A Novel Approach to Deep Learning Applications Using Fork-Join Neural Network Architecture

Sarthak Purohit[1], Anamitra Bhar[2]

[1,2]*Student, Department of Computer Science and Engineering, SRM Institute of Science & Technology, Chennai, India*

---***---

Abstract - *The meteoric rise in the advancements of neural networks has opened a plethora of opportunities for researchers working in this field. Neural networks are a subset of machine learning and are the cynosure of deep learning algorithms. A conventional neural network model comprises of an input layer, one or more hidden layers, and an output layer. Over the years, this model has been processed and fine-tuned to achieve various convoluted tasks of classification and prediction. However, solving yet more complex problems demands a more efficient neural network model. A new approach towards neural networks will either augment the accuracy or the training time. Better accuracy can be attained at the cost of compromising the training time and reduced training time can be achieved with lower accuracy. Nonetheless, the diminution of the magnitude of one factor is comparatively infinitesimal as compared to the upsurge gained in the other factor. In the proposed Fork-Join model, a specific hidden neural layer is cleaved into numerous other layers and subsequently the newly generated independent networks train amongst themselves. Eventually, all the split layers are integrated to obtain a unified layer. The outcome of this model is greater accuracy at the expense of increased training time.*

Key Words: **Deep Neural Networks, Multilayer Perceptron, Accuracy, Validation Loss, Training Time**

1. INTRODUCTION

There is little doubt that machine learning has become one of the most dominant technologies in the last decade. The advancement in algorithms and the exponential burgeoning in computing power have led to an unprecedented escalation of interest in the subject of machine learning. Presently, machine learning techniques are successfully brought into play for classification, regression, clustering, or dimensionality reduction tasks of large sets of high-dimensional input data. [1] As a matter of fact, machine learning has proved to have exceptional potential in numerous fields (such as self-driving cars, [2] image classification, [3] etc.). Moreover, deep neural networks have attained appreciable success in various applications, [4, 5] especially in tasks involving visual information. Numerous state-of-the-art models have been introduced in this field that has exhibited high values of accuracy with cost-effective performance.

Convolutional neural networks (CNNs) have proved to be advantageous and are extensively used to realize various machine learning tasks. However, achieving higher levels of accuracy in image recognition using CNNs is an intricate task as predicting with accuracy is dependent on several factors. Some of the factors involve the characteristics of the dataset, which is being used, the network architecture and so on. Owing to this impediment, analysts are considering the fact that the depth of the network might have a significant role to play in gaining finer performance. [6] Hence, many deep CNNs and deep neural networks are being ideally used.

For the experiment in this study, we are going to make use of two datasets, namely, the MNIST [7] dataset and the Fashion-MNIST [8] dataset. The aforementioned datasets are eminent and are standard datasets that are used in computer vision and deep learning tasks. Although these datasets are credibly solved, they can be used as the basis for learning and practicing how to develop, evaluate, and implement deep learning neural networks for image classification from scratch. This includes how to instigate a robust test harness for estimating the performance of the model, how to explore improvements to the model, and how to save the model and later load it to make predictions on new data. [9]

In previous examinations, it has been corroborated that using the platitudinous fully connected neural networks to train deep neural networks can prove to be computationally expensive. [10] Additionally, it takes significant amount of time to achieve a considerable value of accuracy. The propounded approach of Fork-Join neural networks will aim towards achieving a greater value of one factor, i.e., accuracy, at the expense of another factor, in this case, training time. Nevertheless, the anticipated increase in the training time is not substantial taking into account the additional steps pioneered in the architecture.

2. RELATED WORKS

This section describes the two types of MNIST datasets which will be used in the experiments and then discusses the characteristics of the fully connected neural network model.

