Artificial Neural Network Implementation for Image Classification using CIFAR-10 Dataset

Technical Report · October 2021

DOI: 10.13140/RG.2.2.12974.43844

Sunday Ajala
Norfolk State University
16 PUBLICATIONS **19** CITATIONS

Artificial Neural Network Implementation for Image Classification using CIFAR-10 Dataset

Ajala Sunday Adeyinka

Norfolk State University, 700 Park Avenue, Norfolk, USA, 23504

1. INTRODUCTION

Artificial neural networks (ANN) is a collection of processing nodes called artificial neurons interlinked to model the human neurological framework in a biological brain using mathematical operations. A neural network is defined by the pattern of links (architecture) among the neurons; the methods, which specify the weights on the connections (learning algorithms); and the mathematical equations that determine the output (activation functions) [1]. In this study, we leveraged on the Multi-Layer Perceptron (MLP) ANN (which is a feed forward architecture) for the classification task.

Thus, as shown in Fig. 1, the MLP-ANN architecture shows a visual representation of a neural network and how inputs may be processed through various layers of neurons in order to yield the required output based on learned experience. The aim of an ANN is to be able to perform well not only on the data used to train it, but on unseen or out-of-sample data. A measure of a network's ability to work well in this regard is known as generalization. A system that generalizes well should therefore be keep its training and validation loss equally low [2].

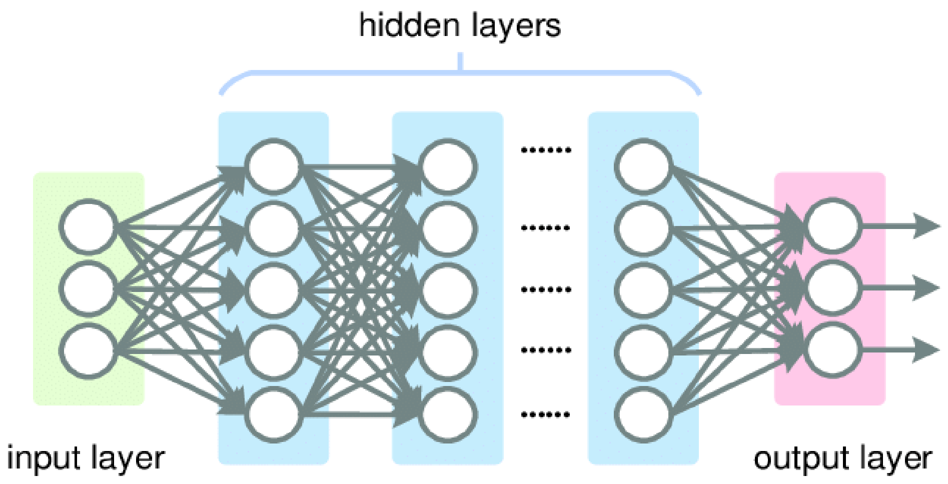


Figure 1. Illustration of a multilayer neural network

In this project, we use an artificial neural network (ANN) to classify images. The CIFAR-10 data set was used in this implementation, which contains 60000 32x32 color images, 50000 training data sets, and 10000 test data sets. The images in this data collection may be clearly divided into ten distinct classes, each of which has been labeled. The system's performance is evaluated using two different topologies with eight and four hidden layers, respectively. The size of the layers are also varied in order to find how the performance of the system responds to different configurations. The implementation has been clearly outlined with results in the interactive python notebook (Artificial_Neural_Network_Implementation_for_Image_Classification_using_CIFAR_10_Data set) attached in a separate file to this report and available on github (Github Link)

2. METHODOLOGY

A MLP-ANN architecture was designed and its performance was evaluated at first. Following that, several performance-improving techniques were applied to this model and analyzed to ensure their usefulness. The following are the various model configurations that were tried and tested:

i. An artificial neural network with 8 hidden layers each of size 512.
ii. A 4-layer artificial neural network with successively reducing layer size (512-256-64-32).
iii. Application of weight decay using L2-norm regularization.
iv. Enabling dropout with the addition of a dropout layer right before the output layer.

In order to actualize the use of the outlined models, the data set was required to be pre-conditioned into a suitable form for model training and evaluation.

