapping that preserves local angles [Zho+21]. This means that any angle between a set of two intersecting lines of an image will remain the same after a conformal mapping has been applied to it. Mobius transformations also preserve the anharmonic ratio [Zho+21].The anharmonic ratio, also known as cross ratio has been used for identifying objects from different perspectives with a high accuracy [Fry00]. The preservation of this ratio will allow Mobius transormations of an image to represent the subject of the image from multiple perspectives.

Mobius transformations can be defined as:

$$f(z) = \frac{az+b}{cz+d} : ad - bc \neq 0$$

Where a, b, c, d are complex numbers. In fact, Mobius transformation is a combination of much simpler conformal transformations such as translations, rotations, and inversions in addition to many other types of translations. Translation can be defined in a way similar to the Mobius transformation, with corresponding parameters $a = 1, c = 0, d = 1$, resulting in the following equation:

$$f(z) = z + b$$

Similarly, scaling, can be defined as:

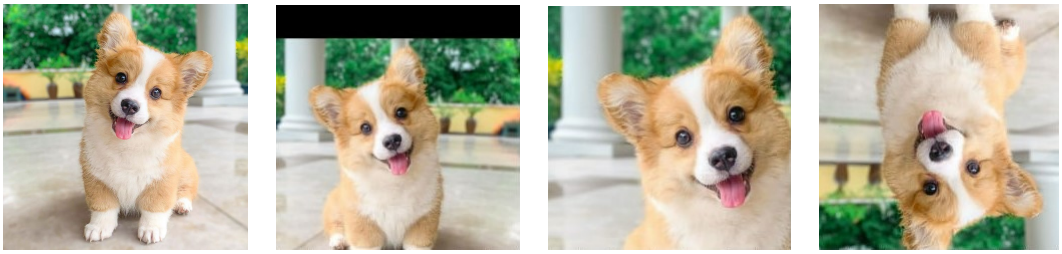
$$f(z) = az$$

For the image inverse, we will use defined Mobius parameters:

$$R(z) = \{0.1x, 0.5x, 0.9x\},$$
$$I(z) = \{0.5y, 0.8y, 0.5y\},$$
$$R(w) = \{x - 1, 0.5x, 1\},$$
$$I(z) = \{0.5y, 0.1y, 0.5y\},$$

where $R(p)$ and $I(p)$ denote the respective real and imaginary components of a point p, and height x and width y are dimensions of the original image.

Examples of these augmentations are shown below in Figure 1.



(a) Original image (b) Image translation (c) Image scaling (d) Image inverse

Figure 1: Examples of Augmentation

2.3 Class of M-admissible Möbius transformations

Mobius transforms reflect around the unit circle which allows for different distortions in scale [Zho+21]. This allows for different representations of the image at different scales which will aid in training the neural network. One issue that comes up is balancing the correct amount of distortion on an image. Too much distortion will result in the image not representing the object it is classified as. Too little distortion will result in the image serving as a duplicate of the original which is not helpful. This is addressed by adding the following constraint as done by Sharon Zhou et al. [Zho+21]:

$$\frac{1}{M} < |f'| < M : M > 1$$

Eventually, we can define a subclass of all Möbius transformations,

$$f(z) = \frac{az+b}{cz+d} : ad - bc \neq 0$$

called M-admissible Möbius transformations as long as the $f(z)$ fulfills the following inequalities by checking the points $0, p, pi, p(1+i), \frac{1}{2}p(1+i)$:

$$\frac{1}{M} < \frac{|a|^2}{|ad-bc|} < M$$

$$\frac{1}{M} < \frac{|a-pc|^2}{|ad-bc|} < M$$

$$\frac{1}{M} < \frac{|pc+ai|^2}{|ad-bc|} < M$$

$$\frac{1}{M} < \frac{|a-p(1+i)c|^2}{|ad-bc|} < M$$

$$\frac{1}{M} < \frac{|a-\frac{1}{2}p(1+i)c|^2}{|ad-bc|} < M$$

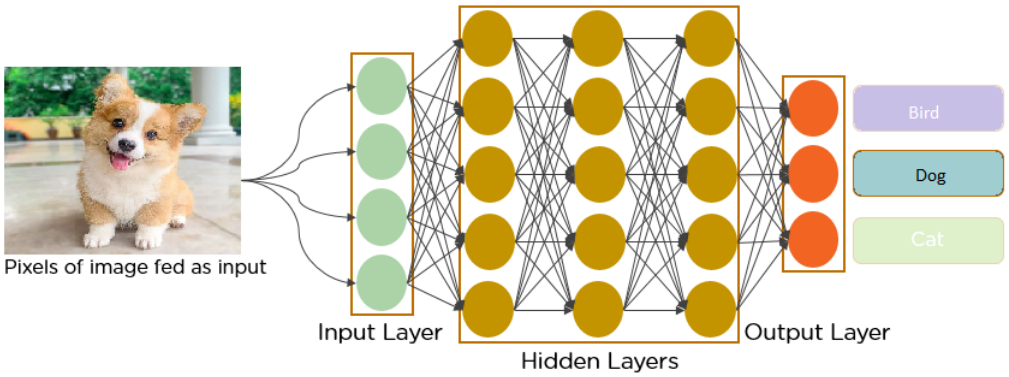
$$|\frac{\frac{1}{2}p(1+i)d-b}{a-\frac{1}{2}p(1+i)c}\frac{1}{2}p(1+i)| < \frac{p}{4}$$

2.4 Neural Networks

In this work, we will use a class of deep neural networks, convolutional neural network (CNN) to classify the images. The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area [Sah18]

Convolutional neural networks are distinguished from other neural networks by their superior performance with image, speech, or audio signal inputs. They have three main types of layers, which are: convolutional layer, pooling layer, fully-connected (FC) layer. The convolutional layer is the first layer of a convolutional network. While convolutional layers can be followed by additional convolutional layers or pooling layers, the fully-connected layer is the final layer. With each layer, the CNN increases in its complexity, identifying greater portions of the image. Earlier layers focus on simple features, such as colors and edges. As the image data progresses through the layers of the CNN, it starts to recognize larger elements or shapes of the object until it finally identifies the intended object [Edu20].

For the CNN architecture we used a sequential model which is appropriate for a plain stack of layers where each layer has exactly one input tensor and one output tensor. There will be three Conv2D layers, where as input, CNN takes tensors of shape (image height, image width, color channels) and creates a convolution kernel that is convolved with the layer input to produce a tensor of outputs. Since images are non-linear, to bring non-linearity, the relu activation function is applied after the convolutional operation. After initializing CNN, we apply a pooling layer, MaxPooling2D, used as an operation of down sampling of the image reducing the number of parameters to learn and the amount of computation performed in the network. Then we add two more convolutional layers. To complete the model, we will feed the last output tensor from the convolutional base into one or more Dense layers to perform classification. Dense layers take vectors as input (which are 1D), while the current output is a 3D tensor. First, we will flatten the 3D output to 1D, then add one or more Dense layers on top. CIFAR has 10 output classes, so we use a final Dense layer with 10 outputs.



(a)

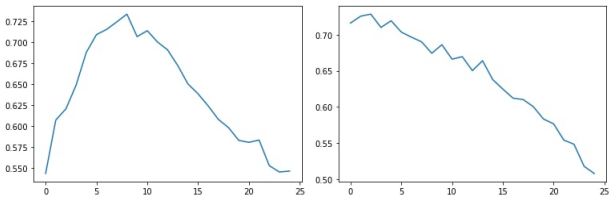
Figure 2: Convolution Neural Networks

2.5 Our analysis

In our analysis we used CIFAR-10 dataset with 10 classes with 10,000 images per class. There are 50000 training images and 10000 test images. First, we applied translation transformation with parameter b being in the range of $\{1, 2, .., 25\}$, in total having 25 CIFAR-10 augmented datasets, containing 100,000 augmented images in training dataset, and 25,000 images in the testing dataset. Similarly, we augmented CIFAR-10 using scaling transformation, changing the parameter a starting with 0.1 up to 2.5, increasing it by 0.1, having again 25 datasets of augmented images. For scaling, only one dataset was used with the defined Mobius parameters. We used a classification accuracy of the original CIFAR-10 trained using Convolutional Neural Networks model to compare it with the accuracy of our augmented datasets to examine how different data augmenting techniques affect the classification accuracy,

