EECS 453 Final Report – December 18th, 2024
Joseph Lynch (lynchjo, 11442632) – No partner

Optimal Methods of Data Augmentation for Image Classification Tasks

Abstract

Data augmentation is a popular technique for improving training efficacy for image segmentation and classification machine learning models. There is a wide range of augmentation methods, from geometric operations like flipping and rotating, to pixel based methods like modifying brightness and blurring. However, the general purpose of all data augmentation methods is to increase the size and diversity of training data[1]. This project sought to identify which data augmentation strategies might be most appropriate for image classification using the MNIST public dataset[2]. Ten augmentations were written: horizontal and vertical flip, rotate, perspective shift, crop, crop+paste, Gaussian blur, Gaussian noise, adjust sharpness, and adjust brightness. Seven combinations of these 10 image modifiers were applied to a dataset of 1000 MNIST samples. Next, a simple image classification model was trained over 25 epochs using the augmented datasets and the performance of the trained model was evaluated using a different 1000 MNIST samples. No data augmentation methods outperformed the control (unaugmented) dataset with simple geometric transformations(flipping and rotating) and simple pixel based methods (blurring, adding noise, etc) equaling the performance of the control dataset. These results indicate that data augmentation may not be impactful as originally believed for image classification tasks.

Motivation

The purpose of this project is to evaluate the effectiveness of data augmentation and to determine which, if any, strategies of data augmentation are most effective for image classification tasks. Data augmentation for machine learning has existed for years, but there is no consensus on which methods are best, just that augmentation as a whole is generally helpful. In addition to a seeming lack of concurrence among the field of machine learning, I have both academic and research reasons for being interested in data augmentation. Academically, writing my own augmentations presents as a fun challenge and helps me to better understand how they can impact training data. Concerning research, I am developing image segmentation models for use in the University of Michigan's Histotripsy laboratory. Knowing which augmentation strategies are most effective would be helpful to my work.

Related Work

As mentioned above, prior work does exist touting the effectiveness of data augmentation. Maharana, in their review of data pre-processing and augmentation techniques finds that, for deep learning models, data augmentation prevents overfitting[3]. Specifically, Maharana names geometric transformations as well as noise injection as effective, but they do not discuss which combinations of these methods are optimal. Mikołajczyk corroborates the conclusions of Maharana regarding geometric transformations but also includes pixel based methods such as adjusting contrast, sharpness, and brightness as having merit[4]. However, like Maharana,

Mikołajczyk does not make mention of an optimal combination or order of data augmentations to achieve the best training results.

Methods

In this report, the data augmentation functions were written, the model was made, and training the model using the augmentation functions was executed. The methods followed for each of these discrete steps are discussed separately below.

Data augmentation functions are widely available through many machine learning python libraries such as PyTorch. While the PyTorch library was used to write the model trained for this report, as will be described below, the library's data augmentation functions were not leveraged. Instead, all augmentations were written from scratch using only the NumPy and Random python libraries. This exercise was educational and was undertaken to gain a better understanding of the augmentations themselves and how they interact with the raw training data. The aforementioned 10 data augmentations were written as methods of a "Compose" class with which a user can apply any combination of the 10 augmentations in any order to an input datum (Figure 1).

```python
import numpy as np
import random as rand

class Compose():
    def __init__(self, matrix, **kwargs):
        self.result = matrix
        available_funcs = ['flipHorizontal', 'flipVertical', 'rotate',
                           'perspectiveShift', 'cropResize', 'cropPaste',
                           'gaussianBlur', 'gaussianNoise', 'adjustSharpness',
                           'adjustBrightness']

        for key in kwargs:
            if key not in available_funcs:
                print(f'{key} is not an available function. Exiting')
                break
            params = kwargs[key]
            function = getattr(self, key)
            if callable(function):
                output = function(self.result, params)
                self.result = output

    def flipHorizontal(self, matrix, params): …

    def flipVertical(self, matrix, params): …

    def rotate(self, matrix, params): …

    def perspectiveShift(self, matrix, params): …

    def cropResize(self, matrix, params): …

    def cropPaste(self, matrix, params): …

    def gaussianBlur(self, matrix, params): …

    def gaussianNoise(self, matrix, params): …

    def adjustSharpness(self, matrix, params): …

    def adjustBrightness(self, matrix, params): …
```

Figure 1: Compose Class Structure

Next, a simple model was written which could be trained to classify samples from the MNIST dataset. As indicated above, the PyTorch library was used to write the model which consisted of three layers – a convolutional layer and two linear layers – separated by the popular ReLu activation function (Figure 2). This simplistic model structure was chosen due to computational resource constraints.

```
def forward(self, x):
    x = self.conv1(x)
    x = F.relu(x)
    x = x.flatten(start_dim=1)

    x = self.linear1(x)
    x = F.relu(x)

    logits = self.linear2(x)
    return logits
```

Figure 2: Model Architecture

Finally, to generate results, the model was trained using 1000 samples of the MNIST dataset (90/10 train/validation). The compositions of data augmentations applied to the 1000 samples are listed in Table 1 with the exception of the control test in which no augmentations were applied.

