AGILE: A General Approach to Detect Transitions in Evolving Data Streams

Jiong Yang
Department of EECS
Case Western Reserve University
jiong@eecs.cwru.edu

Wei Wang
Department of Computer Science
University of North Carolina at Chapel Hill
weiwang@cs.unc.edu

Abstract

In many applications such as e-commerce, system diagnosis and telecommunication services, data arrives in streams at a high speed. It is common that the underlying process generating the stream may change over time, either as a result of the fundamental evolution or in response to some external stimulus. Detecting these changes is a very challenging problem of great practical importance. The overall volume of the stream usually far exceeds the available main memory and access to the data stream is typically performed via a linear scan in ascending order of the indices of the records. In this paper, we propose a novel approach, AGILE, to monitor streaming data and to detect distinguishable transitions of the underlying processes. AGILE has many advantages over the traditional Hidden Markov Model, e.g., AGILE only requires one scan of the data.

Keywords: Stream processing, Transition detection, Variable memory Markov model, Emission tree

1 Introduction

Analyzing stream data has drawn increasing interests recently. A data stream is a long ordered sequence of records that do not support efficient random access. Access to the data stream is typically performed via a linear scan in ascending order of the indices of the records. The overall data volume usually far exceeds the available memory. Only a summary of the past data and/or a recently observed portion of the stream may be able to be stored in the memory. This poses an exceptional challenge to many existing data mining techniques that require multiple scans and/or random access of the data. This, in part, makes many previous algorithms assume that (1) the data was generated from a stationary distribution and (2) any portion of the stream can serve as a good representative of the entire stream. These assumptions, in general, do not hold in many of today's applications because the underlying process that generates the stream may change over time, either as a result of the fundamental evolution or in response to some external stimulus. Applying algorithms designed for stationary process to an evolving stream may lead to invalid conclusions.

Our focus in this paper is on detecting transitions of underlying processes of a data stream, which has great importance in many applications, e.g., web page recommendation, intrusion detection, network traffic analysis, etc. Many of these applications build their models upon some statistics (e.g., count, conditional dependency, etc.) of the data assuming a stationary process. Successful transition detection provides an opportunity to apply many existing techniques directly to streaming data without the concern of deriving invalid models. A model is always valid before a transition of the underlying process occurs. The detection of a transition can be regarded as a signal to trigger verification and/or revision of the existing model. We shall show later in the experimental result that our proposed method can successfully support, for example, the maintenance of a decision tree on an evolving stream.

The most common statistical approach to model an evolving sequence is the Hidden Markov Model (HMM). The learning of a Hidden Markov Model involves constructing the structure (e.g., the set of states and their connections) and estimating the parameters (e.g. transition and emission probabilities). In practice, the structure and topology of the HMM and the transition path are often predetermined, and, the memory length is limited to the first order to make the learning process a feasible task, which make the success of HMM vulnerable to erroneous settings. Furthermore, this learning process typically requires multiple scans of the entire sequence before the transition points can be determined, which makes it not practical for monitoring streaming data. Mining massive stream data has been an active topic in recent years. Much work was developed. Due to the space limitations, we omit the related work discussion, interested readers please refer to [5].

In this paper, we devise a novel approach, AGILE, as an alternative to the HMM. AGILE employs the variable memory Markov (VMM) model [2] to represent each underlying process, which has been proven to be very powerful in capturing longer dependencies and higher order statistics [1, 3] and can be learned with a single scan of the sequence. These advantages make the VMM model an ideal choice for modelling each underlying process of the stream. Moreover, with the VMM model on hand, the potential change can be detected easily by checking whether the most recent observations in the stream still *comply with* the current VMM model. One way to perform the compliance check is to estimate the probability of generating the (recently observed portion of the) sequence using the VMM model. If this probability is significantly higher than the probability under a memoryless random process, we may think that the underlying process remains static. Otherwise, it may signal a potential transition of the underlying processes. When the underlying process remains static, the recent observations will be used to perfect the VMM model; while a new VMM model will be built to reflect the current state if the underlying process changes. This new VMM model will be continuously refined until the next transition point. To facilitate the learning and maintenance of each VMM model, a novel variation of suffix tree, **emission tree**, is utilized to organize the emission probabilities. The emission tree is succinct, highly adaptable, and easy to retrieve and update.

