Classification of Standard Fashion MNIST Dataset using Convolutional Neural Network

JULIA UMMA REDDY   
Computer Science, CalState University- EastBay  
Hayward, USA  
[jummareddy@horizon.csueastbasy.edu](mailto:jummareddy@horizon.csueastbasy.edu)

***Abstract*—Machine Learning with the use of Convolutional Neural Networks has taken a major role in solving problems like fraud detection, predictive analysis and recommendation systems. Most widely implemented in classification of data mainly in the fashion industry. Fashion industry wheels on online shopping and garment recommendation systems. The purpose of this paper is accurate classification of garment images in the standard MNIST Dataset using the CNN(Convolutional Neural Network) model discussed in [1]. The proposed model is based on the model in [1] and attempts to compare the accuracy and loss results with the base model. The main aim of both the models in [1] and in this paper is to address model overfit issues by implementing BatchNormalization and Dropout layer in the CNN model.**

***Keywords—Convolutional Neural Network(CNN), Batch Normalization, Dropout Layer***

# Introduction

In recent years, machine learning and deep learning have been profoundly used in many applications like Natural Language processing in the medical field, Object and Anomaly detection in fraud detection, etc.[2][3][4]. The most commonly used architecture of deep learning is Convolutional Neural Networks (CNN). This architecture[5] has seven main parts: the first is the Input Layer. This is the collection of all the pixels from input data like an image. The second part is the Convolutional Layer in which a filter scans through the input image and extracts features like edges, corners, textures and patterns.The third part is the Activation Layer (Rectified Linear Unit) where the activation function is applied to the results from the convolution. Here nonlinearity is added to the system to help learn complex patterns. The fourth part is the Pooling Layer, this is where spatial dimensions of the image are reduced. The most commonly used pooling is Max-pooling where the pixels with maximum value are retained from a group of pixels. Similar to filters but with no parameters. The fifth part is the Flattening. Here the results from pooling are flattened into a list of numbers. Then comes the sixth part called Fully Connected Layer. Here the flattened data is connected to all the neurons in the networks. The next part is the seventh part, the Output Layer. Here the output of the entire CNN is displayed, that is the classification class of the input image . During the training process, the network back propagates to adjust the weights and parameters to get the classification class predicted accurately.

The current widely used application of machine learning and CNN is image classification and object recognition[6]. This application comes with a lot of challenges. One of them is prediction being affected by scale variations, picture brightness and positional variations in the images.

The main aim of this paper is to address the overfit issue that is faced while classifying data having garment images of multiple classes. CNN model from [1] is used and the training and test results are compared with the model proposed in this paper. [1] includes Batch Normalization and Dropout Layers to deal with overfitting. The model proposed in this paper includes both Batch Normalization and Dropout Layer and recorded results with four changes made to the model:(i) Kernel size changed to (3,3) for all convolutions, (ii) Activation function to LeakyReLU from ReLU, (iii) Data Augmentation performed and (iv) Adding VGGNet-Like architecture to the model.

The database used is the standard FASHION-MNIST The remaining paper is discussed in the following order: Section 2 gives a summary of related work in this domain of CNN. Section 3 explains the proposed model and the methods. Section 4 shows classification results and Section 5 gives a conclusion of this paper as well gives future work

# Related Work

Deep Neural network and Convolutional Neural Network have become core methods in various applications such as medical imaging to help detect tumors, robotics and image classification and natural language processing[2][3][7].

There have been different CNN models that have been completely evaluated for image classification as AlexNet, VGGNet, GoogleNet, ResNet and LeNet[8][9]. Other models for image classification that showed significant results in accuracy are Short-Term Memory Model[18] and Residual Skip Network models and Support Vector Machine where feature detection is done using a hyperplane that separates different features classes the best.

However the model in [1] concentrates on tackling the overfitting issue that is most common when there are many classes for the images to be classified into. [1] includes Batch Normalization’ and ‘Dropout Layers’ to deal with the issue. The training and test results have had high accuracy The proposed model in this paper will use the same model from [1] but also make changes to model like Kernel size, activation function from ReLu to LeakyReLU, Data Augmentation performed and including VGGNet-Like architecture.

# Methodology

In this paper, the results are recorded for five models. The first one is from the model in [1]. The second through fifth will be the proposed models in this paper.

The dataset used is the standard FASHION-MNIST. It has 60,000 images in gray-scale each of size aBefore you begin to format your paper, first write and save the co28X28. The main challenge in image classification of fashion dataset is fashion division. It arises because of multiple clothing categories that are large in number and depth. The CNN models mentioned in the previous section make accurate predictions for such dataset but do not deal with the problem of overfitting. Below is the image (Fig.1) from FASHION-MNIST showing the various categories of the clothing.

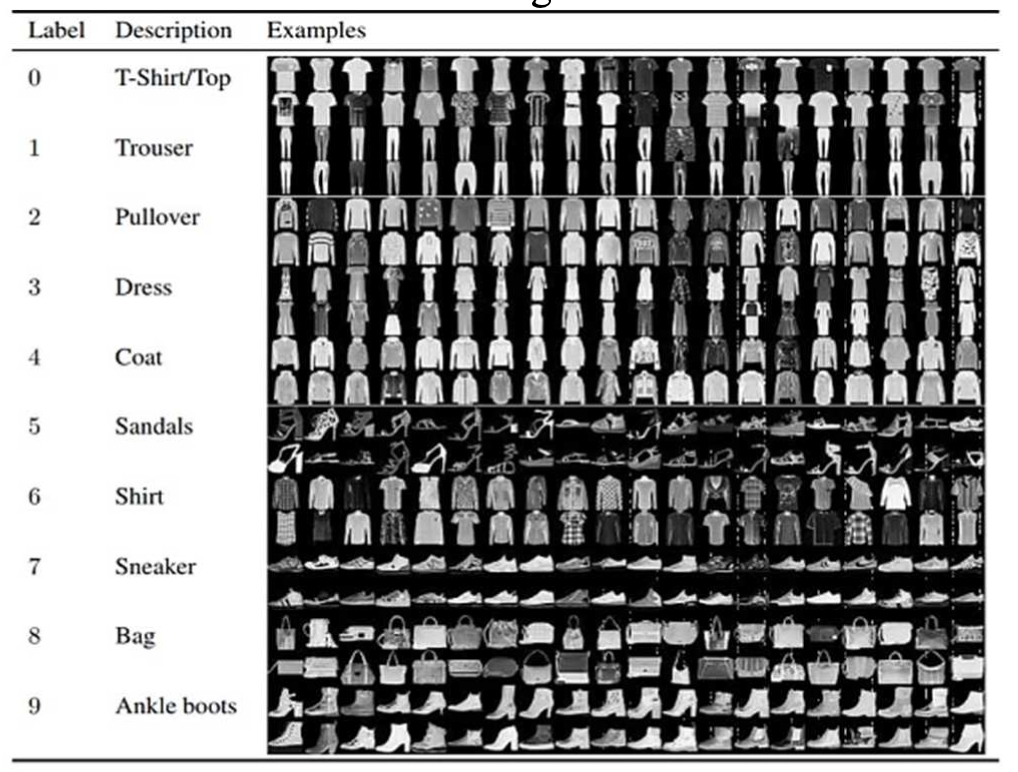


Fig 1: Fashion-MNIST Dataset (image taken from [1])

The model[1] used below mentioned for processing the data classification:

1. Kernels: The kernels have been used as they are independent for each of the layers and calculations can be done quickly using Graphical Processing Unit(GPU). The function for convolution in a 2D image with 2D kernel is,

(1)

1. Activation Function: Rectified Linear Unit(ReLU) is the activation function used as a nonlinear function.

(2)

1. Feature Detection Block: Here Conv2D layer is used.The input size is (28,28,1)

After the first Conv2D layers, Batch Normalization is applied

Then the activation function ReLU is used. Following the same - Conv2D, Batch Normalization and Activation function - three times, that is three convolutional layers.