2.1 Datasets

To scrutinize how deep neural networks function given the task of discerning characters, two prominent datasets were used in this experiment. Firstly, this paper utilizes the MNIST handwritten digit database [11]. This database can be taken from the page of Yann LeCun (Yann.lecun.com, n.d.). It has become a standard for swift testing of hypotheses on pattern recognition and machine learning algorithms. The MNIST database was concoted out of the NIST database, hence the name, modified NIST or MNIST. It comprises of 60,000 handwritten digit images for the classifier training and 10,000 handwritten digit images for the classifier testing, both extracted from the same distribution. The sizes of all the digits are normalized and centered in a fixed-size image wherein the center of intensity lies at the nucleus of the image with 28 × 28 pixels. Each image sample vector has the dimensionality 28 × 28 = 784, where each element is binary (Deng, 2012).

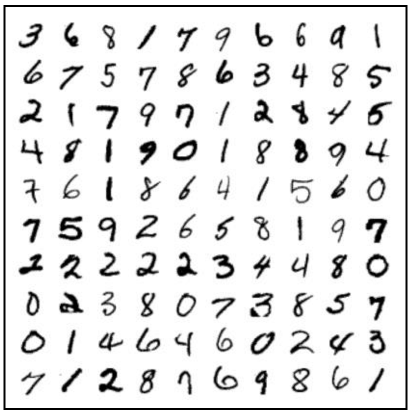
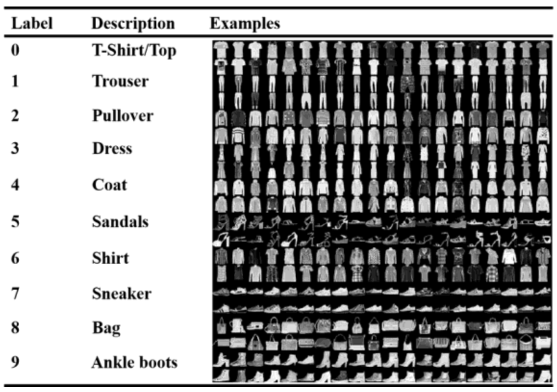


Fig -1: Example images from the MNIST dataset

Additionally, this paper also makes use of the Fashion-MNIST dataset which consists of 60,000 training set pictures and 10,000 test set pictures. Each grey-scale image has a size of 28 × 28 pixels and is catalogued into 10 distinct labels as shown in Fig. 2. [12] The Fashion-MNIST is a substitute for the original MNIST dataset and is considered to be an exemplar for various machine learning algorithms.



Label	Description	Examples
0	T-Shirt/Top	
1	Trouser	
2	Pullover	
3	Dress	
4	Coat	
5	Sandals	
6	Shirt	
7	Sneaker	
8	Bag	
9	Ankle boots	

Fig -2: The Fashion-MNIST dataset

2.2 Multilayer Perceptron

Multilayer perceptron (MLP) [13] is an appendage of feed forward neural network. It includes three types of layers – the input layer, output layer and hidden layer, as shown in Fig. 3. The input layer receives the input signal that is to be processed. The requisite tasks such as classification and prediction are performed by the output layer. An arbitrary number of hidden layers are positioned in between the input and output layer and these hidden layers are responsible for learning the complex patterns in datasets. The data flows in the forward direction from input to output layer and the neurons in the MLP are trained with the back propagation algorithm. [14]

