Final Project – Autonomous Vehicle Maneuvers

Laila Kamel
Erik Jonnson School of engineering and computer science
University of Texas at Dallas
Richardson, United States
lak170130@utdallas.edu

Andrew Su
Erik Jonnson School of engineering and computer science
University of Texas at Dallas
Richardson, United States
axs180070@utdallas.edu

Rami Ismael
Erik Jonnson School of engineering and computer science
University of Texas at Dallas
Richardson, United States
rmi190000@utdallas.edu

Aadi Kothari
Erik Jonnson School of engineering and computer science
University of Texas at Dallas
Richardson, United States
amk170930@utdallas.edu

Abstract— **In this project an adjustable neural network was implemented on a dataset for autonomous driving cars using Python. The neural network had adjustable hyperparameters, including learning rate, number of iterations, number of hidden layers, number of nodes in each layer, and activation function. The data, consisting of 14,527 instances, contains 14 by 32 grayscale pixel values for images and the corresponding correct moves the car makes after seeing each image. The possible moves for the car are forward, stop, right, and left. The images are not sequential, meaning all the data points are very different from each other, which requires a rigorous implementation of machine learning using neural networks. The most accurate model found in this correctly determined the maneuver on 87.5% of unseen images.**

Keywords—Neural networks, supervised learning, parameter learning, autonomous driving, multiple hidden layers, sigmoid, tanh activation functions

Introduction

In this project, we have chosen to work with an autonomous driving car dataset from Kaggle. The first 448 columns of the data correspond to 14 by 32 grayscale pixel values, and the last column corresponds to the correct response for the direction in which an autonomous driving car should move after seeing the image. There are four possible options for where the car should move: F - forward, L - left, R - right, S - stop. The objective of this project was to create a regular neural network with the optimal number of hidden layers and number of neurons in each hidden layer that can correctly predict which direction the car should move given the 448-pixel values of an image. When developing our neural network, we wanted to have the ability to adjust the number of hidden layers, in addition to the number of nodes in each hidden layer. Additional hyperparameters that were adjusted were the learning rate, activation function, and maximum number of iterations allowed.

Image recognition is a popular application of machine learning. Given the difficult nature of image recognition for computers, the field of machine learning has been attempting to improve and perfect image recognition in computers over the past two decades and has found deep neural networks to be the most effective tool for doing so. In essence, the computer analyzes constructs in each image and is trained to make a classification based on what it can see.

I. Theoretical and Conceptual Study of Neural Networks

Neural networks get their name from being loosely representative of neurons in the brain. Each layer contains several nodes that are connected to the nodes of the next layer by weights, creating a large web of connected neurons. Each connection between two neurons receives a weight that is optimized during the training of the entire neural network. The data feeds forward through the neural network, starting at the input layer. The input layer has a node for each predictor variable (pixels in this case). The first hidden layer in a regular neural network has a specified number of nodes. The input to each of these nodes is calculated by multiplying the weights of the connections with the value at each of the nodes in the preceding layer. Additionally, a bias node and bias connection are included. The input to the neuron is then passed through an activation function that determines what the value of that node will be as it feeds into the next hidden layer. The number of hidden layers, and the number of nodes in each hidden layer are specified, and thus determine the number of connections in the network. Finally, the last hidden layer feeds into the output layer that produce a classification for each data point.

To train a neural network, all of the weights across the network are initialized to random values, and as the model is training, these weights are updated. Forward propagation utilizes the specified activation function at each node, as mentioned above. Backward propagation then performs the task of updating the weights via the backward propagation formula, then the next iteration begins again with forward propagation. This forward/backward propagation will run until the specified number of maximum iterations has been reached. The final error is determined by comparing the predicted classifications and the actual classifications, and then reported.

As the neural network trains and updates weights, it learns the patterns of the data and tries to best approximate which patterns lead to correct classifications.

