Convolutional Neural Network Parameter Effects

on validation accuracy using MNIST-Fashion Dataset

Milad Barai

Introduction

In this project, an analysis is made of how different parameters affect convolutional neural network accuracy.

The baseline neural network for parameter testing is:

Input Block:

Layer 1: Convolution 2D, 128 filters, 3x3 kernel size, padded

Convolution Block 1:

Layer 2: Convolution 2D, 128 filters, 3x3 kernel size, padded

Layer 3: Convolution 2D, 128 filters, 3x3 kernel size, padded

Layer 4: Max pooling 2D, with 2x2 pooling size

Layer 5: Dropout, rate 0.25

Convolution Block 2:

Layer 6: Convolution 2D, 128 filters, 3x3 kernel size, padded

Layer 7: Convolution 2D, 128 filters, 3x3 kernel size, padded

Layer 8: Max pooling 2D, with 2x2 pooling size

Layer 9: Dropout, rate 0.25

Output/Classification Block:

Layer 10: Flatten

Layer 11: Dense, 512 neurons

Layer 12: Dense, 10 neurons, softmax

Only one parameter is changed at a time to encapsulate the accuracy changes of the specific parameter. The neural networks are evaluated and trained using 10-fold cross-validation spanning 50 epochs and a batch size of 128 with a checkpoint for storing the result of the best model. Parameters tested are No Dropout, Increased Dropout, Increased Kernel, No Padding and Data augmentations.

Additionally in this project, a visualization of the feature maps for each layer of the base model is performed.

An example of transfer learning is also provided using the VGG19 model with pre-trained weights from the “ImageNet” competition. The pre-trained layers are frozen and the classification layer of the model has been switched to the same classification block as the baseline model.

The project has 3 python files each responsible for some basic functionality.

> *trainAndEvalute*, Fetches and preprocesses the MNIST-fashion dataset into a structure fit for training the model. It also creates, trains, evaluates and stores the models as well as their training progress.
>
> *evaluteCsvResults*, Fetches and calculates the mean and standard deviation of the different models given the folder and output structure of the *trainAndEvalute*
>
> *featureMapVisualization*, restores a model preferably save by the *trainAndEvalute* then restructures the outputs so that a feature map can be extracted from doing model predictions.