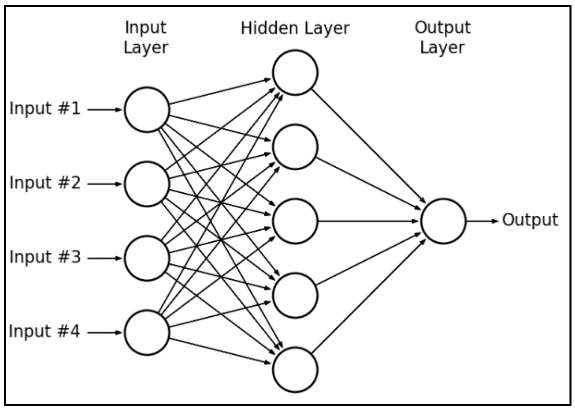


Fig -3: A hypothetical example of Multilayer Perceptron network

Several researchers have used the multilayer perceptron architecture to perform classification tasks on the original MNIST as well as the Fashion-MNIST dataset. The paper written by Wan Zhu suggests the use of the original multiplayer perceptron network, a modified model with autoencoder and another modified model with the use of convolutional neural network for the task of classifying handwritten digits in the MNIST dataset. The results show that the original MLP model performs well, achieving an accuracy rate of 97.65% (Wan Zhu, 2018) with a loss plot which is less smooth than the loss plot generated by the CNN model. Mohammed Kayed et al. [15] perused the performance of the CNN LeNet-5 architecture on the task of classifying garments from the Fashion-MNIST dataset and compared the results with other traditional non-convolute machine learning algorithms. Their results indicate that the MLP model achieved an accuracy rate of 87% which is substantial considering the fact that it dates back to the 1980s.

3. PROPOSED ARCHITECTURE

The motivation for the proposed architecture was obtained from the neural network present in a human brain. The neurons in a human brain aren't always arranged in a fully connected layer-by-layer fashion. Rather, there are anomalies present in the network. In this paper, we make an attempt to visualize and apply one such anomaly in our approach. The neurons present in the distinct layers in Artificial Neural Networks (ANNs) [16] show similar behavior to that of our proposed architecture. The individual layers train and learn on particular features and as a result of that unimportant information (if any) is restrained from flowing from one individual layer to another.

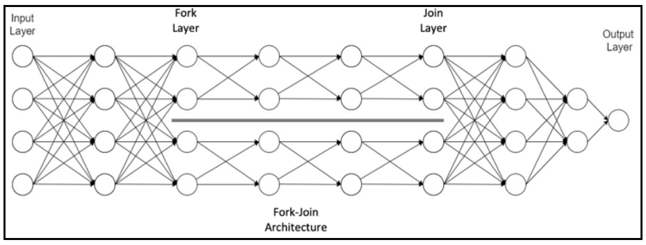


Fig -4: Diagram of the proposed Fork-Join Architecture

The proposed Fork-Join architecture is a rather new-fangled approach which has been derived from the conventional fully connected multi-layer perceptron architecture (Vanilla). Similar to the abovementioned Vanilla model, the Fork-Join architecture also employs the use of three types of layers for the computation namely the input layer, output layer and hidden layer. [17] As evident in Fig. 4, the cardinal point of difference between the two models rests in how the processing occurs in the hidden layers. The job of the input layer is to receive the input signal that is to be processed and the requisite tasks such as classification and prediction is performed by the output layer. An arbitrary number of hidden layers are oriented in between the input and output layer and these hidden layers are split-up in the Fork layer. In general, a single hidden layer is cleaved into two segments. Subsequently, each of the divided hidden pair of layers train amongst themselves.

After the training process is carried out over a predetermined number of hidden layers, the two segments that were concocted in an earlier stage are coalesced or stacked over one another. The recently assembled Join layer is either connected directly to the output layer or trained with additional hidden layers.

All other attributes of the proposed model are kept the same as the traditional model to which it is compared. For instance, the Fork and Join layers are associated with weights and biases. All the layers make use of standard parameters like the activation function thereby generating losses. The data flows in the forward direction from the input layer to the output layer as it conventionally should, and the neurons are trained with the back propagation algorithm. [18]

As a part of the experiments conducted in this research, two neural network architectures were employed and trained on numerous features, which were then compared to each other on the basis of various parameters. Additionally, two datasets were used, namely MNIST and Fashion-MNIST, as mentioned in earlier sections for the sole purpose of producing concrete results. A statistical overview of the two models used has been presented in Table 1.

