Machine Learning Final Report: Survey Analysis of Multi-Layer Perceptrons and Common Convolutional Neural Networks

Julian Paulino

jdp210002@utdallas.edu

School of Engineering and Computer Science

University of Texas at Dallas

Abstract

The article discusses the history and development of machine learning, with a focus on neural networks and their evolution from simple decision-making systems to complex models that can process vast amounts of data. The article emphasizes the importance of understanding the foundational basics of neural networks and how each component interacts with the whole to produce emergent computation. The article also introduces popular neural network models, their structures, and how they process popular datasets. It also discusses the challenges and limitations of earlier models such as the multi-layer perceptron and how they have been overcome through the development of more advanced models.

1. Introduction

Machine Learning is a common topic for modern day research. In both industrial and academic circles, research and development of new architectures, faster processes, and more powerful hardware have given machine learning researchers ample motivation and resources to further expand the field. Machine learning has even broken its way into the mainstream, with web-based models like ChatGPT and SnapChat AI being made available for anyone to use, catapulting the field into the public spotlight [1]. With such a bright future for the field, it remains important to be aware of the beginnings of machine learning, and how past breakthroughs have led to the powerful models we have today.

The field of Machine Learning began with early forays into using computation in more abstract fashions, hoping to develop techniques for pattern recognition. Work on machine learning began around the mid 20th century, with data scientists and computer engineers developing systems of decision making processes like adaptive switching circuits, useful for classification of inputs [2]. These rule-based systems gave a foundation for machine learning to truly hit its stride, with researchers developing techniques for early supervised (training models with human-labeled data) and unsupervised (training models without the use of labeled data) learning.

The 80’s and 90’s saw a shift in focus away from simple decision-making chains and more into neural networks, named so because of the similarity to biological neurons and how they interact in an organism's nervous system. These models added the backpropagation feature, where a model can send data back through itself to learn best how to adjust its weights and biases [3]. The introduction of neural nets began a form of abstract emergent analysis, where the neural network models could extract abstract features from the input data sets to identify common patterns in ways that are not intuitive for human understanding [4]. Neural nets, although powerful, were also very computationally demanding. Thus, the advancements in computational hardware during the turn of the century further boosted the research in neural networks, especially with using Graphics Processing Units, or GPUs, to rapidly compute vast amounts of data, allowing the models to train with huge data sets and fine tune their classification processes [5]. GPU computing platforms like NVIDIA's CUDA are still in popular usage to this day, and are used in the experiments found in this paper [6].

Now, in the modern era, machine learning models like generative pre-trained transformers and their ilk are pushing the boundaries of what machine learning means. Therefore, in order to best contribute to the great coming successes of the research of today, machine learning students must cultivate a rich understanding of the foundational basics of neural networks, and how each component piece interacts with the whole to produce emergent computation. This report aims to provide such a foundation, introducing several popular neural network models, how they are structured, and how they process popular datasets.

2. Related Work

In this modern age of information, it becomes increasingly simple to access forms of knowledge that one may not be able to fully comprehend based solely on their existing experiences. This is exceptionally clear in the machine learning field, where all it takes to implement even some more challenging model architectures is an online blog post and some adequate hardware to follow along. However, most prospective students are hindered from pursuing further into the field because of how complex these models are when one lifts up the hood and takes a peek at the inner workings. In the hopes of assuring these acolytes that not all of these aspects need be incomprehensible apocrypha, we begin by introducing a more tangible component of model design; layer depth.

At the core of a neural network is the neuron. This is an abstract way to reference a specific grouping of values and functions found within the network. The simplest form of this is called the perceptron. First established by Frank Rosenblatt in the 1950s, a perceptron takes in one or more inputs and produces an output based on a system of weights and biases. The weights denote how important a given input is to the end outcome of the network, and a bias acts as a sort of initial barrier for inputs to overcome to activate the function. The output of a given perceptron is then used as the input of another.