Then Max-pooling is applied with pool size (2,2)

1. Transition Block: Here the output of the feature detection is converted to an array of numbers. The process is called Flattening. The input for the Flattening function is (14,14).
2. Classification Block: Here the deep neural network, Dense layer is implemented. Followed by Dropout Layer. This Dense Layer - Dropout is done three times where the size of the Dense layer is 208, 160 and 128 respectively.

Below is the model in [1] shown (Fig.2):

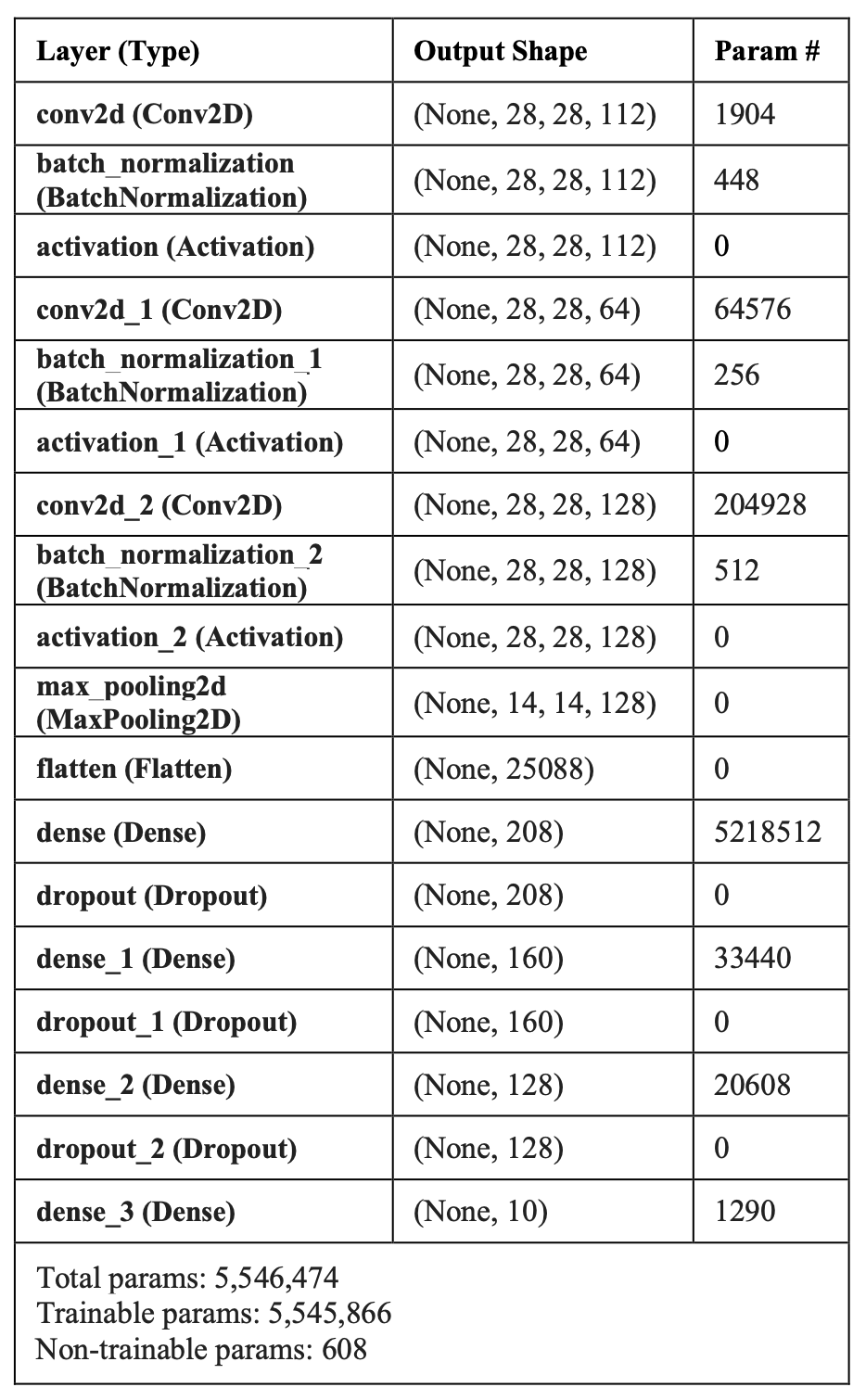


Figure 2: Model from [1] (image is taken from [1])

The Results: 20 epochs are applied with batch size 1000 on the validation/test data.

1. The value of loss started from 0.6749 and ended at 0.0652,
2. The accuracy value started from 0.7637 and finished at 0.9786.
3. Validation/testing loss started from 0.4089 to 0.3195
4. Validation/testing accuracy started from 0.8554 to 0.9260

Batch Normalization: The batch normalization[] is a technique used to normalize input of each convolutional layer into mini-batches. This is done by subtracting the mean and dividing by the standard deviation of the mini-batch.

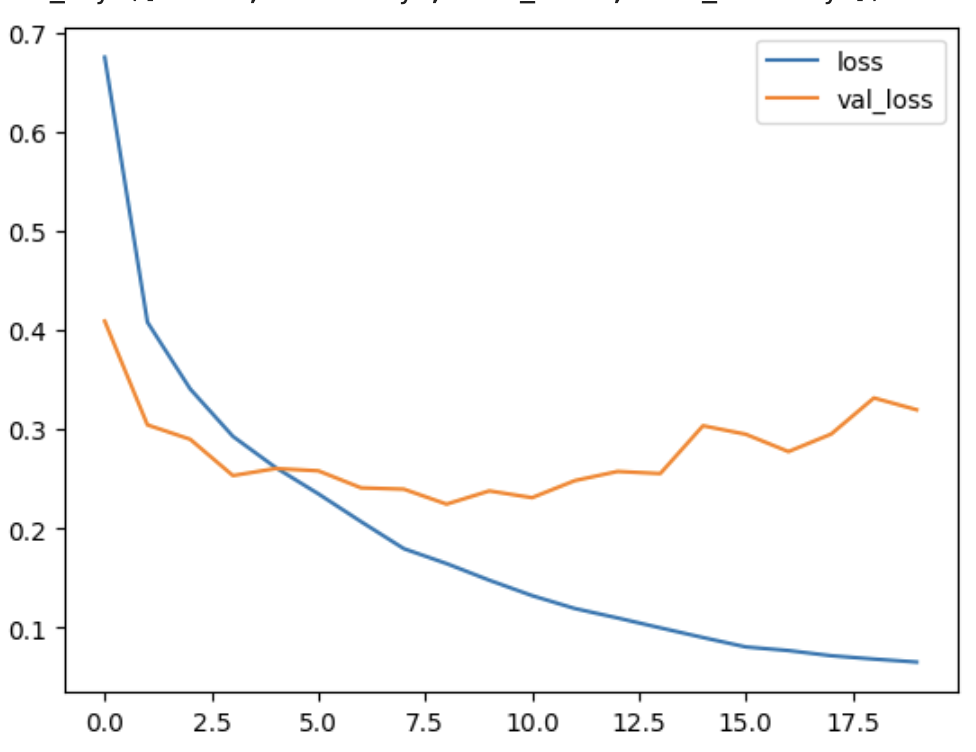
The normalized values are then scaled and shifted using parameters(learnable) called gamma and beta.