2.1 Image Data Processing

The CIFAR-10 data was downloaded first, followed by the training and test data. When a few photographs from the data set are plotted, it becomes clear that CIFAR-10 has a wide range of images, including cars, articulated trucks, and various mammals and birds. The data is normalized to fall within the range of 0.0-1.0 to guarantee that it falls within a reasonable range for our activation functions. The highest value by which we normalize our data is 255 since the data points provided in each image reflect the intensity of each pixel on the red, yellow, and blue scale ranging from 0-255. As a result, 255 was used divide each pixel value so that no value surpasses 1. To avoid programmatic division of integers which discard the decimal values, we cast all data set values as floats which then accommodate decimals which may occur during normalization.

The data is transformed to its grayscale equivalent and plotted again to confirm if the images are properly discernable in order to condense the data and reduce its complexity. Figure 2 shows the colored images and their gray scale equivalents. This normalized, simplified data can now be utilized to train and evaluate the models as part of this project.

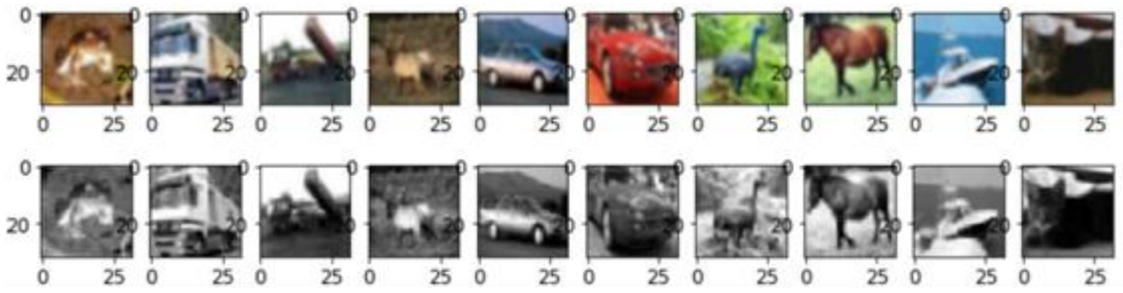


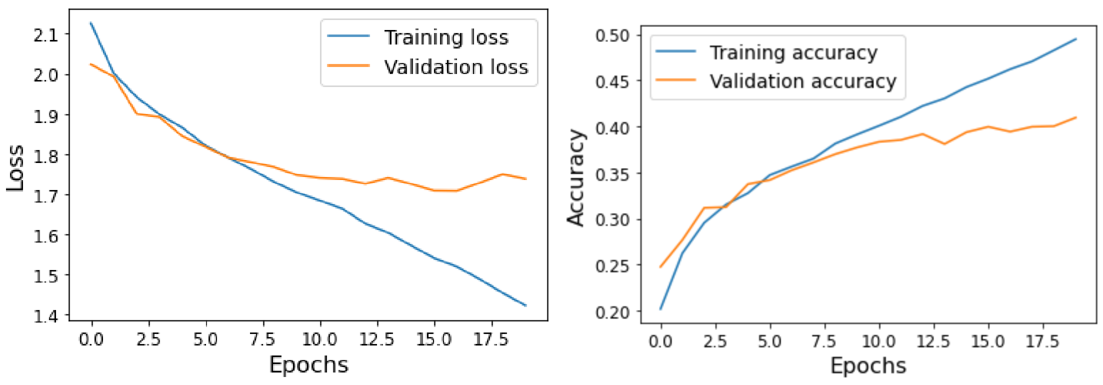
Figure 2. Colored images CIFAR-10 data set sample images (top)
and their gray scale equivalents (bottom)

3. EXPERIMENTATIONS

3.1 8-Layer Ann with Layers of Size 512

As required in the first part of this assignment, an 8-layer model is designed, each layer being given a size of 512 neurons and equipped with a Rectified Linear Unit (ReLU) activation function. Since this is a classification network, the use of a softmax activation function proves suitable for the output layer. Softmax has the property of converting output data into a probability distribution for the outputs, which add up to 1. This therefore makes it suitable in identifying the output that has the highest probability as the target class of the input when well-executed. The system was then trained with 80% for the 50000 training data sets and 20 % was used in validation. A batch size of 128 was used in the training therefore resulting in 313 training batches. The training loss continues to decrease with each training period, as seen in Figure 3. However, after the eighth epoch, the validation loss seems to level before increasing again after the 18th epoch. As a result, the model performs admirably only with training data, and significantly less with validation data.