Composition	Augmentations Applied
Geometric 1	Vertical flip, horizontal flip, rotate
Geometric 2	Perspective shift, crop+resize, crop+paste
Pixel 1	Gaussian noise, adjust brightness, Gaussian blur
Pixel 2	Adjust sharpness, adjust brightness, Gaussian Noise
Mixed 1	Vertical flip, horizontal flip, Gaussian Noise, crop+resize
Mixed 2	Rotate, perspective shift, adjust sharpness, adjust brightness
Mixed 3	Gaussian blur, rotate, crop+resize

Table 1: Augmentation Compositions

The model was trained for 25 epochs using each of the above compositions (as well as a control) and the model's loss was recorded to assess the performance of the augmentations applied. Additionally, after completing training, the model was shown an additional test dataset containing a different 1000 MNIST samples. The model's loss was also recorded on the test set and used as the primary indicator of model's performance.

Results

After training the model on geometrically augmented data, the best performance achieved was with the Geometric 1 composition consisting of vertical and horizontal flips and random rotations. The minimum test loss was 0.59: within 1% of the test loss of the control set. The Geometric 2 composition performed significantly worse with a loss greater than 2 that increased

with further training. Figure 3 shows the loss versus training epoch for the geometric compositions.

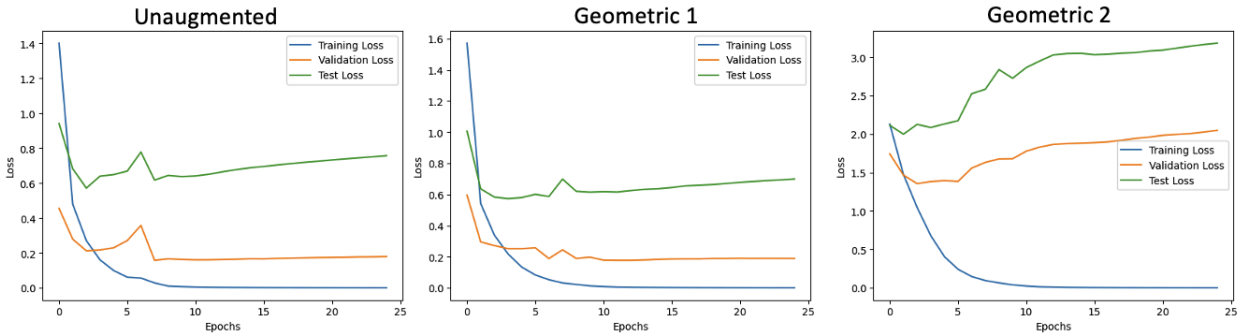

Figure 3: Geometric Composition Results

Pixel based compositions achieved more consistent performance with both compositions achieving training losses comparable to the control (0.59 and 0.58 for Pixel 1 and 2, respectively). The loss versus training epoch data for the pixel compositions are shown below in Figure 4.

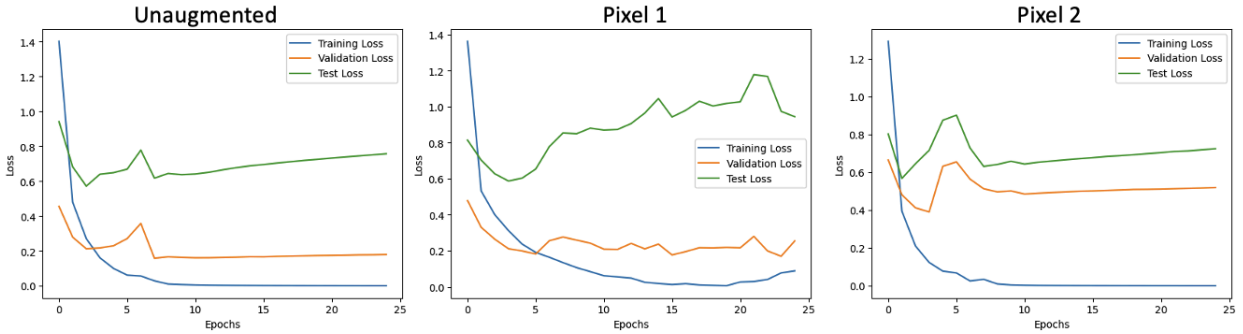

Figure 4: Pixel Composition Results

Combinations of the pixel and geometric compositions performed far worse than the two types did individually. The mixed compositions consistently achieved high test loss with Mixed 1 and Mixed 3 even having higher validation loss than test loss. The mixed composition performances are reported below in Figure 5.