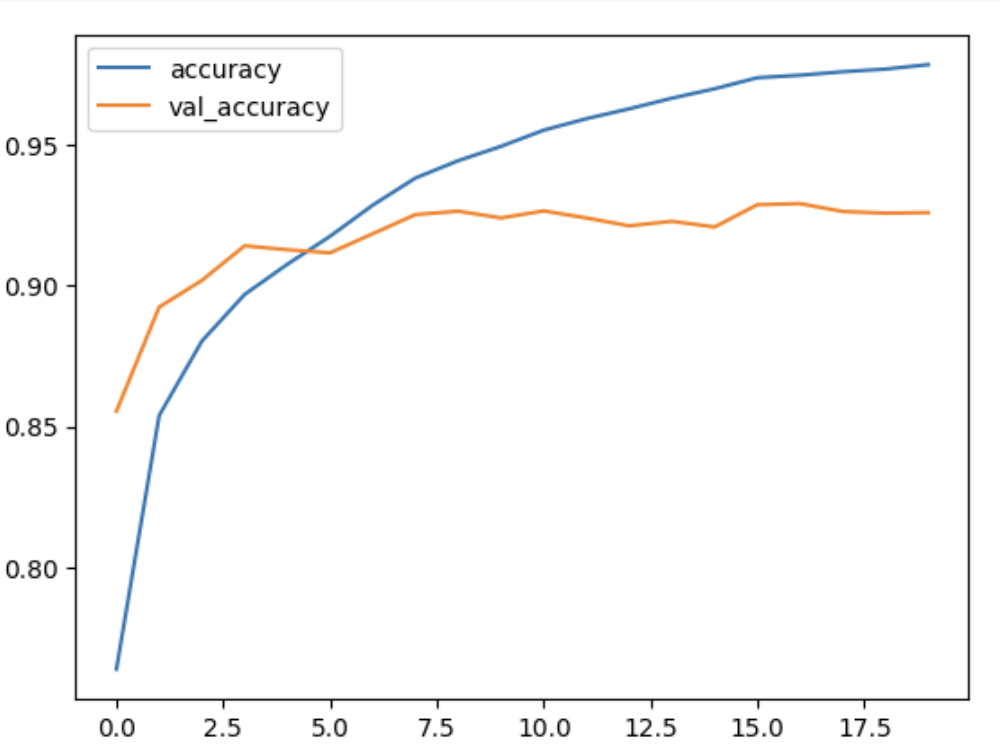
(3)

where x is the input to a layer in mini-batch, is the mean and is the standard deviation of the mini-batch, and are the learnable parameters.

Batch normalization is mainly used to help the input not shift too much towards any pixel information that is non-uniformly high. And also to speed up the training process.

Dropout Layer: This is another way to regularize[] the input to the neural network. This is mainly used to help with the overfitting issue by randomly dropping (releasing) neurons during the training process. Thus it helps the network from relying too much on those neurons that increase the capacity of the model.

The loss and accuracy graphs (refer Fig 3 and Fig 4) for model 1 show that there is no overfitting. Figure 3: Model1 Loss graph

 Figure 4: Model1 Accuracy graph

The loss and accuracy for training and validation are recorded in Table 1.

The model proposed in this paper follows the same model as mentioned above but with changes as listed below.

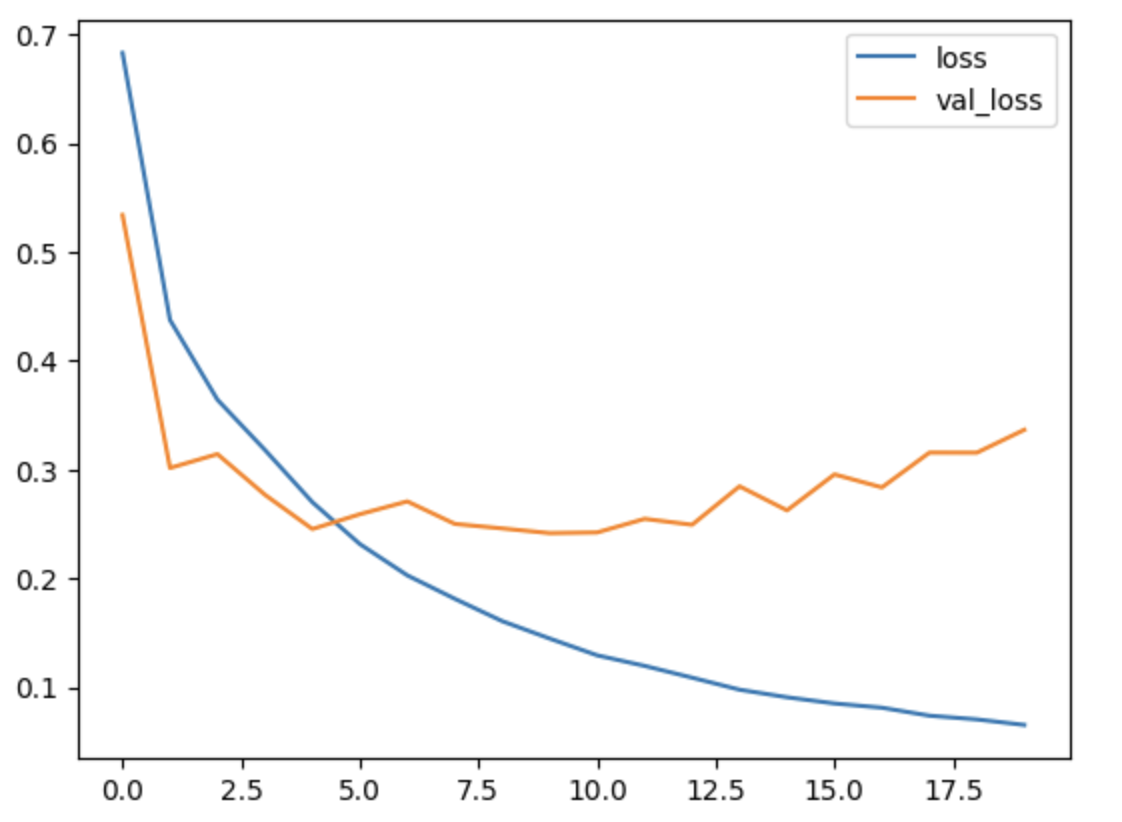
1. Kernel size is changed to 3X3 for all the three Convolutions.
2. The Activation Function is changed to LeakyReLU.
3. Data augmentation by shifting the image.
4. Using VGGNet-Like architecture in the proposed model

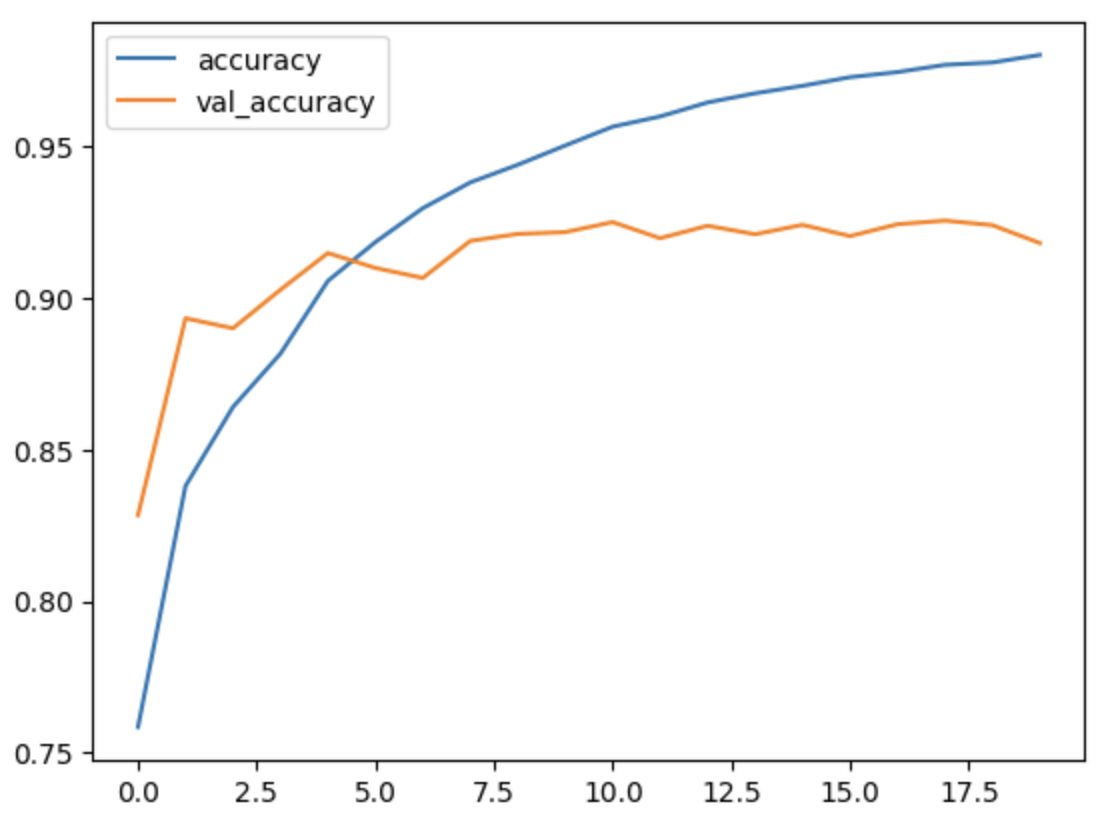
Description of the changes:

Change-1: Changing kernel size leads to change in learnable parameters. Hence the kernel size was reduced (3,3) for all the convolutions to test the training time of the model. With kernel size (3,3), the number of parameters was reduced by 131,856. The proposed model was trained and tested with 20 epochs. Training time remained the same as Model1. The loss and accuracy results are:

1. The value of loss started from 0.6260 and ended at 0.0688,
2. The accuracy value started from 0.7920 and finished at 0.9765.
3. Validation/testing loss started from 0.3635 to 0.3525
4. Validation/testing accuracy started from 0.8701 to 0.9295

Please refer Fig 5 and Fig 6 for loss and accuracy graphs and refer Table 1 for tabular presentation of testing results of change 1.

 Figure 5: Change 1 Loss graph - No overfitting

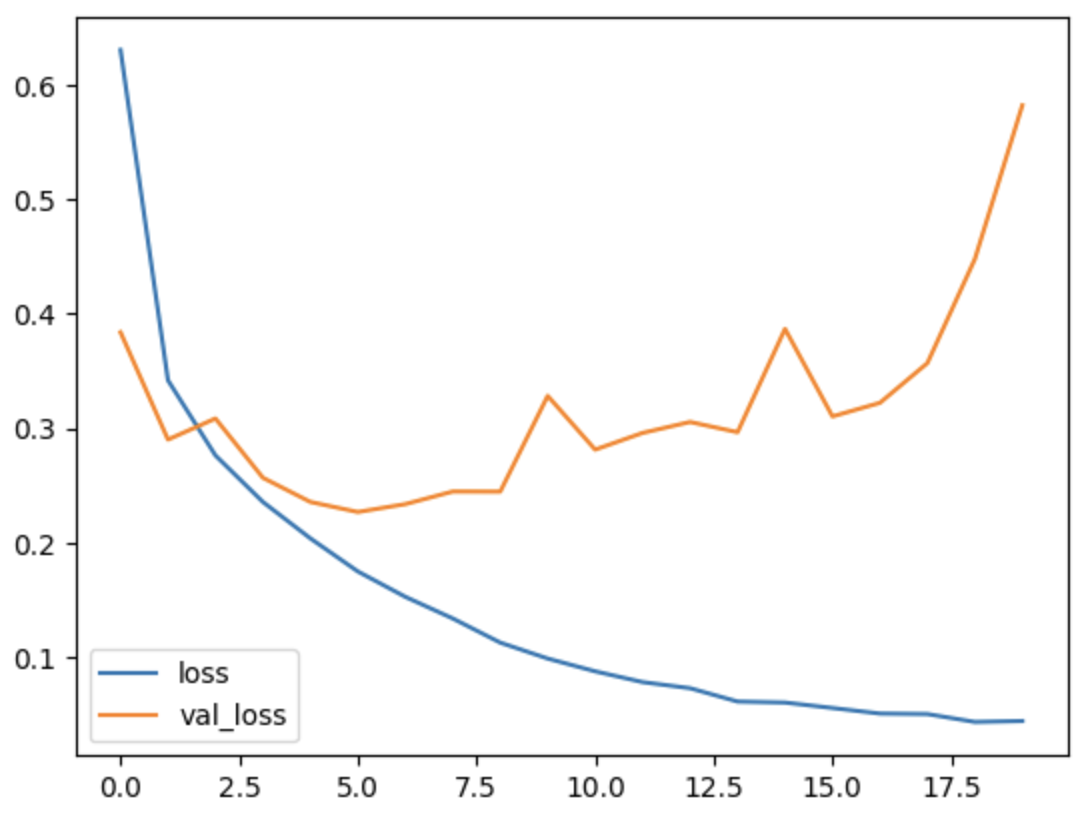
Figure 6: Change 1 Accuracy graph - No overfitting

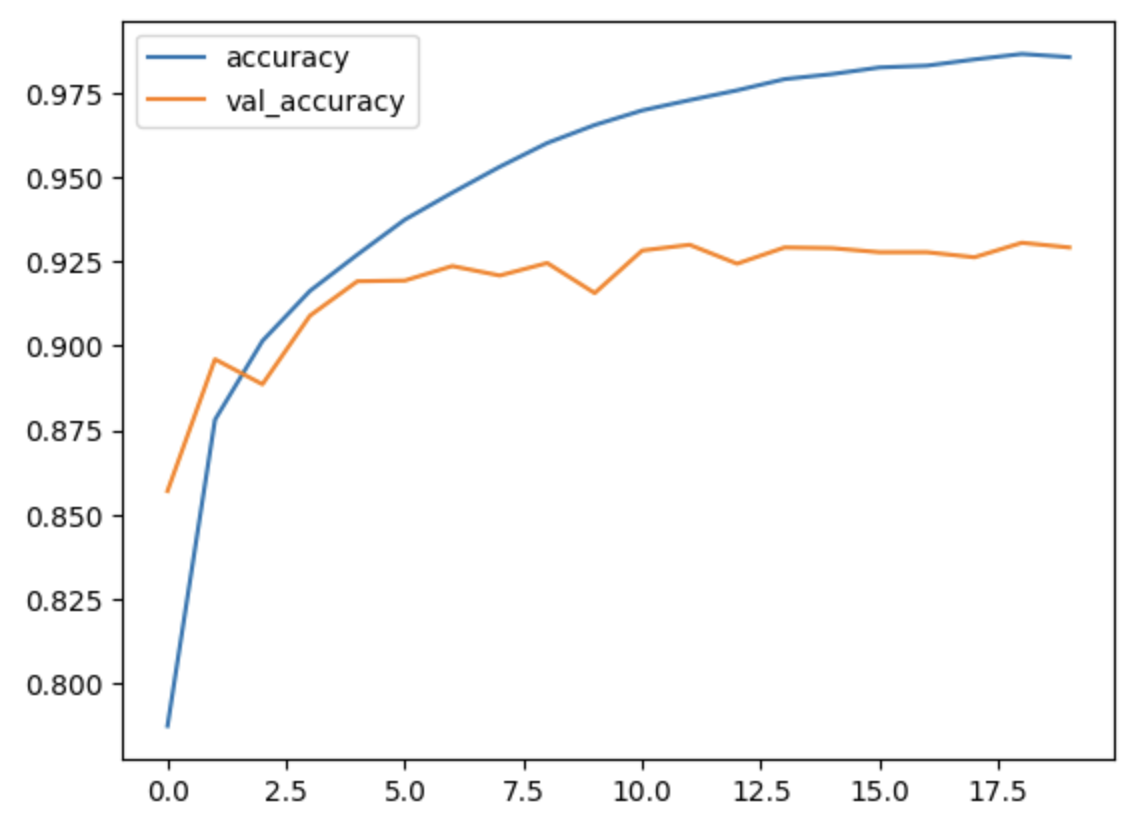
Change-2: Activation function was changed to LeakyReLu from ReLU. Unlike ReLU, this function outputs small positive values to the negative input.