There was a 10.4% difference between final validation accuracy (38.94%) and training accuracy (49.34%).



(a) (b)

Figure 3. Comparison of (a) Training Loss and Validation Loss (b) Training Accuracy and Validation Accuracy for 20 epochs in a 8-layer ANN with a size of 512 nodes per layer

3.1.1 Handling Overfitting in ANN Model

Now, we can try to do something about the overfitting. There are different options to do that.

i. Reduce the network's capacity by removing layers or reducing the number of elements in the hidden layers.
ii. Apply regularization, which comes down to adding a cost to the loss function for large weights.
iii. Use Dropout layers, which will randomly remove certain features by setting them to zero.

3.2 4-Layer ANN with Reducing Layer Size

We proceed to lower the system's complexity by modifying the network structure in the second iteration of the model. The network is divided into four layers, each with a decreasing number of nodes or neurons, so that the four hidden levels are 512, 256, 64, and 32 in order. The complexity here in orders of magnitude smaller than the first 8-layer model. The amount of learning that occurs during the training period is thus limited. This is done to ensure that the training model's anomalies are not misinterpreted as a generalized behavior for processing test data. The same 128-batch, 80-20 test-validation ratio is used in this model's training.

In terms of validation loss, the 4-layer model outperforms the initial model, as shown in Figure. 4. There is presently a 7.4 % difference between training accuracy (48.97%) and validation accuracy (41.57%). The 4-layer model learns the training data slightly less thoroughly than the 8-layer model, as can be observed from the training loss variations of the two models, because the training loss tends to decline less steeply and shows signals of eventual convergence towards the 20th epoch.

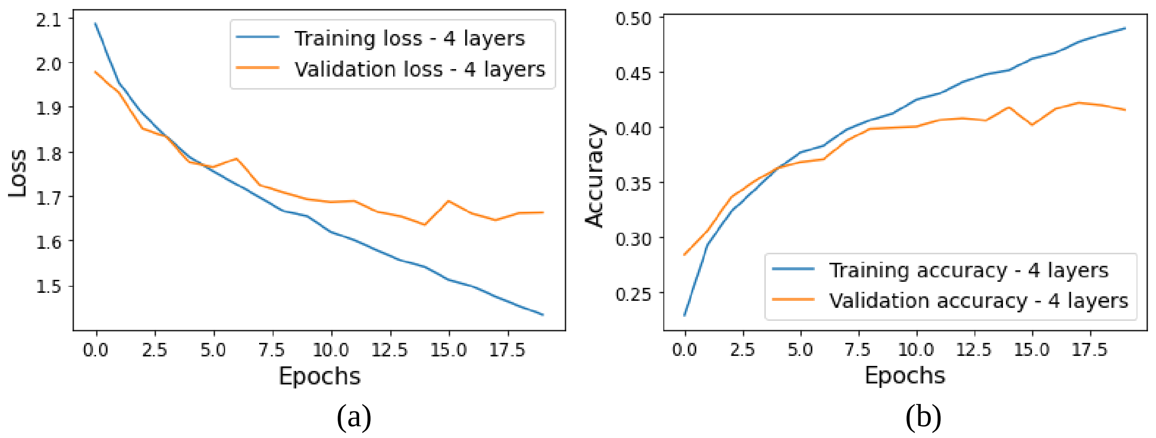


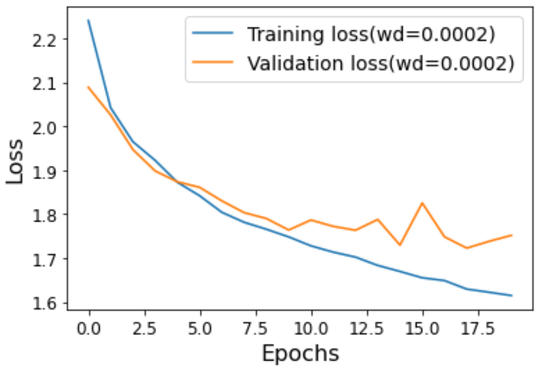
Figure 4. Comparison of (a) Training Loss and Validation Loss (b) Training Accuracy and Validation Accuracy for 20 epochs in a 4-layer ANN with diminishing layer size

However, because of the 8-layer model's over-learning, the training loss continues to decline dramatically during the training phase, widening the gap between training and validation accuracy. It is safe to state that the 4-layer model lowers overfitting when compared to the 8-layer model, demonstrating how simplifying a network can improve its performance. As a result, this network's evaluation accuracy has improved by 28.85 %.