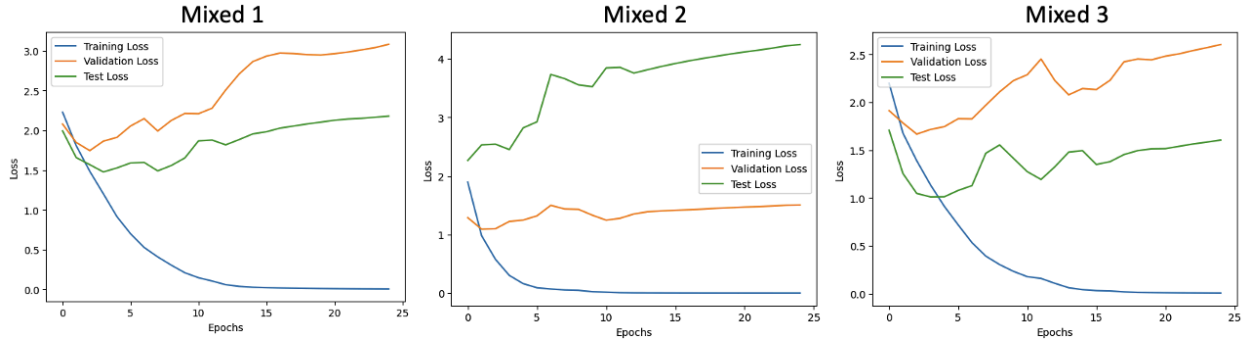

Figure 5: Mixed Composition Results

Composition	Min Test Loss	Epoch
Geometric 1	0.59	3
Geometric 2	2.03	2
Pixel 1	0.59	3
Pixel 2	0.58	2
Mixed 1	1.50	3
Mixed 2	2.31	1
Mixed 3	1.03	4
Control	0.58	3

Table 2: Summarized Results

Conclusion and Discussion

Results indicate that data augmentations may not be suited for the classification of MNIST samples. The lowest test loss achieved only matched the performance of the unaugmented control dataset. However, were augmentations to be employed, results indicate that using pixel based methods (blurring, noise injection, or adjusting sharpness) and only simple geometric methods (flipping and rotating) will produce the best results. Mixing of pixel based and geometric augmentation methods may lead to worse performance.

These results are likely not general to all image classification tasks. Samples from the MNIST dataset are relatively small (28 by 28 pixels), so even small alterations can drastically change the appearance of a sample. This fact is likely why cropping methods performed so poorly whenever used, as the sample was too greatly distorted for the model to properly classify it. This could also explain the increasing test loss over time observed for some compositions as well as the higher than expected validation loss in these compositions. To further broaden the discussion, data augmentations may be best suited for tasks where the shape of an object class is not its most important feature (as it is with the numerals in the MNIST dataset). For example, if a model is trained to evaluate skin lesions for melanoma, the “texture” of the image may be more instructive than the shape of the lesion. In these instances, data augmentations may produce a greater benefit as distorting the shape of the training samples encourages the model to rely on other features to make its classification.

Finally, this study had a number of limitations. As mentioned above, the classification of MNIST samples may not be the optimal arena in which to evaluate the merits of data augmentations. Additionally, creating the augmentation methods from scratch introduces a significant potential for human error. Another potential source of error is the model used. For each composition tested, the minimum test loss was achieved in the first 5 epochs after which the loss would either increase or barely change. Increasing loss was likely a function of the augmentations being poorly suited to this classification task, but the cases where the model’s performance

stalled may be a function of the model being too shallow. As mentioned, the model used was simplistic due to computational resource limitations. A deeper model would generate different results that could alter conclusions on the most effective augmentation methods.

Bibliography

1. Shorten, C., Khoshgoftaar, T.M. A survey on Image Data Augmentation for Deep Learning. J Big Data 6, 60 (2019). https://doi.org/10.1186/s40537-019-0197-0
2. Deng, L. (2012). The mnist database of handwritten digit images for machine learning research. IEEE Signal Processing Magazine, 29(6), 141–142.
3. Kiran Maharana, Surajit Mondal, Bhushankumar Nemade, A review: Data pre-processing and data augmentation techniques, Global Transitions Proceedings, Volume 3, Issue 1, 2022, Pages 91-99, ISSN 2666-285X, https://doi.org/10.1016/j.gltp.2022.04.020.
4. A. Mikołajczyk and M. Grochowski, "Data augmentation for improving deep learning in image classification problem," 2018 International Interdisciplinary PhD Workshop (IIPhDW), Świnoujście, Poland, 2018, pp. 117-122, doi: 10.1109/IIPHDW.2018.8388338. keywords: {Image color analysis;Machine learning;Lesions;Image classification;Neural networks;Cancer;Task analysis;Machine learning;style transfer;data augmentation;deep learning;medical imaging},