Hence the activation function was changed to LeakyReLU to address the dying ReLU problem. The model was trained and tested with 20 epochs. Training time remained the same as Model1. The loss and accuracy results are:

1. The value of loss started from 0.6187 and ended at 0.0399,
2. The accuracy value started from 0.7934 and finished at 0.9870.
3. Validation/testing loss started from 0.3561 to 0.4377
4. Validation/testing accuracy started from 0.8703 to 0.9289

Please refer Fig 7 and Fig 8 for loss and accuracy graphs and refer Table 1 for tabular presentation of testing results of change 2.

 Figure 7: Change 2 Loss graph - No overfitting

 Figure 8: Change 2 Accuracy graph - No overfitting

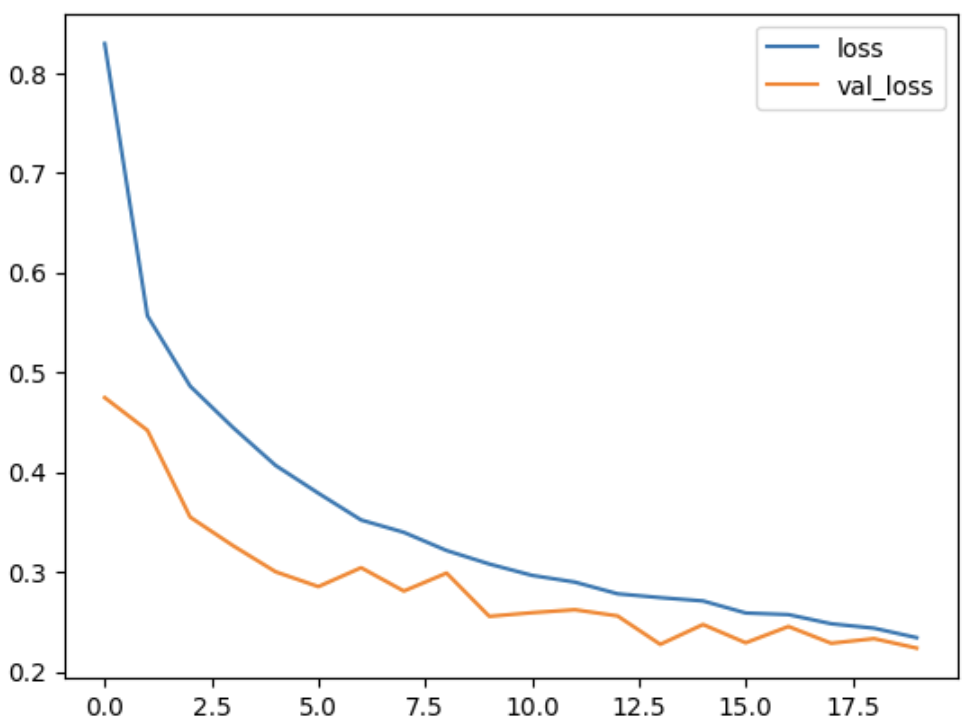
Change-3: Data augmentation helps reduce overfitting problems by generating new image variations for training and hence the model does not train itself specific to certain instances only.

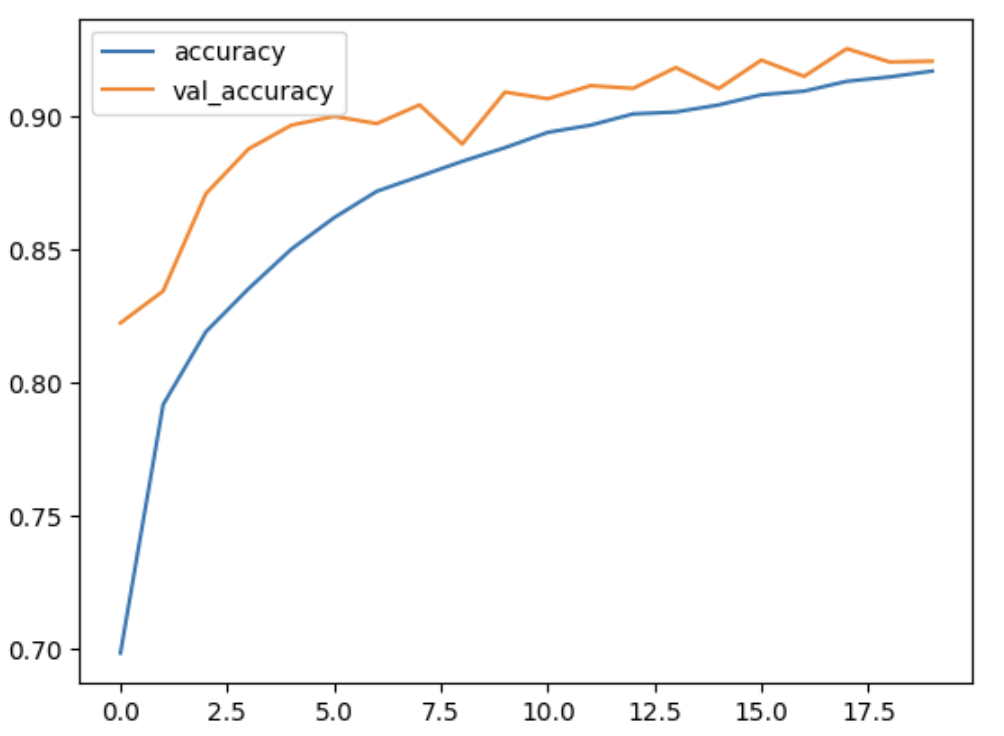
Data augmentation was performed on FASHION-MNIST by shifting the input image horizontally by a maximum of 10% of its width and shifting the input image vertically by a maximum of 10% of its height.

The model was trained and tested with 20 epochs. Training time was more than the training time of Model1. The loss and accuracy results are:

1. The value of loss started from 0.8292 and ended at 0.2341,
2. The accuracy value started from 0.6985 and finished at 0.9168.
3. Validation/testing loss started from 0.4746 to 0.2236
4. Validation/testing accuracy started from 0.8223 to 0.9205

Please refer Fig 9 and Fig 10 for loss and accuracy graphs and refer Table 1 for tabular presentation of testing results of change 3.

Figure 9: Change 3 Loss graph - Overfitting not addressed

Figure 10: Change 3 Accuracy graph - Overfitting not addressed

Change-4: Included VGGNet Architecture to the model proposed in this paper.

Added three blocks of convolutions. Each block has two sets of convolutional layers and Batch Normalization followed by a Max-pooling layer.

After the three blocks are the dense net layers with dropout layers similar to the model in [1].

