Artificial Intelligence B003725
Kernel-based voted perceptron

Filmon Arefayne

2019-11-26 Tue

Abstract

The voted perceptron method is based on the perceptron algorithm of Rosenblatt and Frank. The algorithm takes advantage of data that are linearly separable with large margins. This method is simpler to implement, and much more efficient in terms of computation time as compared to Vapnik's SVM. The algorithm can also be used in very high dimensional spaces using kernel functions.

1 Introduction

In this report, we try to reproduce some of the results reported in section 5 of the Freund and Schapire 1999 article [1] (in particular the graphs for $d = 1$ and $d = 2$ in Figure 2).

In the paper, they introduced a new and simpler algorithm for linear classification which takes advantage of data that are linearly separable with large margins: Voted-perceptron.

Based on the perceptron algorithm (Rosenblatt. 1958, 1962) and a transformation of online learning algorithms to batch learning algorithms developed by Helmbold and Warmuth (1995).

If the data are linearly separable, then the perceptron algorithm will eventually converge on some consistent hypothesis (i.e., a prediction vector that is correct on all of the training examples).

Thus, for linearly separable data, when $T- > \infty$, the voted-perceptron algorithm converges to the regular use of the perceptron algorithm, which is to predict using the final prediction vector.

2 MNIST dataset

The MNIST database of handwritten digits, available from this page, has a training set of $60,000$ examples, and a test set of $10,000$ examples.

It can be found here [2] **http://yann.lecun.com/exdb/mnist**

3 Multiclass classifier

To implement the multiclassifier technique we use the OVA/OVR (One versus All) approach where we compute the score on an instance x using each classifier and choose the highest score as the class.

$$y = argmax(s_l)$$

Figure 1: Example of a training images from the MNIST dataset

4 Kernel-based classification

Ideally, we want to find a linear separator that makes the minimum number of mistakes on training data but this is NP-Hard. We use kernels because Perceptron algorithm works best with linearly separable data and it was suggested because is proven that the Perceptron convergence does not depend on dimensionality. Kernel functions compute inner product between transformed vectors using a shortcut simpler functions that take original vectors as input. In the paper, they use the polynomial kernel.

Classification accuracy: If the data distribution is more separable in the high dimensional space induced by the Kernel, classification accuracy will be better.

4.1 Complexity

Parameters:
d = # of dimensions
n = # of training samples
k = # of errors/support vectors
c = kernel computation complexity

Standard Perceptron Complexity: Training: $O(d * n)$ Test: $O(d)$

Vote/Average Perceptron Complexity with Kernel: Training: $O(c * k * n)$ Test: $O(c * k)$

I'd like to thanks **http://people.cs.georgetown.edu/ huiyang/cosc-878/** for the complexity analysis.

5 Implementation and Technical details

Source code was written in Python although most of the code was compiled with **Numba** that is a JIT compiler that makes the code run faster. The kernel trick is computational heavy so running the experiments on the test platform A(laptop) took 2-3 days. There is also a **Google Colab** notebook that is memory hungry and it stores the **Gram matrix** to save computational time and can run the experiments in a couple of hours.

6 The Experiment

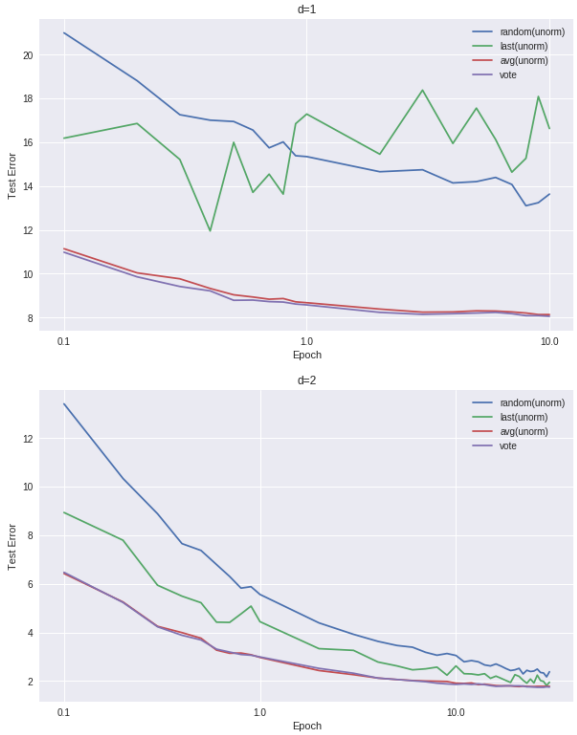

Figure 2: The plots of the test error as a function of epochs for d=1 and d=2

The results agree with the paper's experiments and with $d = 2$ we have a significant improvement on the number of errors and on the time of computation.

6.1 Test Platform Specifications

This are the test platforms used:

Test Platform	A (laptop)	B (Google Colab)
Processor	Intel Core i5-8265U	Intel(R) Xeon(R) 2.20GHz
Memory	8GB	25GB
Operative System	Windows 10	Linux

References

[1] Schapire Freund. “Large Margin Classification Using the Perceptron Algorithm”. In: (1999).

[2] Yann LeCun, Corinna Cortes, and CJ Burges. “MNIST handwritten digit database”. In: *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist/* 2 (2010).