Shallow Dense Network for Effective Image Classification

Abdul Hasib Uddin
Computer Science and Engineering Discipline
Khulna University
Khulna, Bangladesh
abdulhasibuddin@gmail.com

Abu Shamim Mohammad Arif
Computer Science and Engineering Discipline
Khulna University
Khulna, Bangladesh
shamimarif@yahoo.com

***Abstract* — In this study, we demonstrate high image classification performance using a Dense network with only one hidden layer. In this method, we systematically tuned the number of neurons in the hidden layer and trained our model on a benchmark image classification dataset. The shallow model was able to successfully gain state-of-the-art AlexNet level performance. Neural networks with extensively deep architectures typically contain millions of parameters, which are both computationally expensive and time-consuming to train. This study shows that going deeper into neural networks is not always necessary, rather it is more important to focus on the correct number of neurons in each layer.**

Keywords — shallow, dense, neuron, classification

I. Introduction

In 2012, Krizhevsky et al. introduced AlexNet for image classification [1], which has an overall 660,000 neurons, 61 million parameters, and 600 million connections. It took the authors 6 days to train their network two Nvidia Geforce GTX 580 GPUs in parallel over 90 epochs. Later in 2014, VGG-16 was introduced by Simonyan et al. [2]. It contained collectively 138M parameters. From later on, it has become a go-to trend to go design more and more complex neural network structures incorporating a significantly added number of parameters.

The problem with going deeper is that it requires more sophisticated hardware, such as GPUs, which are quite expensive. Also, training a network for days or weeks without hassle is not always an applicable option. In this study, we tuned the number of neurons along with different activation functions and dropout rates having only one layer and attempted to gain AlexNet level accuracy. We performed experimentation on the Fashion-MNIST benchmark dataset introduced by Xiao et al. [3].

II. Objectives

Our main objective in this research is to find out the optimal neural network architectures that have as few parameters as possible while not compromising the performance. Additionally, this work also exposes the capability of a single hidden layer in a network.

III. METHODOLOGY

The Fashion-MNIST dataset contains 60,000 training and 10,000 testings of 28-by-28 pixel grayscale images for 10 classes [3]. The accuracy performance of AlexNet on the Fashion-MNIST dataset, as reported by Ma et al., is 86.43% [4]. Additionally, Duan et al. applied the VGG-11 network structure on Fashion-MNIST and achieved 91.5% accuracy [5].

The generic model structure is visualized in Fig. 1. It contains three layers - An input layer, one hidden layer, and the output layer. 28x28 pixel images are provided to the network via the input layer. The output layer has 10 neurons for 10 classes. We tuned the number of neurons in the hidden layer based on the number of total pixels in an image. An image has 28x28 (784) pixels. We started training our model with the number of neurons (n) equivalent to 1% on the total pixel, which is 7 neurons only (784x0.01). Then, we gradually increased neurons by taking 78 (10%), 392 (50%), and 784 (100%) neurons. Outputs from the hidden layer were flattened before the output layer.

We generally used a 50% dropout (d) for the hidden layer. However, in two cases, we applied 80% drop out because of their convincing performances to reduce overfitting. Each hidden unit was experimented with without any activation functions and with ReLU activation. In the final layer, we applied softmax activation as the classifier. Moreover, in all cases, we initialized biases with zeros and employed glorot_uniform as the kernel initializer. All the tasks were implemented with Keras Function API.

The total number of trainable parameters were around 54K, 611K, 3M, and 6.1M for corresponding 7, 78, 392, and 784 neurons in the hidden layer. There were no non-trainable parameters. We continued training each model until there was 50 consecutive no improvement in validation loss.