The model was trained and tested with 20 epochs. Training time improved by 31% . The loss and accuracy results are:

1. The value of loss started from 0.5692 and ended at 0.0908,
2. The accuracy value started from 0.8107 and finished at 0.9668.
3. Validation/testing loss started from 0.3614 to 0.3248
4. Validation/testing accuracy started from 0.8756 to 0.9255

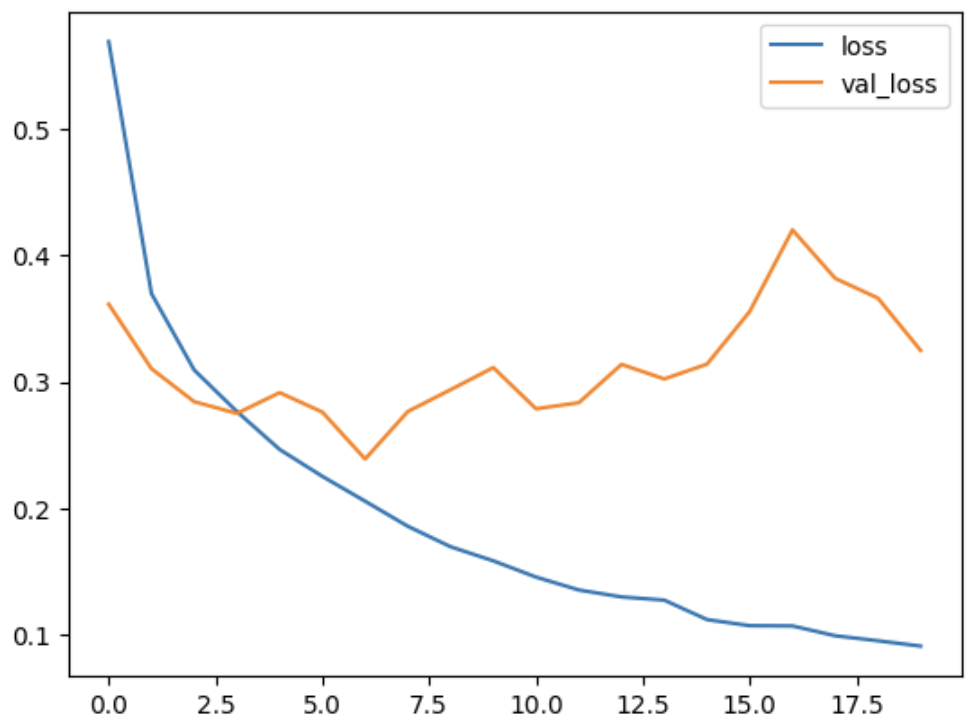
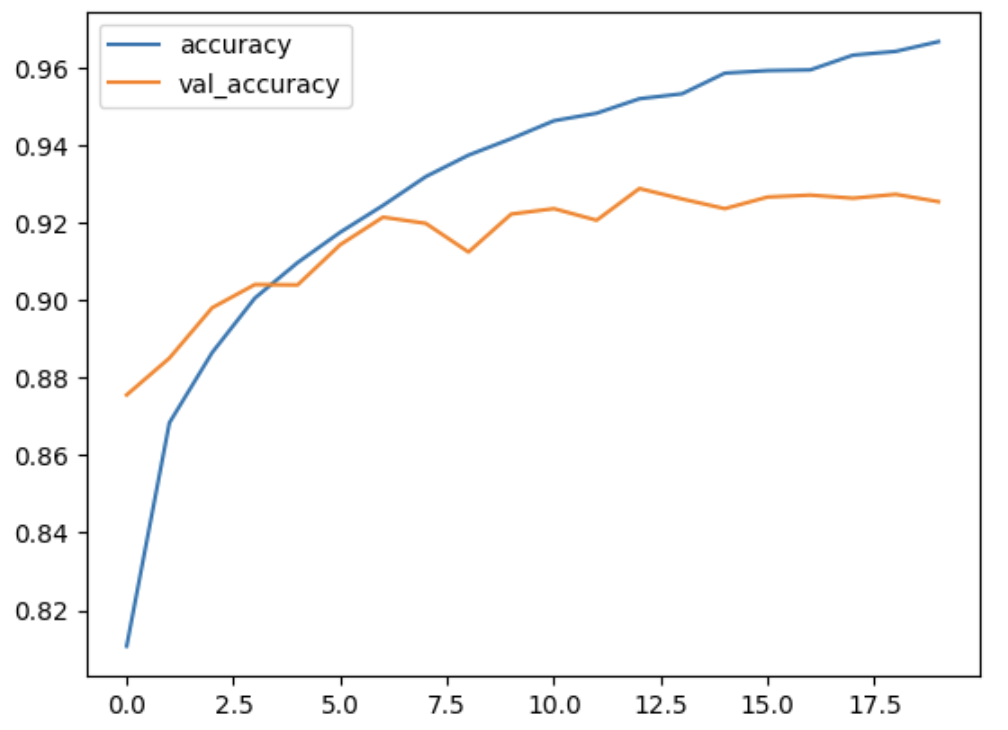
Please refer Fig 11 and Fig 12 for loss and accuracy graphs and refer Table 1 for tabular presentation of testing results of change 4 Figure 11: Change 4 Loss graph - Overfitting addressed 

Figure 12: Change 4 Accuracy graph - Overfitting addressed

Setup: The training and testing of the neural networks are performed online using Google Colab, which provided below mentioned specifications:

1. GPU RAM: 16 GB
2. System RAM: 51 GB
3. DISK: 166.8 GB.
4. Python version 3.6.
5. TensorFlow version 2.5.0
6. GPU V100 with high RAM in Google Colab
7. Jupyter Notebook: There was no support for using GPU with Apple M2 silicon processor. The dataset FASHION-MNIST was too large for the CPU to handle.

On the FASHION- MNIST data sets, the total amount of wall time to learn the CNN model from [1] is 3 minutes 45 seconds, and the time to learn the proposed CNN model with following changes:

1. Kernel size changed to (3,3): Time taken is 3 min 46s
2. Activation Function changed to LeakyReLU: Time taken is 3 min 39s
3. Data Augmentation: Time taken is 6 min 20s
4. Using VGGNet architecture in the proposed model: Time taken is 2 min 53s

# Result and Conclusion

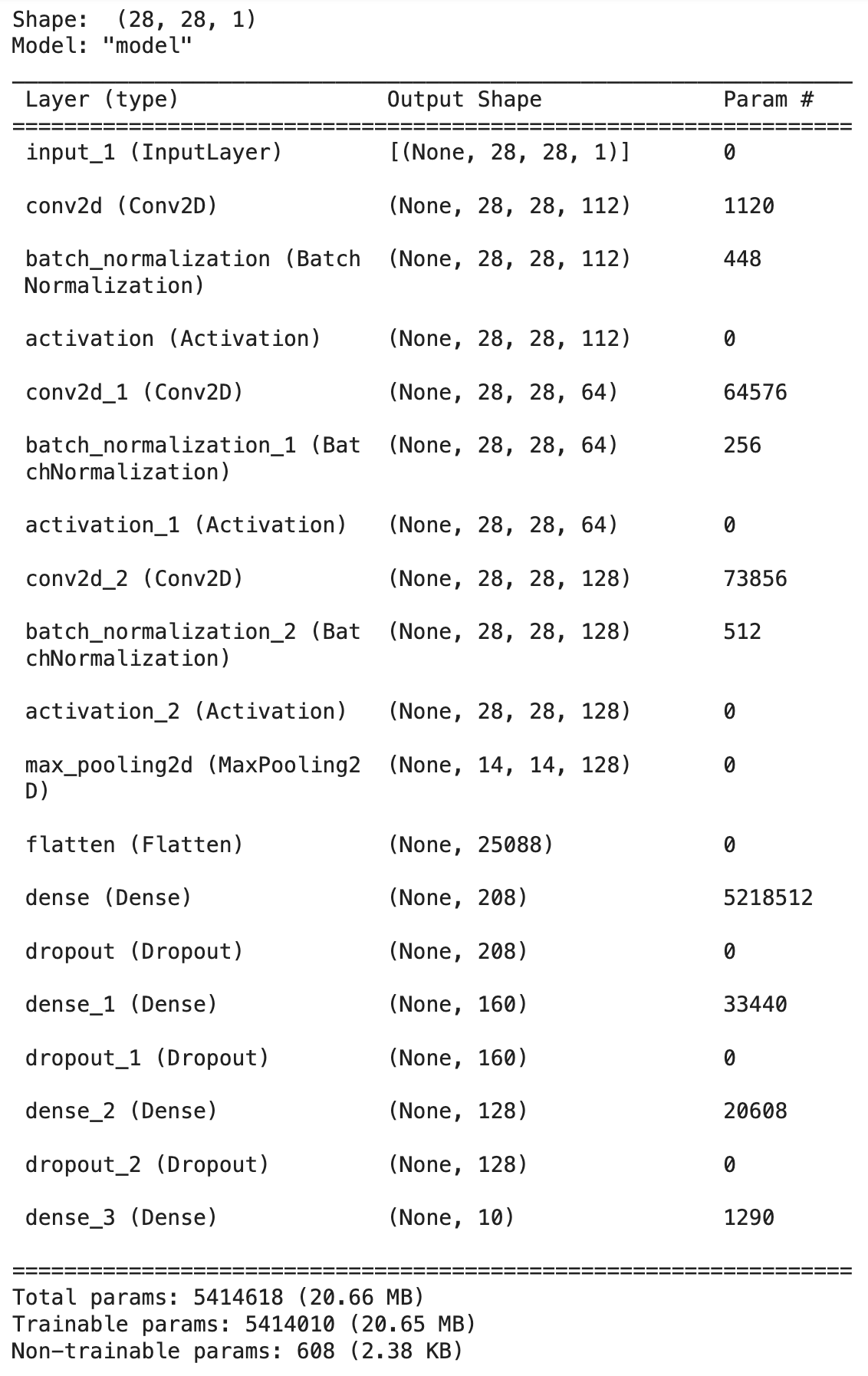
To check the CNN model from [1], the batch normalization and dropout layers are added. To calculate the fitness of the model[1], 20 epochs and batch size of 100 is used on the validation/test data.