3 Results and Discussion

In this section, we discuss our results. We have trained a Convolutional Neural Network on the original CIFAR-10 and obtained an accuracy of 69.96%, and will use it as a control case. After that we obtained an augmented datasets for both translation and scaling, starting with small changes up to considerable changes in the picture as parameters were increasing. Plot of accuracy values for 25 different parameters of scaling and translation transformations can be seen in Figure 3. For translation, the highest accuracy obtained was 72.92% with a corresponding parameter $b = 3$. For scaling, it is 73.77% when a=0.9. It should be noted that the model trained on scaled augmented images, starts performing very poorly but shows an improving accuracy as we increase parameter b, and obtaining the best performance at $b = 3$, and then decreasing again. For translation, on opposite, it starts very good and then we see a downward trend, by the last augmented dataset obtaining the smallest accuracy. It can be concluded that in case of scaling, when the scaling parameter is too small or too big - there is very little of the augmented data, because when scaling is too small - it does not differ from the original image much, and hence, does not improve, similarly, when the parameter is too large and image scales up, it becomes difficult to recognize and hence classify the object. Regarding translation, we can see that wthe higher the coefficient of the transformation, the worse its performance gets. And for one augmented dataset using inverse images, we obtained an accuracy of 66%



(a) Scaling (b) Translation

Figure 3: Plots of accuracy values for 25 different parameters of scaling and translation

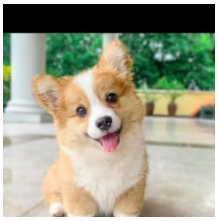
Method	CIFAR	Translation	Scaling	Inverse
Accuracy	69.96%	72.92%	73.77%	65.26%

Table 1: Accuracy values

4 Conclusion

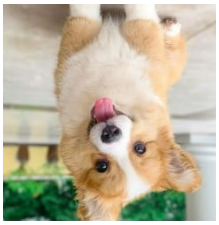
For translation and scaling having accuracy of 72.92% and 73.77%, respectively, the model performed better than the control case 69.96%. We can conclude that firstly, in case when there is a shortage of input images, it is always possible to use different augmentation techniques such as scaling, cropping, flipping, padding, translation and other, as these techniques help to generate a wide

(a) Original image

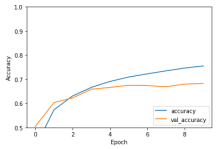


(b) Image translation

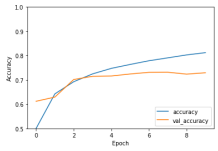
(c) Image scaling



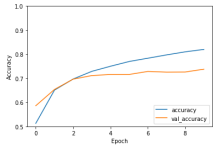
(d) Image inverse



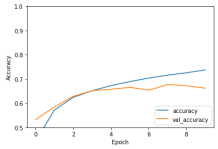
(e) Accuracy plot(a)



(f) Accuracy plot(b)



(g) Accuracy plot(c)



(h) Accuracy plot(d)

Figure 4: Examples of Augmentation with Accuracy plots

range of data. In terms of improving model's accuracy, we can see a small improvement in model's prediction using simple affine transformations such as scaling and translation. Inversion performed worse than the control case. We can assume that these transformations in combination with each other would give us higher accuracy values.

REFERENCES

[Fry00] John Fryer. "An object space technique independent of lens distortion for forensic videogrammetry". In: (Jan. 2000).

[WP03] Gary M. Weiss and Foster Provost. "Learning When Training Data are Costly: The Effect of Class Distribution on Tree Induction". In: *Journal of Artificial Intelligence Research* 19 (2003), pp. 315–354.

[KH+09] Alex Krizhevsky, Geoffrey Hinton, et al. "Learning multiple layers of features from tiny images". In: (2009).

[Sah18] Sumit Saha. "A Comprehensive Guide to Convolutional Neural Networks". In: *Towards Data Science* (2018).

[SK19] Connor Shorten and Taghi M. Khoshgoftaar. "A survey on Image Data Augmentation for Deep Learning". In: *Journal of Big Data* 6.60 (2019).

[Edu20] IBM Cloud Education. "Convolutional Neural Networks". In: *IBM Cloud Learn Hub* (2020).

[Zho+21] Sharon Zhou et al. "Data augmentation with Mobius transformations". In: *Machine Learning: Science and Technology* 2.2 (2021), p. 025016. DOI: 10.1088/2632-2153/abd615.

[BSK] Aayushi Bansal, Rewa Sharma, and Mamta Kathuria. "A Systematic Review on Data Scarcity Problem in Deep Learning: Solution and Applications". In: *ACM Computing Surveys* ().