IV. Results and discussion

We have summarized our experimental results in Table 1. For 7 neurons, our model achieved 84.51% test accuracy and a test loss of 0.43 with 87.72% Precision and 81.19% Recall without activation function over 100 epochs. In contrast, the accuracy, loss Precision, and Recall were 83.95%, 0.46, 88.14%, and 79.35% over 158 epochs. In case of 78 hidden neurons, the accuracy, loss Precision, and Recall were 84.58%, 0.43, 87.81%, and 81.67% over 72 epochs without activation; and 84.60%, 0.43, 87.72%, and 81.92% over 88 epochs, respectively. Again, the corresponding accuracy, loss Precision, and Recall for 392 neurons were 84.69%, 0.43, 87.55%, and 81.90% over 65 epochs without activation, while 86.20%, 38.86%, 88.46%, and 84.35% over 287 epochs with ReLU activation. Furthermore, for 784 hidden units, the respective accuracy, loss Precision, and Recall were 84.24%, 0.44, 87.06%, and 81.78% over 69 epochs without activation. On the other hand, these results for 784 hidden units with ReLU activation were 86.18%, 0.39, 88.24%, and 84.30% over 214 epochs, correspondingly.

Now, as the highest test accuracy (86.20%) with test lowest loss (0.39) in the previous models was for 392 neurons (with ReLU activation), we trained the models with 392 neurons with 80% dropout for both with and without activation. Point to be noted that the second-best model in terms of test accuracy was also with 392 neurons, however, without activation (84.69%).

ISSN 2756-9160 / November 2020.
International Conference on Advances in Computing and Technology (ICACT–2020) Proceedings

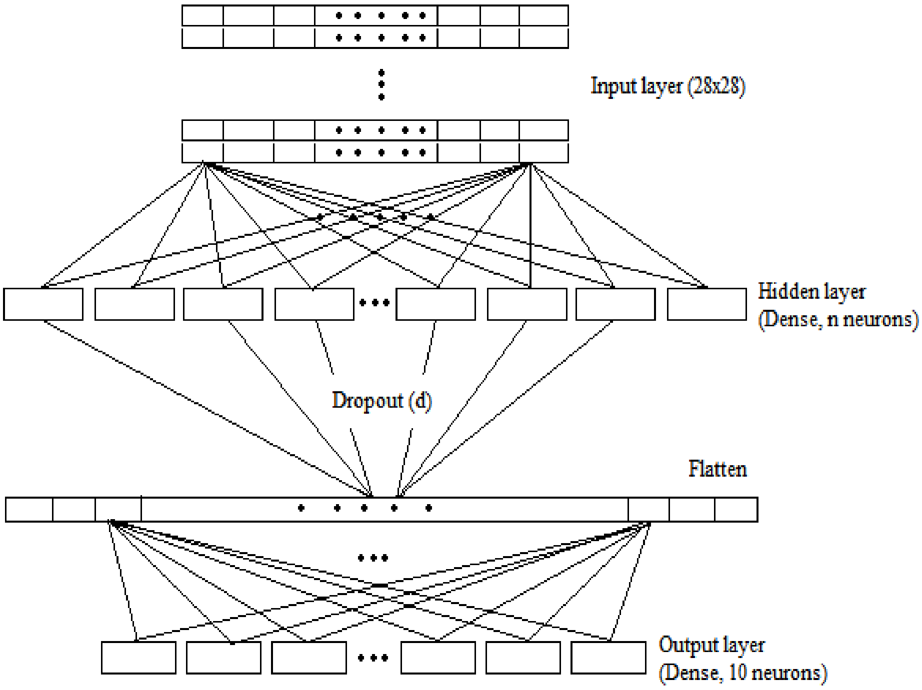

Fig. 1. Generic model structure.

Table 1. Details and evaluation of applied models in comparison to AlexNet (accuracy 86.43%).

# neurons	Activation function	Dropout	# params	# epochs	Test accuracy	Test loss	Precision	Recall
7	None	0.5	54,904	100	0.8451	0.4376	0.8772	0.8119
7	ReLU	0.5	54,904	158	0.8395	0.4626	0.8814	0.7935
78	None	0.5	611,686	72	0.8458	0.4365	0.8781	0.8167
78	ReLU	0.5	611,686	88	0.8460	0.4352	0.8772	0.8192
392	None	0.5	3,074,074	65	0.8469	0.4383	0.8755	0.8190
392	ReLU	0.5	3,074,074	287	0.8620	0.3886	0.8846	0.8435
392	**None**	**0.8**	**3,074,074**	**377**	**0.8636**	**0.3805**	**0.8877**	**0.8455**
392	ReLU	0.8	3,074,074	68	0.8445	0.4390	0.8755	0.8152
784	None	0.5	6,148,138	69	0.8424	0.4443	0.8706	0.8178
784	ReLU	0.5	6,148,138	214	0.8618	0.3884	0.8824	0.8430