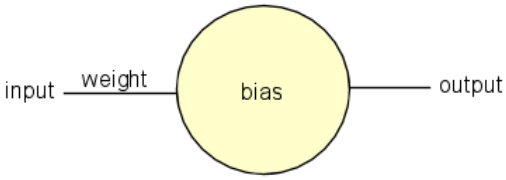

Fig. 1 - a single perceptron

This process can be repeated, producing a model that we call a multi-layer perceptron (MLP) [7]. Alone they are not awfully powerful, but by stacking them together and linking the inputs and outputs, we can draw out abstract features in the input data to help with all sorts of tasks. We can group these perceptrons into layers, with the inputs of a layer consisting of the outputs from the previous layer, and the subsequent outputs serving as the inputs for the following. We abstract the initial data inputs as an "input layer" to help fit the model, the last layer as the "output layer", and all subsequent layers in between what we call "hidden layers" A typical multi-layer perceptron can consist of an input layer, a few hidden layers, and an output layer.

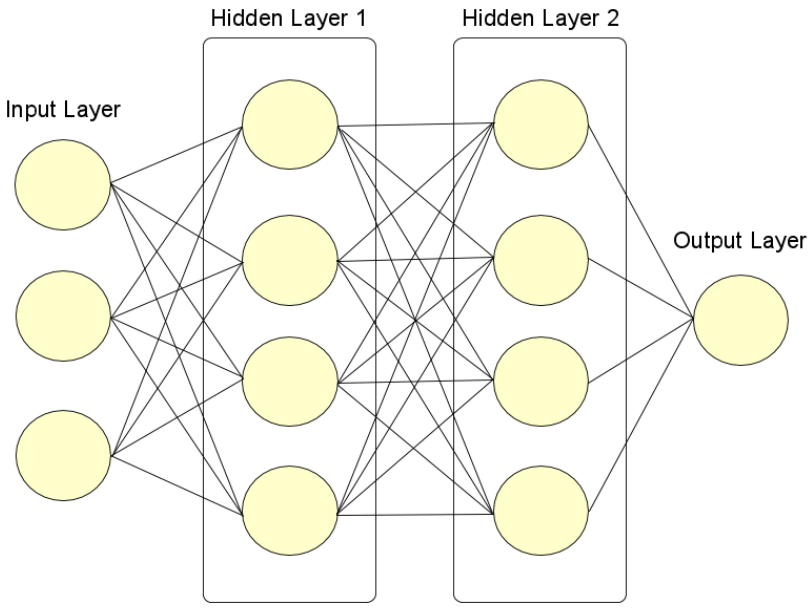

Fig. 2 - a basic multi-layer perceptron

Although powerful for its time, the MLP is more or less obsolete in this day and age. The lack of flexibility means that MLPs can not learn complex non-linear patterns, making it less applicable for real-world scenarios. MLPs are also prone to overfitting, or the process in which a model begins to "memorize" data instead of using common patterns to estimate the output. Overfitting can lead to a model performing poorly when faced with novel data. The biggest shortcoming, however, is the massive amount of computational power needed to train an MLP. Increasing layers and neurons means increased number of weight and biases to keep track of and adjust [8]. These limitations drove researchers to approach the issue with different perspectives, leading eventually to the most commonly taught structure, the Convolutional Neural Network.

Developed near the turn of the century, Convolutional Neural Networks, or CNNs, truly hit their stride in the 2010's. MLPs limitations were greatly apparent when used in image recognition. A group of researchers proposed the usage of a convolutional kernel, or filter, which is a small matrix of values that can "pass over" the image to learn the input's feature map. This process of passing over, called convolution, entails multiplying the values of the filter by the pixel values it is currently over, summing the products, and sending the result into a feature map matrix. The filter then "slides" over to the next section of the image to repeat the process, determining which parts of the image contain important

patterns that we can use to classify the data [9]. Because the pixels in images are directly correlated in a spatial manner, the application of a filter allows the model to determine which groupings of pixel values can determine the classification of an image. During the training process, the model adjusts the values of the filter to better detect important pixel value patterns.