When monitoring a data stream, AGILE uses a buffer to hold the most recently observed portion of the stream and employs an emission tree to maintain the emission probabilities of the current underlying process, which is continuously enriched as AGILE scans through the stream until a transition is detected. Comparing to the traditional HMM approach, AGILE has many important advantages that are crucial to monitoring high speed data streams. (1) The state path, topology, and complexity of each underlying process do not need to be predetermined and can be automatically learned from the data stream. (2) The transition points can be identified promptly without tracing back the history of the stream. (3) The computational complexity to process each record is linearly proportional to the memory length of the underlying process and all computations are performed in memory.

The remainder of this paper is organized as follows. Section 2 describes the problem definition while the basic algorithm is discussed in Section 3. We present the empirical results in Section 4. Finally, we draw conclusions in Section 5.

2 Problem Definition

We now formalize the problem studied in this paper. A stream is an ordered sequence of records $r_1, r_2, \ldots$. For the sake of brevity, we assume that each record r_i is represented by a symbol in a finite alphabet $\Im = \{s_1, s_2, \ldots, s_n\}$. The stream is assumed to be generated from a number of *distinguishable* (but unknown) stationary stochastic processes with some upperbound on the transition rate. That is, the stream can be viewed as the concatenation of an (unknown) number of continuous fragments, each of which is generated by a single process with an (unknown) duration greater than some (unknown) constant ℓ. The concept of distinguishability is introduced to ensure the stability of each underlying process and to enable reliable discrimination between different processes.

The emission probabilities of a general stationary stochastic process specify the conditional probability distribution of the next record given the preceding segments. The probability of generating a segment $r_1 r_2 \ldots r_l$ from a stochastic process π can be calculated as $P^\pi(r_1 r_2 \ldots r_l) = P^\pi(r_1) \times P^\pi(r_2|r_1) \times \cdots \times P^\pi(r_l|r_1 \ldots r_{l-1})$, where $P^\pi(r_i|r_1 \ldots r_{i-1})$ is the probability of generating r_i right after the segment $r_1 \ldots r_{i-1}$. This probability can be used to infer whether an observed segment was generated from a particular underlying process. If the probability $P^\pi(r_1 r_2 \ldots r_l)$ is considerably higher than the probability $P^r(r_1 r_2 \ldots r_l)$ of generating the same segment from a memoryless random process, we may conclude with fairly high confidence that π is the underlying process. In this case, we also say that the segment $r_1 r_2 \ldots r_l$ *complies* with π.

Definition 2.1 *A segment $r_1 r_2 \ldots r_l$* **complies** *with a stationary stochastic process π iff $\frac{P^\pi(r_1 r_2 \ldots r_l)}{P^r(r_1 r_2 \ldots r_l)} > c$ where $c > 1$ is a constant real number*[1].

Under the variable memory Markov (VMM) model, the conditional probability $P^\pi(r_i|r_1 \ldots r_{i-1})$ is equal to (or can be well approximated by) $P^\pi(r_i|r_{i-j} \ldots r_{i-1})$ where j $(0 < j < i)$ is called the *memory length* of the segment $r_1 \ldots r_{i-1}$. In general, the memory length is short (comparing to the length of the segment) and may vary for different segments. The probability $P^\pi(r_1 r_2 \ldots r_l)$ becomes $\Pi_{i=1}^{l} P^\pi(r_i|r_{i-j_i} \ldots r_{i-1})$ where j_i is the memory length of $r_1 \ldots r_{i-1}$. It has been demonstrated [1, 2, 3] that the VMM model is a very powerful model to capture stationary stochastic process.

Two VMM models are said to be **distinguishable** if there exists a segment longer than a certain threshold ℓ such that it complies to only one of these two VMM models. A transition between two VMM models is called a **distinguishable transition** if these two models are distinguishable. Given a stream, our goal is to detect each distinguishable transition of the underlying processes at the earliest possible moment as the stream is parsed.

3 AGILE

In this paper, we propose a novel algorithm, AGILE, to promptly detect the transition of the underlying processes. The emission tree is employed to capture and organize the emission probabilities of each underlying VMM model. The emission tree is similar as the probabilistic tree presented in [4] with one difference. Since the data stream arrives continuously, the emission tree needs to grow with new data. There are two types of nodes in an emission tree. One is the matured nodes where a large amount of evidences are collected so the probability distribution captured in these matured nodes is stable and will not be updated. When new data arrives, the unmatured nodes will be updated to include more evidences for the probability distribution. Figure 2 shows an example of emission tree. Readers may refer to [5] for the detailed description of the emission tree.