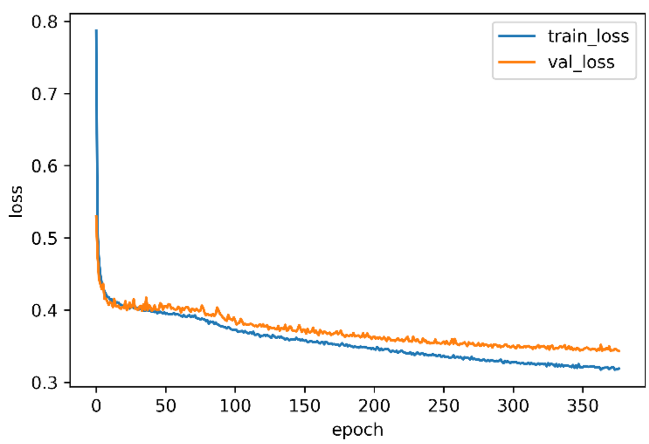

Fig. 2. Training loss versus validation loss for model with 392 neurons and 80% dropout.

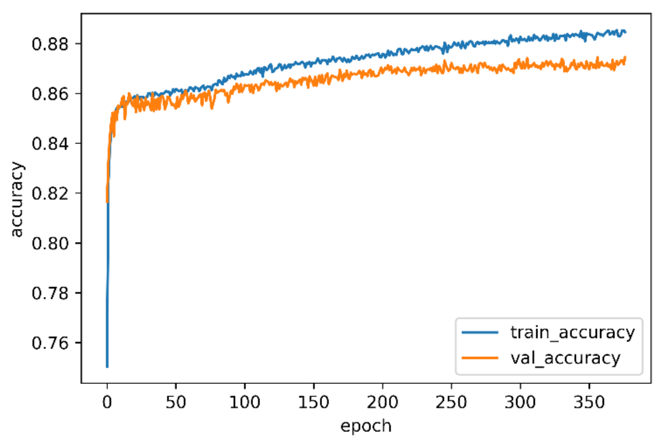

Fig. 2. Training accuracy versus validation accuracy for model and 392 neurons with 80% dropout.

Finally, for 392 hidden neurons with 80% dropout, the accuracy, loss Precision, and Recall were 86.36%, 0.38, 88.77%, and 84.55%, respectively, over 377 epochs, without activation. However, with ReLU activation, the corresponding results were 84.45%, 0.44, 87.55%, and 81.52% over 68 epochs.

Overall, the best performance was for the model consisting of 392 hidden units with an 80% dropout without any activation function. The performance (86.36% test accuracy) was almost the same as the accuracy level of AlexNet (86.43%).

V. CONCLUSION

In this research work, we have demonstrated the powerful capability of the hidden neurons to learn over data. We also investigated the single-hidden layer model competing with very deep AlexNet. Nonetheless, these results should be further investigated intensely with other benchmark datasets. Also, we need to examine if this behavior is also applicable to images with higher dimensions. We also need to construct more similar types of shallow Convolutional models to observe the effects. Considering the proper number of neurons with the correct configuration, we hope that this type of shallow model would largely eradicate our necessity for heavyweight models, thus reducing the requirements of expensive hardware.

REFERENCES

[1] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).

[2] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[3] Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747.

[4] Ma, B., Li, X., Xia, Y., & Zhang, Y. (2020). Autonomous deep learning: A genetic DCNN designer for image classification. Neurocomputing, 379, 152-161.

[5] Duan, Chao, Panpan Yin, Yan Zhi, and Xingxing Li. "Image classification of fashion-MNIST data set based on VGG network." In Proceedings of 2019 2nd International Conference on Information Science and Electronic Technology (ISET 2019). International Informatization and Engineering Associations: Computer Science and Electronic Technology International Society, vol. 19. 2019