KVPY – SUMMER PROJECT REPORT

Dropout method for Neural Networks

Done by –
PANKAJ GUPTA
Application No. – SX-11011041

Under supervision of –
Dr. SHARMISHTHA MITRA
Department of Mathematics and Statistics
DR. ARNAB BHATTACHARYA
Department of Computer Science & Engineering

Dropout Method for Neural Networks

The aim of the project was to analyze dropout method for neural networks
When a large feedforward neural network is trained on a small training set, it typically performs poorly on held-out test data. This “overfitting” is greatly reduced by randomly omitting half of the feature detectors on each training case. This prevents complex co-adaptations in which a feature detector is only helpful in the context of several other specific feature detectors. Instead, each neuron learns to detect a feature that is generally helpful for producing the correct answer given the combinatorially large variety of internal contexts in which it must operate. Random “dropout” gives big improvements on many benchmark tasks and sets new records for speech and object recognition.

Overfitting can be reduced by using “dropout” to prevent complex co-adaptations on the training data. On each presentation of each training case, each hidden unit is randomly omitted from the network with a probability of 0.5, so a hidden unit cannot rely on other hidden units being present. Another way to view the dropout procedure is as a very efficient way of performing model averaging with neural networks. A good way to reduce the error on the test set is to average the predictions produced by a very large number of different networks. The standard way to do this is to train many separate networks and then to apply each of these networks to the test data, but this is computationally expensive during both training and testing. Random dropout makes it possible to train a huge number of different networks in a reasonable time. There is almost certainly a different network for each presentation of each training case but all of these networks share the same weights for the hidden units that are present.

We applied the dropout on a dataset of handwritten digits(500 training examples with each example constituted by 20 x 20 pixels image). While using the NN without dropout the accuracy reached was about 88 %. On applying the dropout the accuracy reached to 95%. The method was also tried on few randomly generated datasets.

Project Details

The project was entirely made on Matlab
mat_orig.m was the code without dropout implementation
mat_drop.m was the central file without dropout implementation
randInitializeWeights.m was to initialize theta randomly
nnCostFunction.m computes cost and gradient
fmincg.m was used to minimize cost
predict.m was used to predict the accuracy

The codes are attached with this report.