3.1 Overview

Figure 1 is the flowchart of major steps taken by AGILE as the data stream is being parsed. At the beginning, a buffer is allocated to track the recently observed portion of the stream and a VMM model is initialized. Since only *mature* node(s) in the emission tree will be used for compliance verification, AGILE first goes through a short *warm-up* stage until the root node matures. In this warm-up stage, every time a new record arrives, it will be put in the buffer and will be used to train the VMM model. Once the root node matures, AGILE proceeds to the *verification and training* stage where, in addition to training the VMM model, the buffered segment will also be used to check for potential transition. Every time a new record arrives, it will be put in the buffer and the most obsolete record will be replaced if the buffer is full. (Records in the buffer are stored in the chronological order.) For example, after the 41st record is read in, the buffer content becomes $babbaabbba$ as shown in Figure 2 where the most obsolete record (the 31st record, a, in the example) is discarded. The buffered stream is then examined to see whether it still complies with the current VMM model. If so, it will be used to further train the current VMM model. Otherwise, the conflict may signal a transition of the underlying process. The buffer is purged by removing previous records that show compliance to the obsolete VMM model. A new VMM model is then initiated based on the remaining records in the buffer. AGILE will then enters the warm-up stage again.

[1] In theory, c can be any real number greater than 1. We choose c to be 2 in this paper. A study on the optimal range of c is in Section 5.2.2.

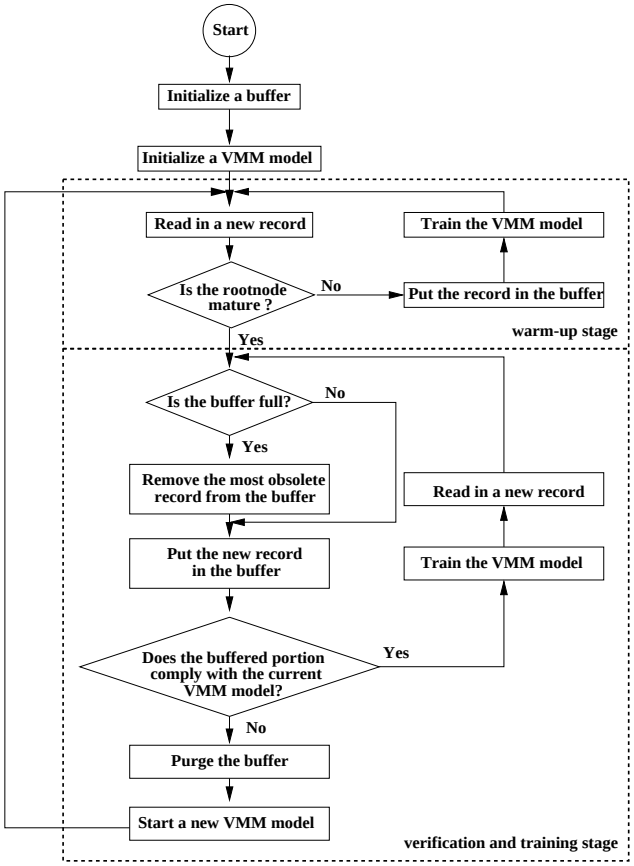

Figure 1. The Flowchart of AGILE

3.2 Compliance Verification

Without loss of generality, let's assume that the buffer size is much larger than the memory length of the underlying process of the stream and the buffer currently holds records $r_1 \dots r_l$. By definition, the compliance of $r_1 \dots r_l$ with the current VMM model, say π, is

$$\frac{P^\pi(r_1 \dots r_l)}{P^r(r_1 \dots r_l)} = \frac{\Pi_{i=1}^{l} P^\pi(r_i | r_{i-j_i} \dots r_{i-1})}{P^r(r_1 \dots r_l)}$$

where j_i ($1 \leq j_i < i$) is the memory length of the segment $r_1 \dots r_{i-1}$ in the VMM model π, for $i = 1, \dots, l$. In fact, $r_{i-j_i} \dots r_{i-1}$ should be the longest suffix of $r_1 \dots r_{i-1}$ which labels a mature node in the emission tree. This node can be located by, starting from the root of the emission tree, traversing along the path $root \rightsquigarrow r_{i-1} \rightsquigarrow r_{i-2} \rightsquigarrow \cdots \rightsquigarrow r_1$ to the furthest mature node. The emission probability of r_i in the emission probability table at this node is the value of $P^\pi(r_i | r_{i-j_i} \dots r_{i-1})$.

The computational complexity is $O(l \times \min\{l, h\})$ where l and h are the buffer size and the memory length of the VMM model (i.e., the height of the emission tree), respectively.

3.3 Purging the Buffer

Once a transition is detected, the buffer needs to be purged to ensure that records generated by the previous underlying process will not interfere the learning of the new model. All records except the most recent record are discarded.

