Comparing Random Forests and Feed-Forward Neural Networks in sentiment analysis

Kenneth Tor Blomqvist

kenneth.blomqvist@aalto.fi

Abstract

This paper compares random forests and feedforward neural networks for classification and regression. An experimental evaluation is done on a dataset of yelp reviews. A theoretical overview of the different methods is given. The implementation of a feedforward neural network is explained. The best performing method for classification was the feedforward neural network. The random forest turned out to yield better results for regression.

I. Introduction

Neural networks are all the rage with the kids these days. News articles are constantly published about neural nets are being applied with staggering results to image recognition, speech recognition, image synthesis, pattern recognition and other problems.

This paper does not mean to provide an in-depth analysis of the best possible approach for a given method. In this paper we seek to compare the performance characteristics and ease of use of a simple feed-forward neural network implementation to random forests. We conduct experiments against a dataset containing yelp reviews for a classification and a regression task.

Our dataset consists of yelp reviews of which the count of 50 different words are given without any contextual information. The regression task consists of predicting how many people found a review useful based on the labeled training data. The classification task consists of trying to predict wether the review received over 3 ”stars” from other users of the service.

Experiments are conducted against these tasks to compare the performance characteristics and other properties of these two very different methods.

This paper is laid out in several sections. In section II a general overview of the different methods compared is given. Section III describes the experiments performed and section IV presents the results of the experiments performed. Section V discusses the results.

II. Methods

I. Feedforward Neural Networks

Feedforward neural networks are a non-parametric estimator that can be used for classification and regression [2]. They are called neural networks because they are formed by composing functions connected by a weight matrix. Each layers output is passed to the next layer thus forming a network. The activation functions can be thought of as ”neurons” and the matrices as connections between them.

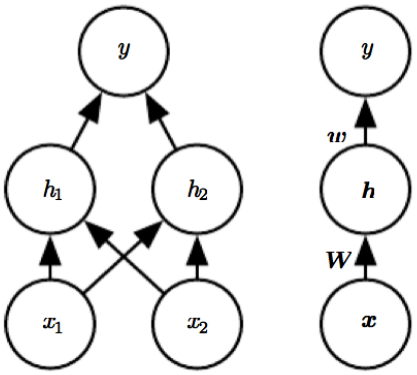

Figure 1: *A depiction of a feedforward neural network drawn in two different styles [1]. The circles are neurons and the lines between them connections between them. x and y are the input and output layers respectively.*

Neural networks can be used to approximate functions such as probability distributions. However, as opposed to many linear models, neural networks can not be trained by using a closed form optimization method. The way neural networks are trained is that a cost function is defined as the optimization goal. The cost function is then minimized using gradient based learning. The backpropagation algorithm is used to update the weights and biases of the model to minimize the error defined by the cost function.

For neural networks, an online learning approach is typically used. Training examples are given in small batches which are evaluated against the cost function. The weights and biases are adjusted to minimize the cost function. Training in this fashion means that the training procedure does not have to store all of the training data in external memory. This makes neural networks well suited for problems with a lot of data.

In this paper we examine perhaps the most simple form of a feedforward neural network: a network with only fully connected layers. Fully connected layers are layers where the input is multiplied by a weight matrix, a bias term is added and the result is passed to some activation function. Typical activation functions include the sigmoid, tanh and restricted linear unit functions.

For classification we create a network with n neurons at the output layer where n is the amount of output classes. These neurons represent each of the output classes. At the output layer a softmax activation function is typically used. The softmax function outputs a n-dimensional vector that sums to one yielding what can be interpreted as a normalized probability distribution over the output classes. Categorical cross-entropy is used as the cost function to optimize the network.

For regression tasks linear activation functions are typically used and the output values are summed to yield the predicted value [**?**]. Mean squared error is often used to minimize the error of the network.

Neural networks are very versatile and can be used for a wide variety of problems. They also work very well in high dimensional settings. They are however computationally expensive to train. They also have quite a lot of parameters that can be tuned. This can make finding a good set of parameters time consuming. Grid search and other search techniques can however be used to help in the search [1].

II. Random Forests

Decision trees are a parametric method for supervised learning [2]. The hypothesis is represented using a binary tree where at each node a binary decision is made. The training examples at the leaf nodes are used to make a prediction. For classification each leaf node represents the output class and for regression the labels for the training data are averaged to get the prediction.

Decision trees are learned by recursively expanding the tree using one of many learning algorithms available.

Random forests are an averaging based ensemble method specifically designed for decision trees. [3] Several models are trained by sampling with replacement from the training set. The predictions of each model is then averaged to construct the final predictions. This way we can use several individual models with

high variance and use averaging to reduce the overall variance. Decision trees typically can have high variance suffering from over fitting. They thus lend themselves very well for ensemble methods.

III. Experiments

In the experiments we studied two different tasks. Classifiying yelp reviews into two classes: those that received over 3 stars and those that didn't. In the regression task we try to predict how many people find the review useful.

The datasets both consist of 50 different features for each datapoint. Each feature indicates how many occurences of a word where found in the review. The data consists of 5,000 training examples and 1000 test examples.

.1 The Neural Network

The neural network was trained by using stochastic gradient descent and the backpropagation algorithm to tune the weights and biases. The network is trained for an indefinite amount of training epochs until the validation set classification accuracy has not improved for 15 traininig epochs. The validation dataset is split off from the training data and it's size is 1/10th of all the available training data.

For optimization, I implemented the RMSProp algorithm as presented in [1]. Theano [4] was used to implement the computational graph and to derive the gradients.

Since there are a vast amount of different hyperparameters available for tuning, the model was tuned by hand. The networks where hand tuned and adjusted after each run. Different networks depths and widths where tried along with tanh, sigmoid and rectified linear unit activation functions. Several l2 norm regularization weights were tried.

.2 The Random Forests

The scikit-learn [3] implementation of a random forest classifier was used. The classifier was tuned by running a grid search to find the best possible model parameters. The parameters that where tuned were the minimum samples left at a leaf of a tree and the maximum depth of each individual tree. The values for the minimum samples at a leaf node ranged from 1 to 10 with a step of 1. The values for the maximum depth ranged from 1 to 48 with a step of 3.

At each step of the search the model was evaluated using k-fold cross validation with 5 folds. The random forest that achieved the best k-fold validation accuracy was used.

IV. Results

I. The Classification Task

I.1 The Neural Network

I.2 The Random Forest

II. The Regression Task

V. Discussion

VI. Conclusion

I still need to write this.

References

[1] I. Goodfellow, Y. Bengio, and A. Courville, "Deep learning," 2016, book in preparation for MIT Press. [Online]. Available: http://www.deeplearningbook.org

[2] E. Alpaydin, *Introduction to machine learning*. MIT press, 2004.

[3] "Scikit-learn: Machine learning in python," 2016. [Online]. Available: http://scikit-learn.org/stable/

[4] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints*, vol. abs/1605.02688, May 2016. [Online]. Available: http://arxiv.org/abs/1605.02688