Table -1: Statistical overview of the two models

Linear Layer	Vanilla		Fork-Join	
	Input features	Output features	Input features	Output features
1	784	1568	784	1568
			Fork(1568) -> 784,784	
2	1568	800	784, 784	400, 400
3	800	400	400, 400	200, 200
4	400	150	200, 200	75, 75
			Join(75,75) -> 150	
5	150	40	150	40
6	40	10	40	10

4. RESULTS

In order to obtain unbiased outcomes both the neural network models that have been employed in our experiment had been allotted equivalent number of hidden layers along with the total number of neurons in each layer. The hyperparameters like learning rate, dropout probability, number of epochs, activation function, batch-size etc. were kept tantamount to one another in the two models. The two neural network architectures were trained independently over the MNIST and Fashion-MNIST datasets with congruent sampled training, testing and validation sets.

4.1 Results as observed on the MNIST dataset:

Vanilla and Fork-Join architectures with parameters presented in Table 1 were trained on MNIST dataset over 10 epochs and the following results were observed:

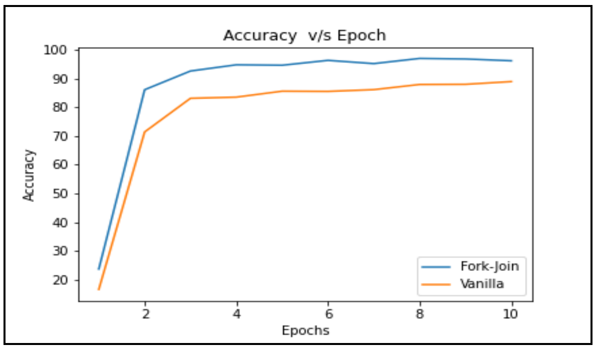


Chart -1: Graph comparing the Accuracy achieved by the two models on MNIST dataset

It is evident from the graph in Chart 1 that the Fork-Join architecture outperforms the conventional Vanilla architecture. Moreover, it can be espied that the highest accuracy achieved by the Fork-Join architecture was approximately 7% above the highest accuracy achieved by the Vanilla architecture.

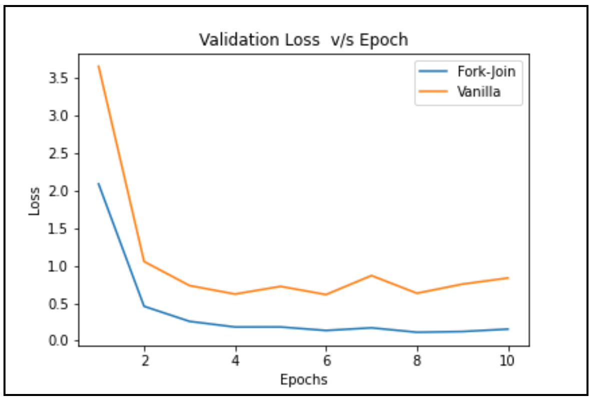


Chart -2: Graph representing the Validation Loss in the two models on MNIST dataset

The validation loss as observed in the Fork-Join architecture was comparatively lower than in the Vanilla architecture.

Table -2: Classification Metrics (MNIST dataset)

Metric	Vanilla	Fork-Join
Precision Score	0.904053	0.971248
Recall Score	0.903600	0.971100
F1 Score	0.903676	0.971085
Jaccard Score	0.824502	0.943983
ROC_AUC Area	0.946475	0.983971

Table 2 presents the calculated classification metrics which provides positive evidence towards the proposed hypothesis. It is apparent that the suggested Fork-Join MLP model is a cut above the traditional Vanilla MLP model.

The aforementioned results indicate that the proposed Fork-Join architecture has an edge over the customarily used Vanilla architecture. We experiment further using another dataset in order to corroborate our results.