3.4 Initializing a VMM model

This step is invoked in one of the two following conditions: when we start to monitor a new stream and when a transition of the underlying process is detected. Before any information is injected, the initial status of a VMM model is represented by a degenerate emission tree with a single root node. In the case when a new stream starts to be monitored, the user may also choose to construct an emission tree to reflect the *a priori* knowledge of the emission probabilities as an alternative initial VMM model.

3.5 Training the VMM Model

Again, let's assume that the buffer holds records $r_1 \dots r_l$ where r_l is the most recently observed symbol. We need to update the information related to r_l at the following nodes: the root, node r_{l-1}, node $r_{l-2}r_{l-1}$, ..., and node $r_{l-j} \dots r_{l-1}$, where j is the memory length of $r_1 \dots r_{l-1}$. These nodes all locate on a single path, referred to as the **updating path**, in the emission tree and can be accessed by traversing down along the branch $root \rightsquigarrow r_{l-1} \rightsquigarrow r_{l-2} \rightsquigarrow \cdots \rightsquigarrow r_{l-j}$. For each node $r_{l-k} \dots r_{l-2}r_{l-1}$ ($k = 1, 2, \dots, j$), the counter $N(r_{l-k} \dots r_{l-2}r_{l-1}r_l)$ is incremented by 1 and the emission probability entries (if immature) are also updated accordingly.

$$N_{new}(r_{l-k} \dots r_{l-2}r_{l-1}r_l) = N_{old}(r_{l-k} \dots r_{l-2}r_{l-1}r_l) + 1$$

$$P_{new}(r_l | r_{l-k} \dots r_{l-2}r_{l-1}) = P_{old}(r_l | r_{l-k} \dots r_{l-2}r_{l-1}) \times \frac{N_{new}(r_{l-k} \dots r_{l-2}r_{l-1}r_l)}{N_{new}(r_{l-k} \dots r_{l-2}r_{l-1}r_l) - 1}$$

$$\forall y \neq r_l \quad N_{new}(r_{l-k} \dots r_{l-2}r_{l-1}y) = N_{old}(r_{l-k} \dots r_{l-2}r_{l-1}y)$$

$$P_{new}(y | r_{l-k} \dots r_{l-2}r_{l-1}) = \frac{N_{new}(r_{l-k} \dots r_{l-2}r_{l-1}y) \times P_{old}(y | r_{l-k} \dots r_{l-2}r_{l-1})}{N_{new}(r_{l-k} \dots r_{l-2}r_{l-1}y) + P_{old}(y | r_{l-k} \dots r_{l-2}r_{l-1})}$$

If the updated value of $N(r_{l-k} \dots r_{l-2}r_{l-1}r_l)$ is equal to α (i.e., the segment $r_{l-k} \dots r_{l-2}\ r_{l-1}r_l$ just becomes significant), then a new node $r_{l-k} \dots r_{l-2}r_{l-1}r_l$ needs to be initiated (to monitor the emission probabilities given $r_{l-k} \dots\ r_{l-2}\ r_{l-1}r_l$ as the preceding segment). Assume that there is still space available in main memory[2]. This new node is inserted as a child of node $r_{l-k+1} \dots r_{l-2}r_{l-1}r_l$ in the emission tree, which may reside on a different branch other than the one we just visited. The parent node of the new node can be located by traversing along the updating path $root \rightsquigarrow r_l \rightsquigarrow r_{l-1} \rightsquigarrow \cdots \rightsquigarrow r_{l-k+1}$ to reach the furthest node.

Due to space limitations, we do not present the formal proof of correctness and several additional improvements on the basic algorithm of AGILE. Interested readers please refer to [5].

[2]If the main memory is full, AGILE needs to adjust the significance threshold α. This scenario is investigated in [5].

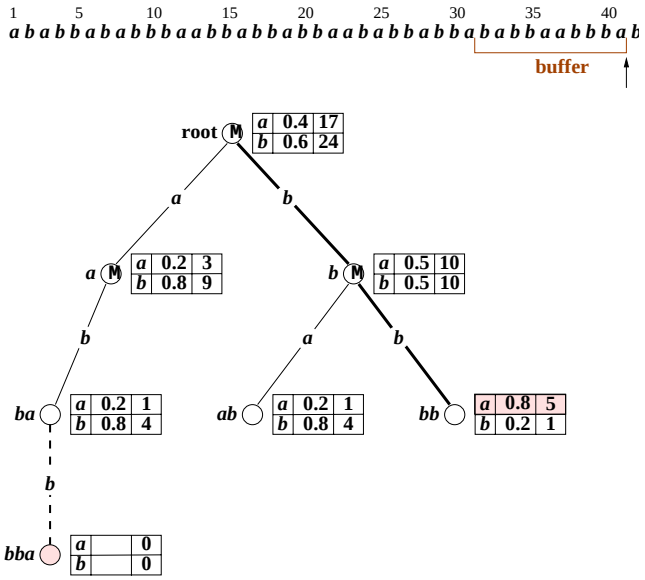

Figure 2. The Emission Tree After Examining the 41st Record

4 Experimental Results

We implement the AGILE algorithm in C. The experiments are performed on a SUN Ultra-Sparc 10 workstation with 1GB main memory and a 500 MHz CPU. In all tests, the initial significance threshold α is set to 25.

