Assignment 2

Sonny Desai

February 2020

Abstract

This project concerns Deep Neural networks and analyzing how various factors affect the fitting and ultimate test set performance of these networks. The topology under focus is single versus multi-hidden layer networks. The project examines which model creates the best result and how multi-layers affects accuracy. The main aim is to emulate how a hidden layer learns features by constructing "classes" of input data where classes are perceived to be similar. Then the data is put into those classes that are perceived to have similar features. Lastly, five experiments are conducted to determine what the hidden nodes are actually learning. The experiments are: **Experiment 1:** DNN with 2 layers (no regularization), **Experiment 2:** DNN with 3 layers (no regularization), **Experiment 3:** CNN with 2 convolution/max pooling layers (no regularization), **Experiment 4:** CNN with 3 convolution/max pooling layers (no regularization) and **Experiment 5:** You will conduct several more experiments. (a) Redo all the 4 experiments with some regularization technique. (b) Create more experiments on your own by tweaking architectures and/or hyper parameters.

Introduction

The primary intent of this assignment conduct experiments and compare dense neural networks with a simple single layer versus more complex multiple hidden layered networks. The project hinges on understanding how hidden nodes learn to extract features from their inputs. When there are multiple hidden node layers, each successive layer extracts more generalized and abstract features. When a hidden layer "learns" the kinds of features that are inherent in its input data, it is using a generative method. This project is seeking

to emulate how a hidden layer learns features by constructing "classes" of input data. Through the aforementioned five experiments, an analysis will be done on best hyperparameters, tuning, and number of layers, with a special focus upon the difference in results of a single layer versus multiple layers. The project will thus explore the difference between a single layer Neural network which can only learn linear functions compared to a multilayer perceptron which can also learn non-linear functions. For our project, the non-linear functions are the aforementioned classes in which we will train our models, and then predict upon the test dataset to see which returns the best results.

This project uses the CIFAR-10 dataset. This dataset is a collection of images that are commonly used to train machine learning and computer vision algorithms. Our experiments will examine the affects of hidden layers and nodes on these images and use the predictions to see which model can produce the highest accuracy results.

Literature Review

I found a similar scholarly article through my research comparing single layer and multilayer artificial neural networks. In the article entitled “A comparison between single layer and multilayer artificial neural networks in predicting diesel fuel properties using near infrared spectrum”, the authors chose to create and compare single layer and multi-layer neural networks in predicting the diesel fuel properties using near infrared spectrum (Al-kaf, Chia, Alduis 2018). The researchers created a single layer ANN and compared this model to multilayer ANN with hidden nodes of one to ten. Their results showed that the single layer ANN

achieved better accuracy results than the multilayer models. The research team noted that without data reduction, “the proposed method achieved better performance for all diesel fuel properties when compared with genetic inverse least squares (GILS), and achieved better performance than partial least square (PLS) and ELM for prediction of total aromatics” (Al-kaf, Chia, Alduis 2018). Lastly, although the multilayer ANNs outperformed the single layer network model in the field of total aromatics, the similar scores warrants further investigation of a single layer ANN with different coefficients.

Methods

As mentioned previously, our data comes from the CIFAR-10 dataset which was imported from the keras open source library in python. Pre-processing of the data was done before the creation and testing of the models. The dataset is divided into five training batches and one test batch, each with 10,000 images. The images are 28x28 NumPy arrays, with pixel values ranging from 0 to 255. We used a Sequential class defined in Keras to create our model. For all of the experiments, we used the same optimizer, adam. We defined the loss function as SparseCategoricalCrossentropy. This function acted as the difference between the predicted outputs and the actual outputs given in the dataset. This loss was minimized in order to have a higher model accuracy. For our multi-class classification problem, we used a sparse categorical cross entropy in comparison to binary cross entropy as seen in other DNN multi-classification problems. Finally, we used the accuracy during training as a metric to keep track of as the model trains.

For each experiment, the model was first trained on the training dataset, then tested on the test dataset which was split during our pre-processing. The first experiment was a DNN with

2 layers, the second a DNN with 3 layers. Experiment 3 was a CNN with 2 convolution/max pooling layers, while Experiment 4 used 3 convolution/max pooling layers. These four experiments were conducted with no regularization. Regularization is a technique which makes slight modifications to the learning algorithm such that the model generalizes better. This in turn improves the model's performance on the unseen data as well. One of its main uses is to reduce overfitting. Regularization was added to experiment 5, which re-did all previous experiment with regularization in addition to more experiments with different hyper parameters and architectures.

Results

Experiment 1, a DNN with 2 layers (no regularization), had a final accuracy score for 8.011%. **Experiment 2:** DNN with 3 layers (no regularization), had a final accuracy score of 48.31 %. **Experiment 3:** CNN with 2 convolution/max pooling layers (no regularization), had a final accuracy score of 4.41% **Experiment 4:** CNN with 3 convolution/max pooling layers (no regularization) had a final accuracy score of 10% and **Experiment 5:** You will conduct several more experiments:(a) Redo all the 4 experiments with some regularization technique

Experiment 1: 10.13%

Experiment 2: 37.3%

Experiment 3: 10.08%

Experiment 4: 10%

(b) Create more experiments on your own by tweaking architectures and/or hyper parameters.

Conclusions

The results clearly show that the best performing model with the highest accuracy is the original DNN with 3 convolution/max pooling layers (no regularization) and 2 additional layers. It performed at 91% accuracy nearly double the next best model which was the model with 3 layers and no regularization. The analysis from all the models shows that a lot of the models are very similar in results, for example the two layer DNN with no regularization and the CNN with convolution/max pooling layers (no regularization). The DNN with 3 layers and no regularization performed the second best and with a much greater accuracy than the model with only 2 layers implying that as layers are added, accuracy of the model will increase. This was also the outcome with the two vs three layer models with convolution/max pooling layers, with the three layer model performing best. The outcomes were different for the models with the regularization techniques with the model for 2 layer DNN and the models with the multiple convolution/max pooling layers all having around 10% accuracy. This seems to conclude that the regularization technique I chose improved or kept the data accuracy for the three models the same suggesting the regularization evened out all of the data models to reduce overfitting. Regularization, however, did drop the accuracy score of the 3 model DNN with an 11% drop. I chose the L2 regularization technique which works by adding a term to the error function used by the training algorithm. The additional term penalizes large weight values. The two most common error functions used in neural network training are squared error and cross entropy error. Our models were based on Cross entropy and were thus applicable with L2. The results, unfortunately, show that with this technique either drops the models score or doesn't improve it past 10%

References

Al-kaf Hasan Ali Gamal, Chia Kim Seng, Alduis Nayef Abdulwahab Mohammed (2018). A comparison between single layer and multilayer artificial neural networks in predicting diesel fuel properties using near infrared spectrum.