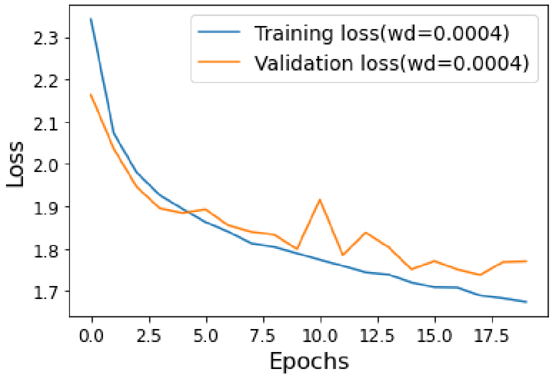
3.3 4-Layer ANN with Weight Decay

To address overfitting, we can apply weight regularization to the model. This will add a cost to the loss function of the network for large weights (or parameter values). As a result, you get a simpler model that will be forced to learn only the relevant patterns in the train data. There are L1 regularization and L2 regularization. L1 regularization will add a cost with regards to the absolute value of the parameters. It will result in some of the weights to be equal to zero. L2 regularization will add a cost with regards to the squared value of the parameters [3]. This results in smaller weights. Weight decay is a regularization method that applies a penalty to weights based on their magnitudes, consequently keeping the magnitudes small.

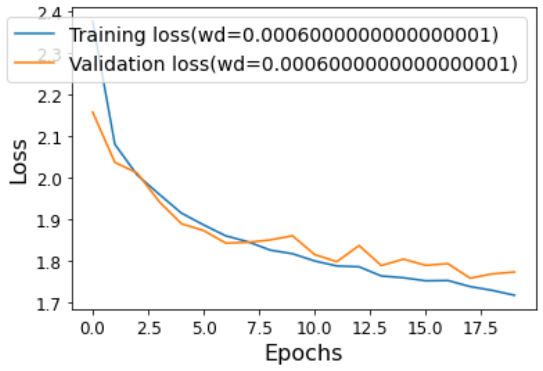
In the definition of each hidden layer in the model, an L2-norm regularization is used to implement this regularization method. The most suitable penalty factor was determined using iteration for values from 0.0002 to 0.0012. The best acceptable penalty factor was identified to be 0.0012. Figure 5 shows the variation of loss as the penalty factor is varied. The optimal value was determined to be at 0.0012.



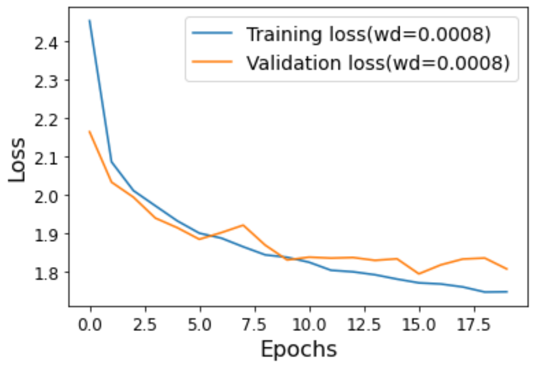
(a)



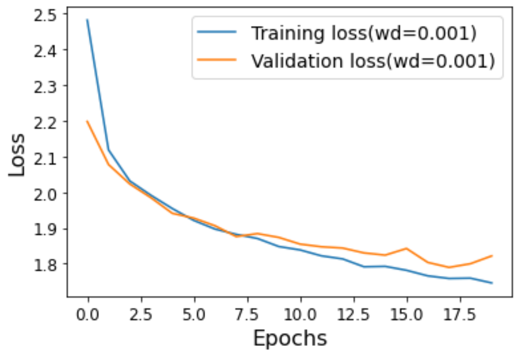
(b)