3. Proposed Approach

To cover the foundations of machine learning, this project implemented 3 different architectures; multi-layer perceptron, basic convolutional neural network, and the more advanced LeNet-5 CNN [4]. Each was chosen as they represent a strong starting point for hopeful data scientists and machine learning researchers. As we have discussed MLPs and CNNs in length in the previous section, we will simply go over the reasons they were chosen for this report, and their individual strengths and shortcomings. Multi-layer perceptrons, as we discussed earlier, are able to take on a wide range of issues; classification, linear regression, prediction, and more are all within the capabilities of this framework. Moreover, MLPs are easily scalable and quick to implement; when a dataset consists of a huge amount of data, we can simply add more layers to process the information more holistically. This general, one-size-fits-all approach to MLPs is unfortunately also part of its downfall, as the scaling issue conflicts directly with computational power. When we increase the number of layers in the network to deal with bigger data sets, we exponentially increase the number of parameters needed to output the final results. MLPs are also what can be referred to as "black boxes", or models whose inner workings and methods are difficult to explain in a meaningful way [10]. Some of the issues found

with multi-layer perceptrons can be easily addressed by convolutional neural networks. The ability for CNNs to share parameters via the convolutional kernel method means that CNNs are less computationally demanding than MLPs can be. CNNs are also currently the most efficient way to perform image classification, a leading force in machine learning real-world application.

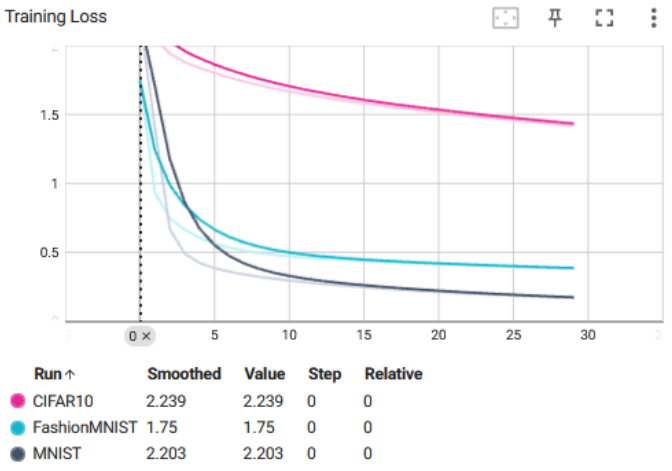

Fig. 3 - graph of training loss for MLP

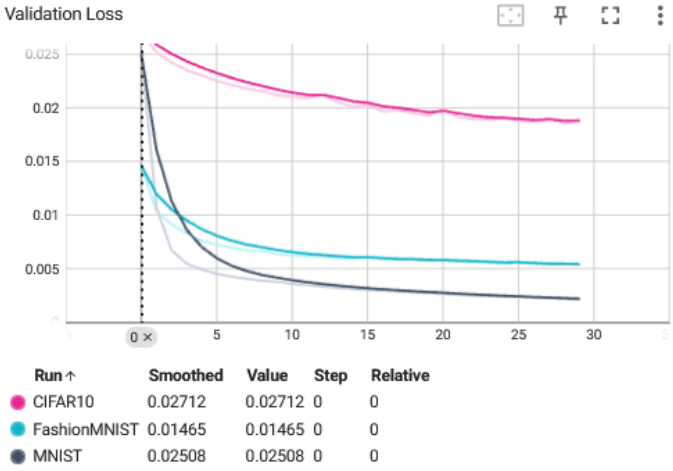

Fig. 4 - graph of validation loss for MLP

4. Experiment

The experiment makes use of Google Colaboratory to run the code, taking advantage of their backend GPUs available for students. We also used tensorflow and tensorboard to capture and visualize the data from the networks. The process consisted of modeling the three aforementioned architectures – multi-layer perceptron, convolutional neural network, and LeNet-5 CNN – and using each to learn and classify 3 different datasets; MNIST, CIFAR10, and Fashion-MNIST.

We begin with the multi-layer perceptron. For the hyperparameters, we set the learning rate to 0.01 and the number of epochs to 30. Above in Fig. 3 and 4, we have graphs of the training and validation or testing losses per each epoch. As we can see, the MLP performed adequately on classifying both the MNIST and the Fashion-MNIST sets, approaching a local minima at around epoch 10. The MLP struggled with CIFAR10 however, unable to move past around a 50% validation rate. We hypothesize this may be due to the increased complexity of CIFAR10, being RGB color scale images as opposed to MNIST and Fashion-MNIST's grayscale.