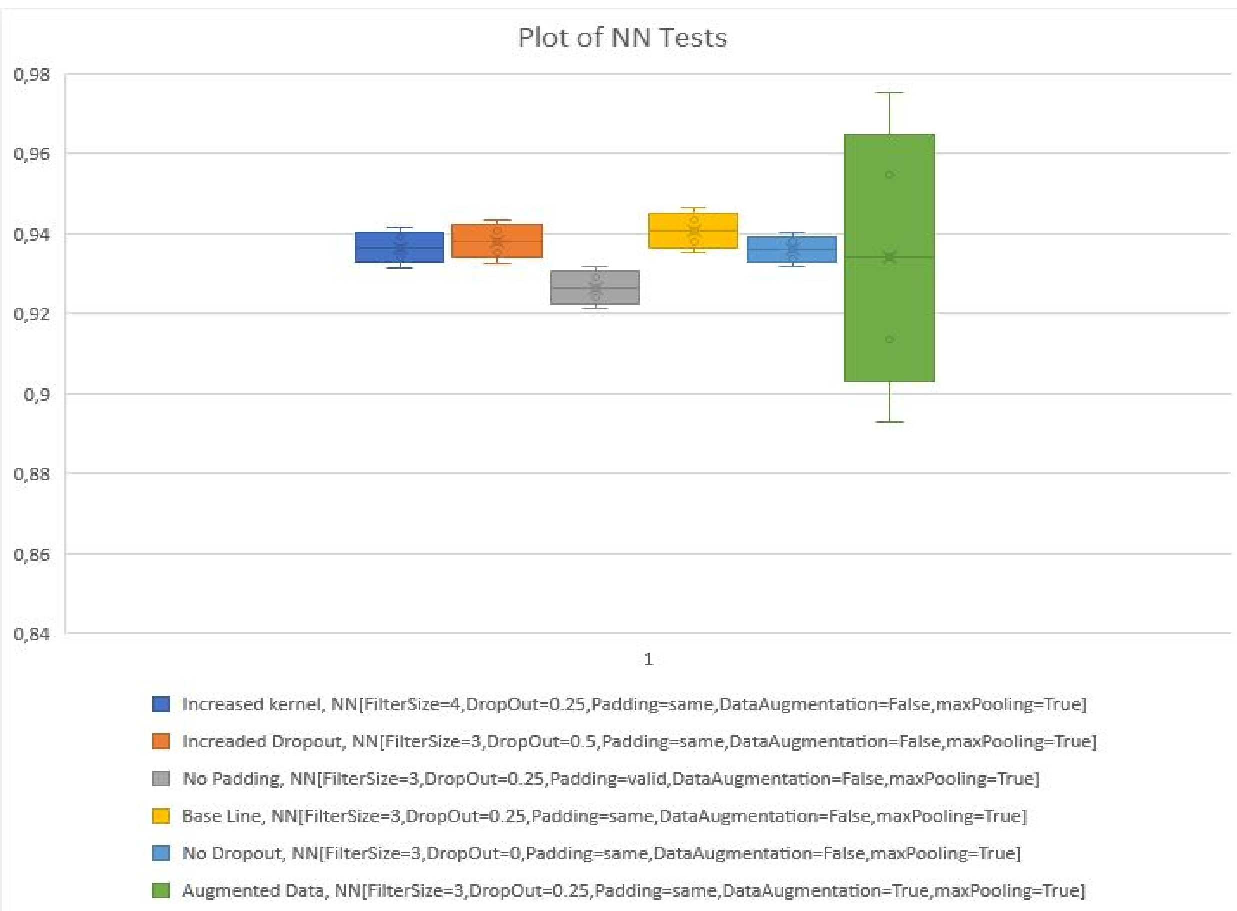
Result

The result of the 10-fold cross-validation:

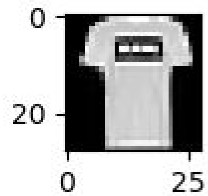
Baseline,	$\mu = 0.940814285$, $\sigma = 0.002792884$
Increased Dropout,	$\mu = 0.938128569$, $\sigma = 0.002708595$
Increased Kernel,	$\mu = 0.936557146$, $\sigma = 0.002468962$
No Dropout,	$\mu = 0.935985709$, $\sigma = 0.002116452$
Augmented Data,	$\mu = 0.933987500$, $\sigma = 0.020568138$
No padding,	$\mu = 0.926542854$, $\sigma = 0.002661746$
VGG19,	$\mu = 0.398095239$, $\sigma = 0.006958581$

The baseline turned out to be the neural network with the highest mean validation accuracy.

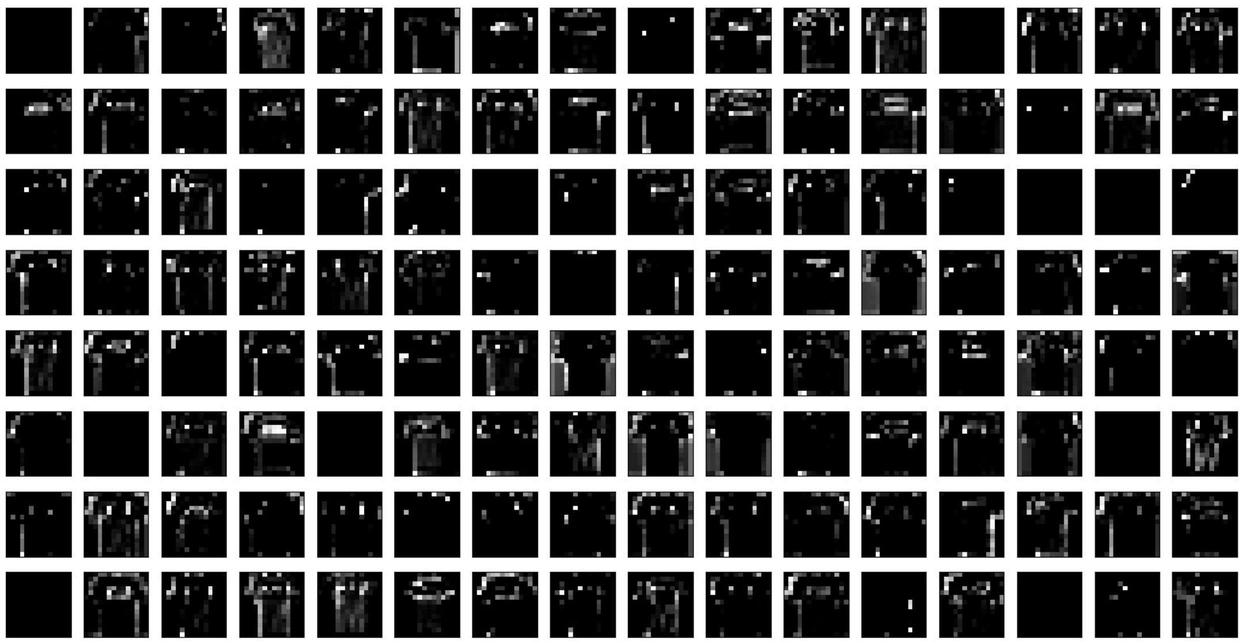
Plot representations of the data above:



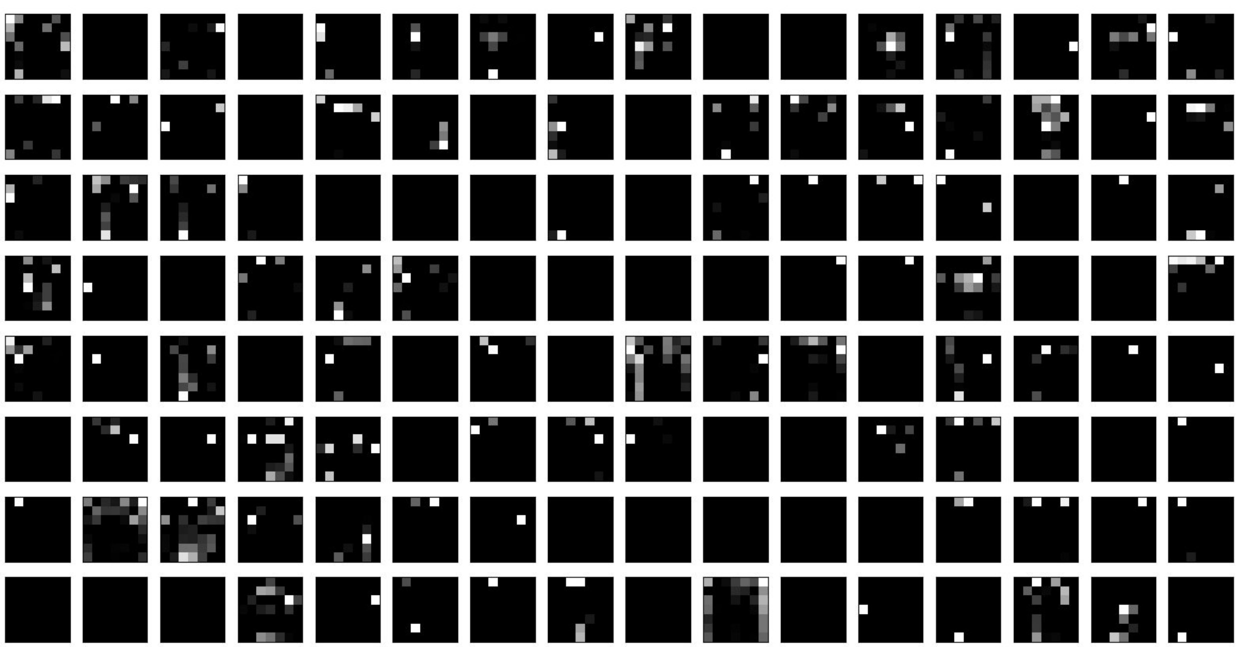
Feature Map visualization of a t-shirt



Convolutional Block 1 feature maps:



Convolutional Block 2 feature maps:



Discussion

The baseline model resulted in the highest mean validation accuracy.

No dropout vs Baseline:

Compared to the baseline no dropout resulted in the model being overfitted as the epochs continued onwards. The last epochs resulted in a training accuracy of 98% however, only a validation accuracy of 83%.

This might also be the cause of the mean of the best scoring models is below that of the baseline.

Increased dropout vs Baseline:

The mean between the increased dropout and the baseline model differs by approximately 0.2%. This might have been due to the different layers not being able to train properly as the dropout rate of neurons in the previous layer has been too high.

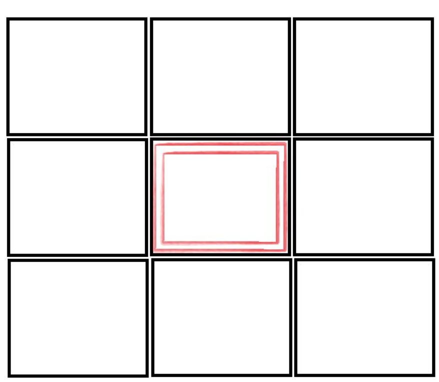
No padding vs Baseline:

Due to no padding information was lost along with the training of the model resulting in lower output shapes as the data was flowing through the convolutional layers. This might have resulted in a loss of data and in turn a lower mean than baseline

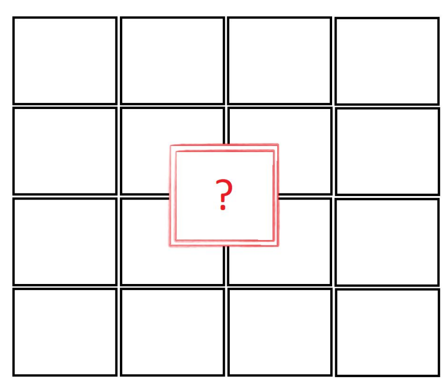
Increased Kernel vs Baseline:

The increase of Kernel size resulted in a decrease of mean by approximately 0.04%. This might have been a result of selecting an even-sized filter as the intention of convolution is to encode data surrounding a pixel. Since 4x4 has no middle pixel the encoding of the data within the 4x4 filter will be for a nonexisting pixel resulting in a loss of accuracy.

3x3



4x4



Augmented Data vs Baseline:

Augmenting data and using it for the baseline gave a quite large standard deviation as it augmented randomly. The mean ended up bellow the baseline and this might be due to the fact that the augmentation provided data which was not present in the validation set. Although small augmentations were selected such as horizontal flip and zoom this might still have resulted in a loss overall as the training data no longer reflected the validation data.

VGG19 vs Baseline:

VGG19 got a horribly low accuracy rate and it might have been since it's been trained for general classification with larger images in 3 channels rather than smaller images for clothes.