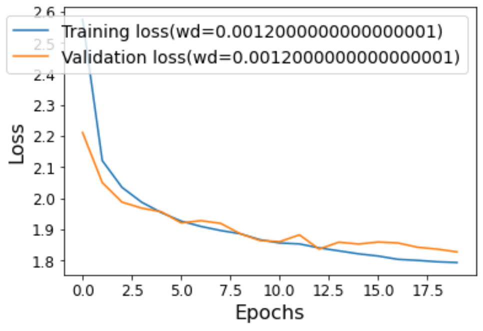
(c)



(d)



(e)



(f)

Figure 5. Comparison of Training and Validation Loss for 4-layer ANN with L2-norm Regularization Penalty Factor of (a) 0.0002 (b) 0.0004 (c) 0.006 (d) 0.0008 (e) 0.001 (f) 0.0012

Figure 6 shows the training and validation loss of the L2-normalized network at this penalty factor. The difference between the final training and validation accuracy can be seen to have reduced to 1.10% and graphically, there is a much higher correlation between the validation and training loss throughout the training period compared to the other five values. Upon evaluation with test data, an accuracy of 38.34% is achieved.

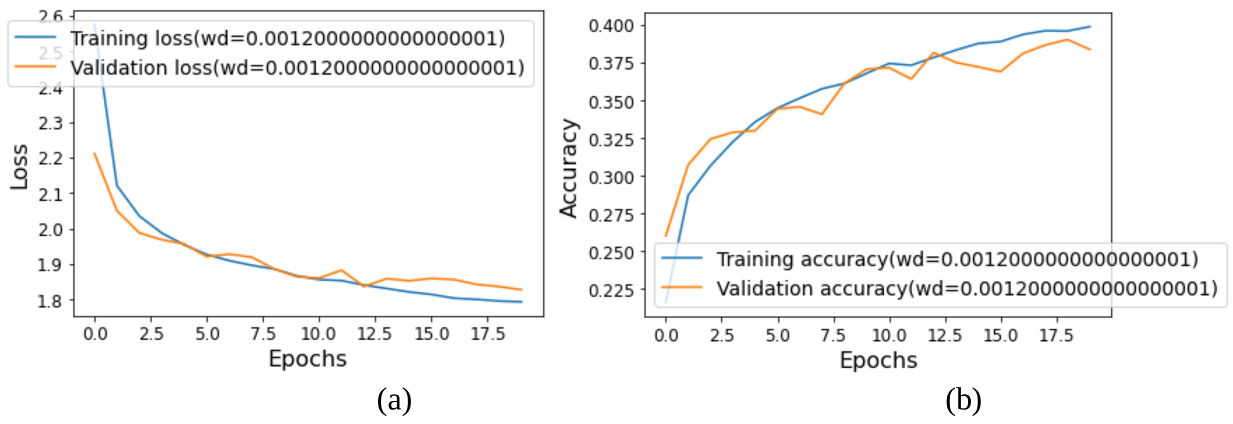


Figure 6. Comparison of (a) Training Loss and Validation Loss (b) Training Accuracy and Validation Accuracy of a 4-layer ANN with L2-norm regularization penalty factor of 0.0012

3.4 Adding Dropout to 4-Layer ANN Model

The last option we'll try is to add Dropout layers. A Dropout layer will randomly set output features of a layer to zero. Dropout is another regularization method that probabilistically simulates the removal of neurons from a neural network. In view of this, dropout achieves the effect of algorithmically reducing the capacity of a network without changing the model layout. By using dropout, we essentially make it very flexible [4]. In essence, certain nodes' outputs are ignored as inputs to succeeding layers in the hopes of neutralizing some nasty reaction of that node to its preceding input. During a training epoch, nodes in various layers are assigned a probability of being dropped. Hidden layers typically have a value of 0.5, while visible layers have a value of 0.2. To the L2-norm regularized 4-layer model in the prior section, we add a dropout layer with a drop rate of 20% in this model.

As illustrated in Figure 7, there is a significant reduction in the disparity between training and validation loss. The difference between the validation and training accuracy was only 0.85%. When evaluating against test data, a validation accuracy of 37.95% is reached, compared to a training accuracy of 37.10%. The usefulness of using a dropout regularizer on a neural network is demonstrated by a good correlation between training and validation performance. This demonstrates the important role of dropout in reducing overfitting.