II. Results and Analysis

Table 1: Results Table

Trial #	Chosen Parameters					Test Accuracy	Notes
	No. of Hidden Layers	Activation Function	Array for Nodes in Hidden Layer	No. of Iterations	Learning Rate		
1	3	Sigmoid	[80,30,10]	3000	0.001	0.8757	Best result
2	3	Sigmoid	[80,30,10]	3000	0.01	0.6716	The learning rate was too high
3	3	Sigmoid	[80,30,10]	1000	0.001	0.8692	This trial ran much faster with ⅓ of the iterations, but the test accuracy was almost as good.
4	3	Sigmoid	[80,30,10]	1500	0.001	0.8668	Not much change between 1000 and 1500 iterations.
5	3	Sigmoid	[160,60,10]	1000	0.001	0.86265	Decent results
6	3	Sigmoid	[80,30,10]	1000	0.01	0.6716	The poor results from this trial confirms that a learning rate of 0.01 is too high.
7	4	Sigmoid	[160,80,30,10]	1000	0.001	0.84509	This is consistent with the results of trial 5; there may be too many nodes.
8	4	Sigmoid	[100,80,30,10]	3000	0.001	0.86403	Decent results
9	3	Sigmoid	[50,25,10]	3000	0.001	0.8609	Decent results
10	2	Sigmoid	[40,5]	1000	0.001	0.824785	This neural network is too small.
11	3	Sigmoid	[200,100,50]	3000	0.001	0.6716	Poor results, trial confirms complex neural network does not mean best result
12	6	Sigmoid	[50,50,50,50,50,50]	3000	0.001	0.6716	Poor results; conclusion – deep learning
13	4	Sigmoid	[30,20,15,10]	1000	0.001	0.834	Decent results
14	2	Sigmoid	[80,20]	1000	0.001	0.8589	Decent results
15	3	TanH	[80,30,10]	3000	0.001	0.67229	Poor results
16	3	TanH	[50,25,10]	1000	0.001	0.67091	Poor results
17	3	TanH	[40,18,10]	1000	0.001	0.53873	Poor results, tanh did not work well with this configuration
18	4	Sigmoid	[20,20,20,10]	1000	0.001	0.821	Good results, we can look into more optimization.
19	4	Sigmoid	[60,15,15,10]	1000	0.001	0.8547	Decent results
20	2	Sigmoid	[120,15]	1000	0.001	0.8602	Decent results
21	1	Sigmoid	[150]	2000	0.001	0.8551	Decent results
22	2	Sigmoid	[140,10]	1000	0.001	0.85611	Decent results

After running 22 trials with various hyperparameters, we achieved a maximum test accuracy of 87.5% using the sigmoid activation function, 3000 iterations, a learning rate of 0.001, and 3 hidden layers with 80, 30, and 10 nodes respectively. The 22 trials conducted varied in each of these hyperparameters, including the number of hidden layers and the number of nodes in each hidden layer. We had many different trials result in 85-86% test accuracy, indicating that our best result is likely close to as accurate as a neural network can be for this non-sequential autonomous car pixel image data. Although our best result required 3000 iterations, a very competitive result of 86.9% accuracy was achieved in trial 3, with all hyperparameters the same as our best trial (Trial 1), except only taking one third of the number of iterations. When we extended to 5000 iterations the accuracy reduced, indicating overlearning. This means with 3 hidden layers composed of 80, 30, and 10 hidden nodes, our neural network is able to optimally find the solution. During our trials, we also found sigmoid to be a much more effective activation function than tanh for this data.

III. Conclusion and Future Work

In conclusion, our deep neural network with sigmoid activation and 3 hidden layers of 80, 30, and 10 nodes performed best with 3000 iterations and a learning rate of 0.001, producing a test accuracy of 87.5% on separate, unseen data.
In this project, we have implemented and optimized a deep neural network for the purpose of image classification for autonomous driving. However, for image recognition, regular neural networks have limitations due to their difficulties in improving model performance given the computationally intensive algorithm.
A possible avenue of future work to improve this project would be implementing a Convolutional Neural Network instead of a Regular Neural Network. A Convolutional Neural Network differs from a deep neural network in that it is not fully connected - it is not a requirement that each node in a hidden layer is connected to each node in the next hidden layer. In industry today, machine learning models have had higher accuracy in image recognition when Convolutional neural networks are implemented in place of regular neural networks. The other option is cost-sensitive neural networks

which would help deal with the imbalance in the dataset. Currently, most of the moves are forward, while only 1% of the moves are "stop". A cost-sensitive neural network may be better at determining when the car should stop.

References

[1] Larry Hardesty | MIT News Office. "Explained: Neural Networks." MIT News | Massachusetts Institute of Technology, news.mit.edu/2017/explained-neural-networks-deep-learning-0414. April 14, 2017.

[2] "Neural Networks for Image Recognition: Methods, Best Practices, Applications." MissingLink.ai, missinglink.ai/guides/computer-vision/neural-networks-image-recognition-methods-best-practices-applications.

[3] Jason Brownlee | "How to Configure the Number of Layers and Nodes in a Neural Network" machinelearningmastery.com machinelearningmastery.com/how-to-configure-the-number-of-layers-and-nodes-in-a-neural-network. July 27, 2018.