We then switch out the MLP architecture for a basic CNN. The structure we used is as follows:

1. Convolutional Layer (kernel size = 5)
2. Convolutional Layer (kernel size = 5)
3. MaxPooling Layer (kernel size = 2)
4. Fully Connected Layer
5. Fully Connected Layer

The in-channels, out-channels, and other links were changed to fit each dataset accordingly.

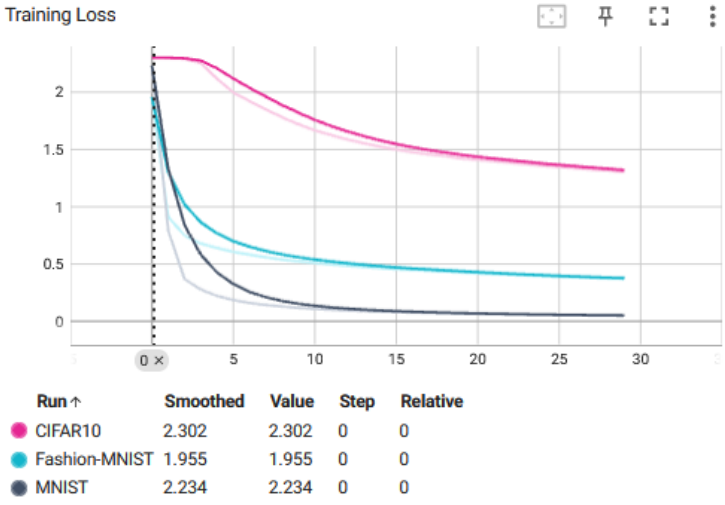

Fig. 5 - graph of training loss for CNN

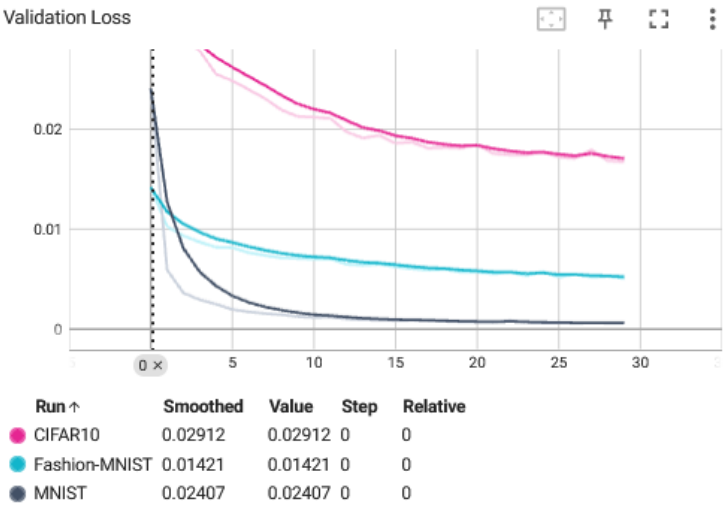

Fig. 6 - graph of validation loss for CNN

As we can see in Fig. 5 and 6, the introduction of the convolution process greatly increased the classification capabilities of the model, with MNIST seeing the most substantial increase. Interestingly, although the CNN had a similar issue with CIFAR10, it was still able to produce a more correct outcome than the MLP.

The final architecture we will discuss is the LeNet-5 Convolutional Neural Network. The LeNet-5 was

first proposed by Yann LeCun et. al. in the late 1990's. It consists of the following:

1. Convolutional Layer
2. Pooling Layer
3. Convolutional Layer
4. Pooling Layer
5. Fully Connected Layer
6. Fully Connected Layer
7. Fully Connected Layer

The combination of convolution and pooling allowed LeNet-5 to pave the way for CNNs in performing image recognition [11]. As we see in our graphs, LeNet performs much faster than both MLP and CNN, able to reach 90% accuracy on MNIST in 4 epochs. The bugs with CIFAR10 still seem to manifest even in this architecture, but we have resolved that the introduction of images with 3 separate color channels will inevitably prove more difficult to classify.