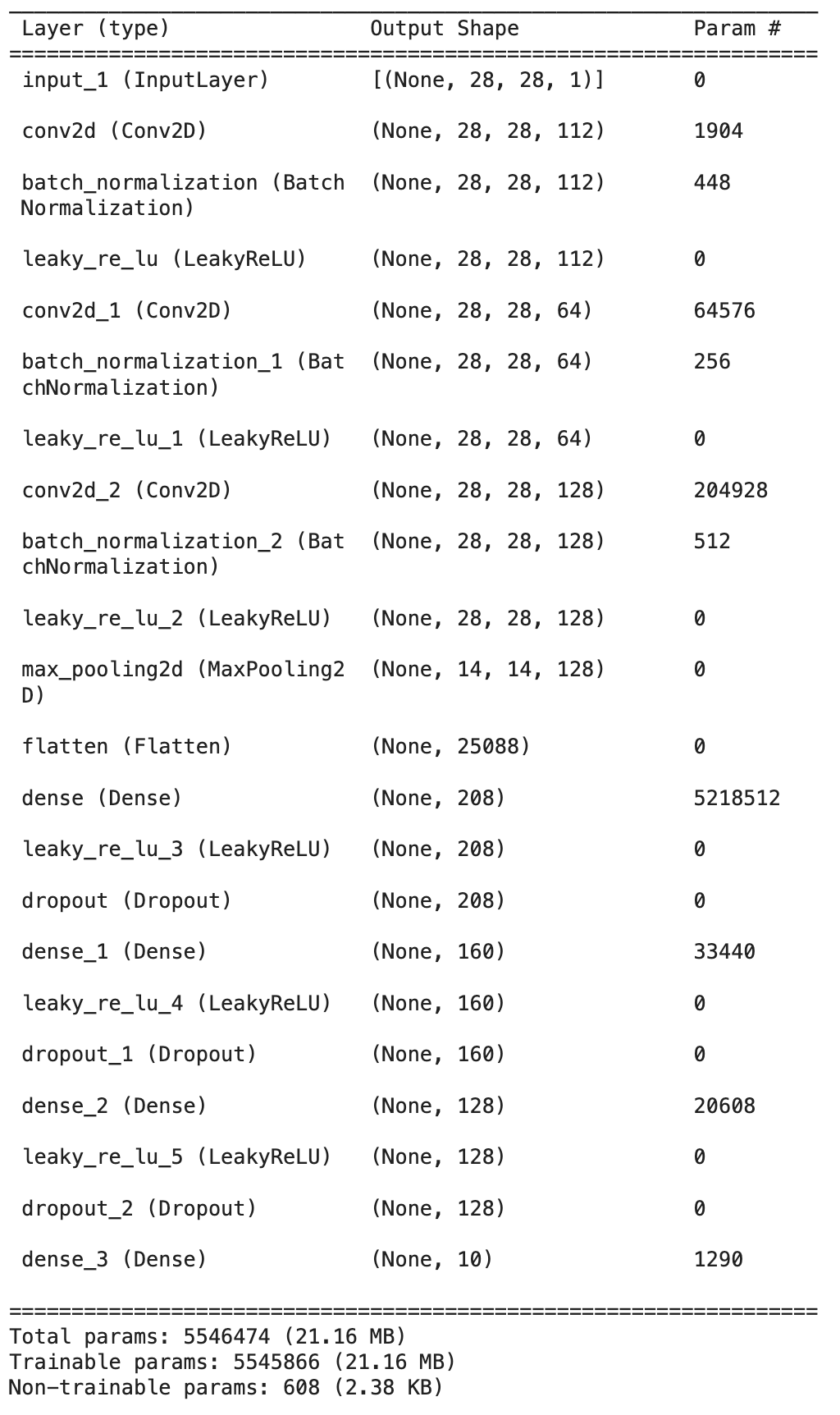
Similarly the four changes have all the components of Model1 along with their respective changes.Refer Table 1 for the tabular result.

1. Loss and Accuracy of All the Models

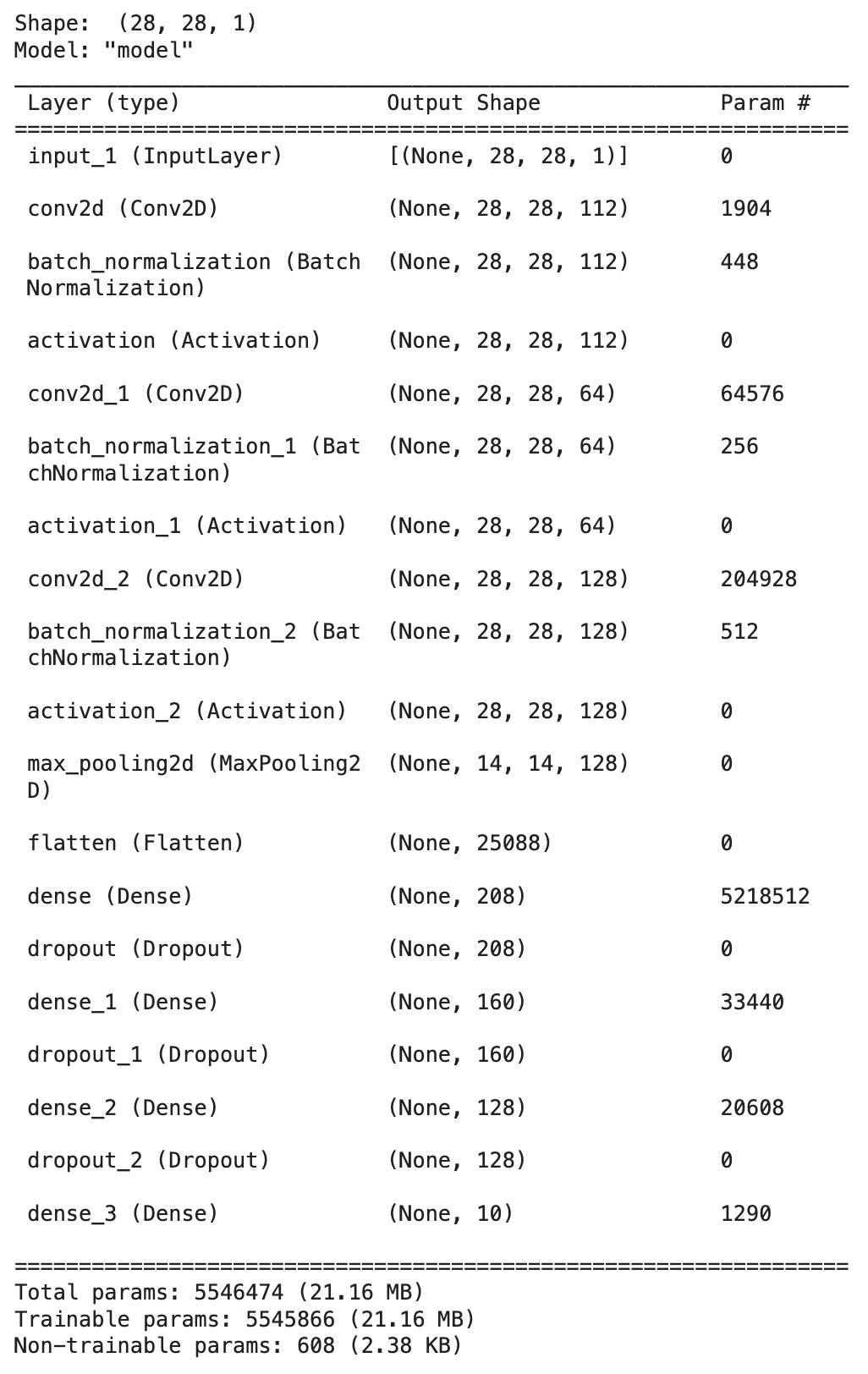
| ***Models*** | | ***Training Loss*** | ***Training Accuracy*** | ***Testing Loss*** | ***Testing Accuracy*** | ***Time taken*** |
| --- | --- | --- | --- | --- | --- | --- |
| Model 1 | Start | 0.6749 | 0.7637 | 0.4089 | 0.8554 | 3min 45s |
| End | 0.0652 | 0.9786 | 0.3195 | 0.9260 |
| Change 1 | Start | 0.6260 | 0.7920 | 0.3635 | 0.8701 | 3min 46s |
| End | 0.0688 | 0.9765 | 0.3525 | 0.9295 |
| Change 2 | Start | 0.6187 | 0.79345 | 0.3561 | 0.8703 | 3min 39s |
| End | 0.0399 | 0.9870 | 0.4377 | 0.92891 |
| Change 3 | Start | 0.8292 | 0.6985 | 0.4746 | 0.8343 | 6 min  20 s |
| End | 0.2341 | 0.9168 | 0.2236 | 0.9205 |
| Change 4 | Start | 0.5692 | 0.8107 | 0.3614 | 0.8756 | 2min 53s |
| End | 0.0908 | 0.9668 | 0.3248 | 0.9255 |