We experimented the AGILE system on both real and synthetic data sets. Due to the space limitations, we only show the results on the real data set here. Interested readers please refer to [5] for more detailed empirical results. The real data stream is constructed from a web page access log of an online merchant[3]. This trace log consists of the merchandize web pages that were accessed by users between March 15, 2003 to July 31, 2003. Each user is identified by the IP address. A user session is a list of web pages that accessed by a single user (IP address) within a certain time period, e.g., 1 hour. A stream is generated by concatenating all sessions in chronological order of their starting time, separated by a special symbol, \$. There are over 20 million sessions in the trace and the overall length of the stream is over 400 million merchandize visits. The overall distinct number of merchandizes is 1678. When constructing the emission tree and verifying the compliance of the VMM model, we do not consider any segments that cross the session boundary. This is achieved by purging the buffer every time \$ is reached. Roughly 500MB space is used to store the emission tree.

Three transitions are detected by AGILE. One transition corresponds to a major renovation of the web site, e.g., merchandize re-categorization. The second transition is due to a major revision of recommendation list for each merchandize when it is browsed by a user. We can observe that the recommendation system influences the user's browsing behavior significantly. The last transition reported by AGILE does not correspond to any important "mechanical" change of the web site. After further investigation, we found that the transition is a consequence of some recommendation list becoming obsolete when the NBA season ends. The probability that a user would follow a NBA-related recommendation declines dramatically after the NBA season is over. It is interesting to know that the merchant was not aware of this transition before AGILE detects it. From this test, we can see that AGILE can be very useful to detect unknown yet important changes in a stream. Last but not least, it takes less than 4ms to process each symbol on average.

To further demonstrate the usefulness of AGILE, we compare three alternative methods of online classification based on the web access trace. The objective is to build a decision tree to predict user's purchasing behavior in the immediate future. (1) The decision tree is rebuilt once a month. (2) A sliding window is employed and the decision tree is (partially) rebuilt after every session. (3) AGILE is employed and the decision tree is rebuilt once a transition is detected be AGILE. We found that the third method can produce equally good result[4] as the second method during the entire course of the trace. This suggests that AGILE is powerful to capture every transition that may result in a different decision tree. Further, AGILE can successfully avoid the expensive computation required by the second method. In our experiment, AGILE manages to save more than 70% of the computational cost consumed by the second method. Even though the first method requires less computational effort than the second method, it fails to respond to the transition promptly.

5 Conclusions and Future Work

In this paper, we develop a framework, AGILE, for online detection of changes of streaming data in the form of transitions of the underlying processes. The variable memory Markov model is used to characterize each stationary underlying process and is represented via a compact data structure — emission tree. The potential transition can be promptly detected by checking whether the recently observed records still well comply with the current emission tree. It has been shown that AGILE is very effective and efficient in terms of high accuracy and low storage requirement and computation cost. AGILE guarantees to report any distinguishable transition with an incubation period no longer than ℓ.

References

[1] G. Bejerano and G. Yona. Modeling protein families using probabilistic suffix trees. *Proc. of ACM RECOMB*, pp. 15-24, 1999.

[2] D. Ron, Y. Singer, N. Tishby. The power of amnesia: learning probabilistic automata with variable memory length. *Machine Learning*, vol. 25, no. 2-3, pp. 117-149, 1996.

[3] Y. Seldin, G. Bejerano, and N. Tishby. Unsupervised sequence segmentation by a mixture of switch variable memory Markov sources. *Proc. of ICML*, 2001.

[4] J. Yang and W. Wang. CLUSEQ: efficient and effective sequence clustering. Proceedings of the 19th IEEE International Conference on Data Engineering (ICDE), 2003.

[5] J. Yang and W. Wang. A general approach to detect transitions in in evolving data streams. *UNC Technical Report TR03-023*, 2003.

[3] Due to confidential agreement, we are not able to disclose the identity of the web site.

[4] The quality of the result is assessed by the accuracy of the classification model.