Data Stream Mining

Jesse Read
Universidad Carlos III, Spain

Albert Bifet
Yahoo! Research Barcelona, Spain

INTRODUCTION

Streaming data analysis in real time is becoming the fastest and most efficient way to obtain useful knowledge from what is happening now, allowing organizations to react quickly when problems appear or to detect new trends helping to improve their performance.

Evolving data streams are contributing to the growth of data created over the last few years. In 2013 it is estimated that we are creating more data every two days, that the quantity of data we created from the dawn of time up until 2003. Evolving data streams methods are becoming a low-cost, green methodology for real time online prediction and analysis..

BACKGROUND

Nowadays, the quantity of data that is created every day is growing fast. Moreover, it was estimated that 2007 was the first year in which it was not possible to store all the data that we are producing. This massive amount of data opens new challenging discovery tasks, and the goal of this paper is to discuss them.

Data stream real time analytics (Masud, 2013) are needed to manage the data currently generated, at an ever increasing rate, from such applications as: sensor networks, measurements in network monitoring and traffic management, log records or click-streams in Web exploring, manufacturing processes, call detail records, email, blogging, twitter posts and others. In fact, all data generated can be considered as streaming data or as a snapshot of streaming data, since it is obtained from an interval of time.

In the data stream model, data arrive at high speed, and algorithms that process them must do so under very strict constraints of space and time. Consequently, data streams pose several challenges for data mining algorithm design. First, algorithms must make use of limited resources (time and memory). Second, they must deal with data whose nature or distribution changes over time.

We need to deal with resources in an efficient and low-cost way (Gaber, 2005). In data stream mining, we are interested in three main dimensions:

- Accuracy.
- Amount of space (computer memory) necessary.
- The time required to learn from training examples and to predict.

These dimensions are typically interdependent: adjusting the time and space used by an algorithm can influence accuracy. By storing more precomputed information, such as look up tables, an algorithm can run faster at the expense of space. An algorithm can also run faster by processing less information, either by stopping early or storing less, thus having less data to process. The more time an algorithm has, the more likely it is that accuracy can be increased.

DOI: 10.4018/978-1-4666-5202-6.ch061

MAIN FOCUS

The most important challenges in data stream mining are how to perform low-cost data mining analysis in real time. In evolving data streams we are concerned with

- Evolution of accuracy.
- Probability of false alarms.
- Probability of true detections.
- Average delay time in detection.

Some learning methods do not have change detectors implemented inside, and therefore it may be hard to define ratios of false positives and negatives, and average delay time in detection. In these cases, learning curves may be a useful alternative for observing the evolution of accuracy in changing environments.

The main challenges of an ideal learning method for mining evolving data streams are the following: high accuracy and fast adaption to change, low computational cost in both space and time, theoretical performance guarantees, and minimal number of parameters.

The state-of-the-art methods for classification of evolving data streams are learners based on decision trees (Gama, 2010). A *Hoeffding tree* (Domingos, 2000) is an incremental, anytime decision tree induction algorithm that is capable of learning from massive data streams, assuming that the distribution generating examples does not change over time. Hoeffding trees exploit the fact that a small sample can often be enough to choose an optimal splitting attribute. This idea is supported mathematically by the Hoeffding bound, which quantifies the number of observations (in our case, examples) needed to estimate some statistics within a prescribed precision (in our case, the information gain of an attribute). A theoretically appealing feature of Hoeffding Trees not shared by many other incremental decision tree learners is that it has sound theoretical guarantees of performance. Using the Hoeffding bound one can show that the output of a Hoeffding tree is asymptotically nearly identical to that of a non-incremental learner using infinitely many examples. See (Domingos, 2000) for details.

Bagging and Boosting are ensemble methods used to improve the accuracy of classifier methods. Non-streaming bagging builds a set of M base models, training each model with a bootstrap sample of size N created by drawing random samples with replacement from the original training set. Each base model's training set contains each of the original training example K times where P(K=k) follows a binomial distribution. This binomial distribution for large values of N tends to a Poisson(1) distribution, where Poisson(1)= exp(-1)/k!. Using this fact, Oza and Russell (Oza, 2001) proposed *Online Boosting* and *Online Bagging*. Online Bagging is a powerful streaming method that instead of sampling with replacement, gives each example a weight according to Poisson(1). In (Bifet, 2009) two new state-of-the-art bagging methods were presented: ASHT Bagging using trees of different sizes, and ADWIN (Bifet, 2010) Bagging using a change detector to decide when to discard underperforming ensemble members.

FUTURE TRENDS

The main challenges in the future will be how to deal with this task using thousands of computers using distributed computation.

A way to speed up the mining of streaming learners is to distribute the training process onto several machines. Hadoop MapReduce is a programming model and software framework for writing applications that rapidly process vast amounts of data in parallel on large clusters of compute nodes. A MapReduce job divides the input dataset into independent subsets that are processed by map tasks in parallel. This step of mapping is then followed by a step of reducing tasks. These reduce tasks use the output of the maps to obtain the final result of the job.

Apache S4 (Neumeyer, 2010) is a platform for processing continuous data streams. S4 is designed