4.2 Results as observed on the Fashion-MNIST dataset:

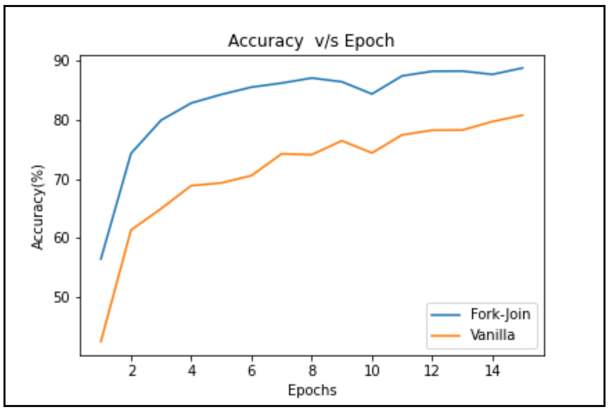


Chart -3: Accuracy comparison on Fashion-MNIST

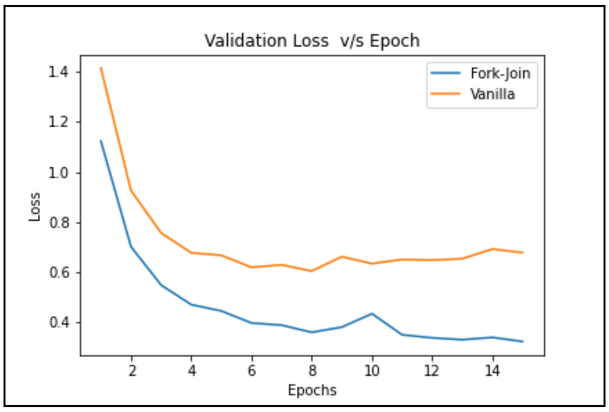


Chart -4: Graph representing the Validation Loss in the two models on Fashion-MNIST dataset

In this case, an intriguing result which can be observed in the Accuracy (Chart 3) and Validation (Chart 4) plots is that both the architectures exhibit fairly analogous trends. Plots obtained from both the models demonstrate congruent positioning of crests and troughs around same nearby epoch. A plausible explanation could be that both the architectures had same number of neurons and hidden layers and they followed the same hyperparameters, thereby displaying a concordant trend in the accuracy and validation loss plots.

Table -3: Classification Metrics (Fashion-MNIST dataset)

Metric	Vanilla	Fork-Join
Precision Score	0.816782	0.877874
Recall Score	0.815700	0.878100
F1 Score	0.815305	0.876896
Jaccard Score	0.698185	0.793180
ROC_AUC Area	0.897611	0.932278

The performance achieved from the Fork-Join architecture is patently superior over that of the Vanilla architecture. Be that as it may, this advantage is obtained at the cost of another factor, which is training time.

Table -4: Training Time of the two models

Dataset	Epochs	Training Time (sec)		GPU
		Vanilla	Fork-Join	
MNIST	10	162	178	Tesla K80
Fashion-MNIST	15	131	157	Tesla T4

From Table 4 it can be concluded that the traditional Vanilla architecture utilizes less time than the proposed Fork-Join architecture in the training process. This anomaly is observed even though there are far more number of inter-layer connections (weights) for the entirety of the separated hidden layers in the Vanilla architecture. Nevertheless, a detailed analysis suggests that the implementation of Fork and Join operations are computationally expensive tasks and this function has to be carried out every time the model is used for training and evaluation, ergo increasing the training and testing time of the Fork-Join architecture.

5. CONCLUSION

The world around us is changing at a rapid pace and there are a lot of concrete advances in the field of technology that one might not hear about, that are nevertheless having a dramatic impact on the lives of humankind. Some of these immense new stories are related to neural networks which are responsible for driving all sorts of progress in numerous sectors from entertainment to medicine. As humans, we have always been aiming towards advancements and changes, and we have discovered that we can build better models by bringing together these artificial neurons and make them work cooperatively in various novel ways. Our paper proposes a newfound neural network architecture, named Fork-Join architecture, and compares its performance against the conventional Vanilla architecture on several factors. From the experiments conducted on the two distinct datasets, it can be concluded that the proposed model achieves greater accuracy and lesser validation loss than the Vanilla architecture. However, it was observed that this supremacy was attained at the cost of higher training time in the case of the Fork-Join architecture and there might be several plausible explanations for this peculiarity. The scope of the posited architecture is not limited to what is presented in this paper. For instance, instead of cleaving the network into two segments, multiple divisions can be generated. Additionally, after splitting the network into a particular number of segments, each of those segments can further be cleaved into multiple segments. These proposals might give rise to neural network models that achieve even greater accuracy and resolve the complication of higher training time.