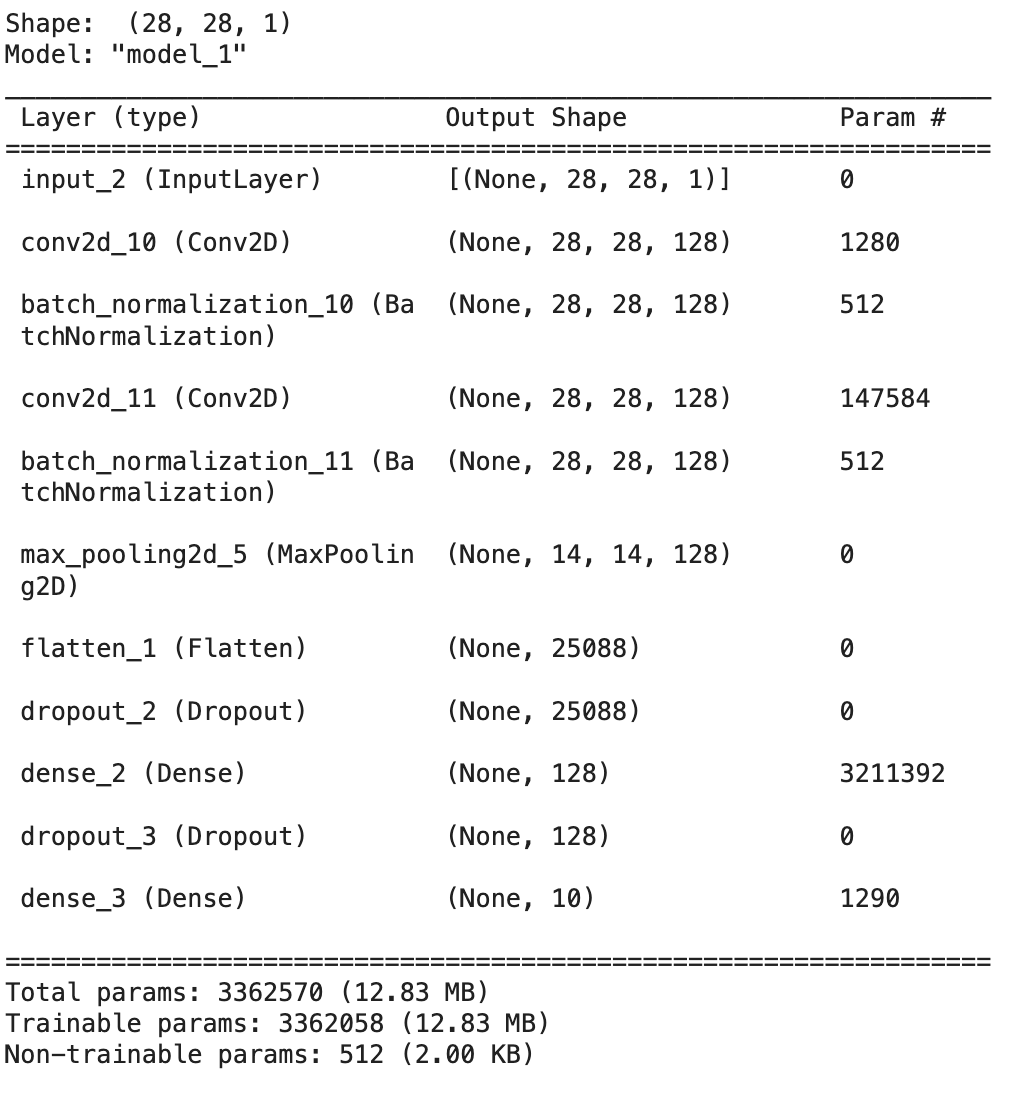
Below are the models of each of the changes.

 Figure 13: Model for the Change 1

 Figure 14: Model for Change 2

To conclude, the inclusion of VGGNet-like (change 4) architecture has yielded improvement by 23% in the learning time of the model at the same accuracy.

 Figure 15: Model for Change 3

 Figure 16: Model for Change 4

##### References

1. E. Xhaferra, E. Cina, and L. Toti, “Classification of Standard FASHION MNIST Dataset Using Deep Learning Based CNN Algorithms ,” 2022 International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT),October 2022, IEEE,DOI: [10.1109/ISMSIT56059.2022.9932737](https://doi.org/10.1109/ISMSIT56059.2022.9932737)

*(*[*https://ieeexplore.ieee.org/document/9932737*](https://ieeexplore.ieee.org/document/9932737)*)*

1. A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” Advances in neural information processing systems, vol. 25, pp. 1097-1105, 2012.
2. M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, “Deep learning applications and challenges in big data analytics,” Journal of big data, vol. 2, no. 1, pp. 1-21, 2015
3. X.-W. Chen and X. Lin, “Big data deep learning: challenges and perspectives,” IEEE access, vol. 2, pp. 514-525, 2014
4. S. Indolia, A. K. Goswami, S.P. Mishra and P. Asopa, “Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach”, International Conference on Computational Intelligence and Data Science, 2018,Procedia Computer Science, Pages 679-688
5. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, 1998
6. R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 580-587
7. M. H. Kolekar and S. Sengupta, “Hidden markov model based video indexing with discrete cosine transform as a likelihood function,” in Proceedings of the IEEE INDICON 2004. First India Annual Conference, 2004., 2004: IEEE, pp. 157-159.
8. C.Szegedy Et Al.,“Going Deeper With Convolutions,”Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1-9.