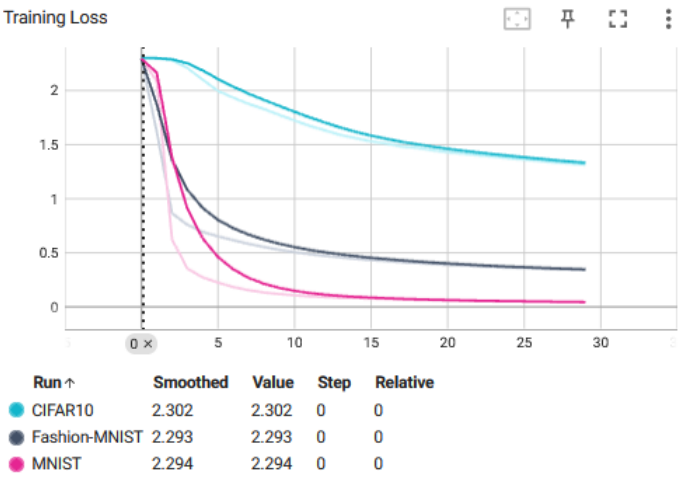

Fig. 7 - graph of training loss for LeNet-5

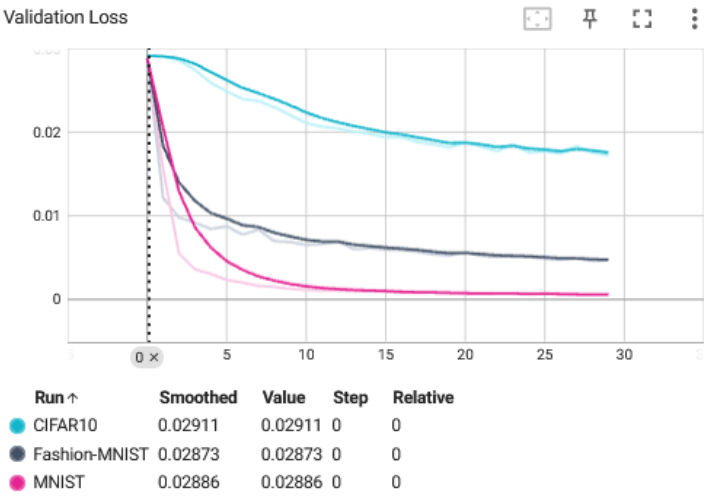

Fig. 7 - graph of training loss for LeNet-5

References

[1] E. Mollik, “ChatGPT is a tipping point for AI,” Harvard Business Review, https://hbr.org/2022/12/chatgpt-is-a-tipping-point-for-ai (accessed May 10, 2023).

[2] B. Widrow and M. Hoff, “Adaptive Switching Circuits,” in 1960 IRE WESCON Convention Record, 1960, pp. 96–104, doi: 10.1109/WESCON.1960.62.

[3] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature, 323*(6088), 533-536.

[4] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation, 1*(4), 541-551.

[5] Raina, R., Madhavan, A., & Ng, A. (2009). Large-scale deep unsupervised learning using graphics processors. Proceedings of the 26th International Conference on Machine Learning (ICML), 873-880.

[6] Nvidia Corporation. (2008). NVIDIA CUDA Compute Unified Device Architecture Programming Guide. Retrieved from https://docs.nvidia.com/cuda/pdf/CUDA_C_Programming_Guide.pdf

[7] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," Psychological Review, vol. 65, no. 6, pp. 386-408, Nov. 1958. doi: 10.1037/h0042519.

[8] H. Guo, "Implementation of the Multilayer Perceptron (MLP) Using Different Activation Functions," 2018 IEEE International Conference on Smart Computing (SMARTCOMP), Taormina, Italy, 2018, pp. 311-315, doi: 10.1109/SMARTCOMP.2018.00062.

[9] 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.

[10] Montavon, G., Samek, W., & Müller, K. R. (2018). Methods for interpreting and understanding deep neural networks. Digital Signal Processing, 73, 1-15.

[11] M. R. Khan, “Lenet-5-a classic CNN architecture,” Medium, https://medium.datadriveninvestor.com/lenet-5-a-classic-cnn-architecture-c87d0b03560d (accessed May 9, 2023).