6. REFERENCES

[1] Marsland, S. Machine Learning (CRC Press, Taylor & Francis Inc., Boca Raton, FL, 2014).

[2] Bojarski, M. et al. End to end learning for self-driving cars. Preprint at arXiv:1604.07316 (2016).

[3] He, K., Zhang, X., Ren, S. & Sun, J. Delving deep into rectifiers: surpassing human-level performance on ImageNet classification. In 2015 IEEE International Conference on Computer Vision (ICCV) (eds Bajcsy, R. & Hager, G.) 1026–1034 (IEEE, Piscataway, NJ, 2015).

[4] Kim, KI, and Lee, KM (2018). Context-aware information provisioning for vessel traffic service using rule-based and deep learning techniques. International Journal of Fuzzy Logic and Intelligent Systems. 18, 13-19.

[5] Lee, HW, Kim, NR, and Lee, JH (2017). Deep neural network self-training based on unsupervised learning and dropout. International Journal of Fuzzy Logic and Intelligent Systems. 17, 1-9.

[6] Gorban, A.N., Mirkes, E.M. & Tyukin, I.Y. How Deep Should be the Depth of Convolutional Neural Networks: a Backyard Dog Case Study. Cogn Comput 12, 388–397 (2020). https://doi.org/10.1007/s12559-019-09667-7

[7] L. Deng, "The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]," in IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 141-142, Nov. 2012, doi: 10.1109/MSP.2012.2211477.

[8] Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. ArXiv, abs/1708.07747.

[9] https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-from-scratch-for-mnist-handwritten-digit-classification/

[10] Graves A. Neural networks. In: Supervised sequence labelling with recurrent neural networks. Berlin: Springer. p. 15–35.

[11] Zhu, W. (2018). Classification of MNIST Handwritten Digit Database using Neural Network.

[12] Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *ArXiv, abs/1708.07747*.

[13] S. Abirami, P. Chitra, Chapter Fourteen - Energy-efficient edge based real-time healthcare support system, Editor(s): Pethuru Raj, Preetha Evangeline, Advances in Computers, Elsevier, Volume 117, Issue 1, 2020, Pages 339-368, ISSN 0065-2458, ISBN 9780128187562, https://doi.org/10.1016/bs.adcom.2019.09.007.

[14] Silva, F.M., Almeida, L.B. Acceleration techniques for the backpropagation algorithm in L.B. Almeida, C.J.

Wellekens (eds.), Neural Networks, Springer, Berlin, pp.110–19, 1990.

[15] Kayed, M., Anter, A., & Mohamed, H. (2020). Classification of Garments from Fashion MNIST Dataset Using CNN LeNet-5 Architecture. 2020 International Conference on Innovative Trends in Communication and Computer Engineering (ITCE), 238-243.

[16] M. Mishra and M. Srivastava, "A view of Artificial Neural Network," 2014 International Conference on Advances in Engineering & Technology Research (ICAETR - 2014), 2014, pp. 1-3, doi: 10.1109/ICAETR.2014.7012785.

[17] J. Singh and R. Banerjee, "A Study on Single and Multi-layer Perceptron Neural Network," 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC), 2019, pp. 35-40, doi: 10.1109/ICCMC.2019.8819775.

[18] V. Skorpil and J. Stastny, "Neural Networks and Back Propagation Algorithm," Electron. Bulg. Sozopol, pp. 173-176, Sep 2006.