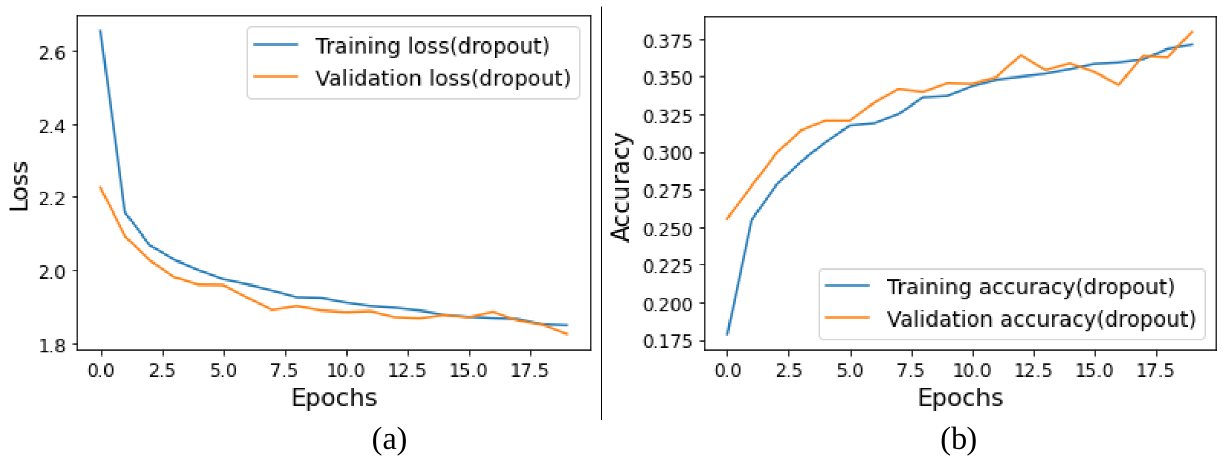


Figure 7. Comparison of (a) Training Loss and Validation Loss (b) Training Accuracy and Validation Accuracy of a 4-layer ANN with L2-norm regularization penalty factor of 0.0012

4.0 CONCLUSION

In this research, we look at Artificial Neural Networks using the CIFAR-10 dataset. Initially, an overfit model is trained using an extremely complex 8-layer model with 512 hidden layers. With the use of a less sophisticated four-layer model with a decreasing number of neurons per layer, overfitting is gradually reduced and eliminated. As a result, the variation between training and evaluation accuracy was reduced from around 10.4% to 7.4%, resulting in an improvement in generalization. This reduced system is then regularized using weight decay (L2-norm regularization) and a dropout layer, which decreases the overfitting further from 7.4% to 0.85%.

REFERENCES

[1]E. Adetiba, V.C. Iweanya, S.I. Popoola, J.N. Adetiba and C. Menon, " Automated detection of heart defects in athletes based on electrocardiography and artificial neural network," Cogent Engineering, vol. 4, 2017, doi: 10.1080/23311916.2017.1411220.

[2]M. Paul, Regularization INFO-4604, Applied Machine Learning University of Colorado Bould, 2018. [Online]. Available: https://cmci.colorado.edu/classes/INFO-4604/files/slides-6 regularization.pdf

[3] V. Flovik, Handling overfitting in deep learning models, 2018. Accessed on: Oct. 9, 2021 [Online]. Available: https://towardsdatascience.com/handling-overfitting-in-deep-learning-models-c760ee047c6e

[4] J. Brownlee, A Gentle Introduction to dropout, 2018. Accessed on: Oct. 9, 2021. [Online]. Available: https://machinelearningmastery. com/dropout-for-regularizing-deep-neural-networks/

[5] Adetiba, E., Ajayi, O. T., Kala, J. R., Badejo, J. A., Ajala, S., Abayomi, A., Badejo, J. A., Adetiba, E., & Adetiba, E. (2021). LeafsnapNet: An Experimentally Evolved Deep Learning Model for Recognition of Plant Species based on Leafsnap Image Dataset. Journal of Computer Science, 17(3), 349–363. https://doi.org/10.3844/jcssp.2021.349.363