MobiMine: Monitoring the Stock Market from a PDA

Hillol Kargupta, Byung-Hoon Park, Sweta Pittie, Lei Liu, Deepali Kushraj
CSEE Department
University of Maryland Baltimore County
Baltimore, MD 21250, USA
{hillol, bpark1, spitti2, lliu3, deepali1}@cs.umbc.edu

Kakali Sarkar
Agnik LLC
Columbia, MD 21045, USA
kakali@agnik.com

ABSTRACT

This paper describes an experimental mobile data mining system that allows intelligent monitoring of time-critical financial data from a hand-held PDA. It presents the overall system architecture and the philosophy behind the design. It explores one particular aspect of the system—automated construction of personalized focus area that calls for user's attention. This module works using data mining techniques. The paper describes the data mining component of the system that employs a novel Fourier analysis-based approach to efficiently represent, visualize, and communicate decision trees over limited bandwidth wireless networks. The paper also discusses a quadratic programming-based personalization module that runs on the PDAs and the multi-media based user-interfaces. It reports experimental results using an ad hoc peer-to-peer IEEE 802.11 wireless network.

1. INTRODUCTION

Mobile computing devices like PDAs, cell-phones, wearables, and smart cards are playing an increasingly important role in our daily life. The emergence of powerful mobile devices with reasonable computing and storage capacity is ushering an era of advanced data and computationally intensive mobile applications. Monitoring and mining time-critical data streams in a ubiquitous fashion is one such possibility. Financial data monitoring, process control, regulation compliance, and security applications are some possible domains where such ubiquitous mining is very appealing.

This paper reports the development of the *MobiMine*, a distributed data mining environment that allows "intelligent" monitoring of stock market data from mobile devices. It allows the user to store stock portfolio data, manage these portfolios, and monitor relevant portion of the stock market. It facilitates the monitoring process by identifying the interestingly behaving stocks and detecting their causal relationship with different features characterizing the stocks. MobiMine is not a market forecasting system. It is neither a traditional system for stock selection and portfolio management. Instead it is designed for drawing the attention of the user to time critical "interesting" emerging characteristics in the stock market.

This paper particularly explores one aspect of the system—construction of a focus area from the incoming voluminous data sets. The focus area is comprised of a WatchList and its context. The WatchList is a set of "interestingly behaving" stocks. A stock may be considered interesting for many reasons. For example, if the HP stock is rallying and the user is interested in the large-capital technology sector then it is an "interestingly behaving" stock for the user. The context of the WatchList captures the causal relationship between the interesting behavior of a stock and other influencing factors. For example, let us say HP exceeds its quarterly earning expectation by a large amount triggering a rally. Now if HP has been influencing the Compaq stock in the current market then it is a time-critical useful information for an user interested in large-capital technology stocks. The MobiMine system monitors the stock market data stream collected from different financial sites on the web and constructs causal relationships among these factors and the interesting behavior of a stock. The focus area is also a function of time. So the objective is to quickly identify the personalized focus area of the user at a given time using different online data stream mining techniques that are suitable for mobile applications.

Existing mobile portfolio management and other related software primarily rely on manual input to create the watch list of stocks that defines the user's focus area. Manual construction of the focus area is particularly cumbersome for mobile devices since the user cannot be expected to watch the market continuously and perform difficult time consuming data analysis tasks while on the move. So construction of the focus area seems to be an ideal mobile application for the data mining technology.

Mining data streams in a mobile environment [12; 17] offers several unique challenges:

1. handling the continuous flow of incoming data,
2. efficiently representing and communicating the data mining models over the wireless network with limited bandwidth,
3. visualizing in a small screen, developing alternate multimedia-based human-computer interaction, and
4. minimizing power consumption.

This paper considers the first three of the issues listed above. Although the system supports the use of different data mining algorithms this particular paper reports a hybrid technique involving a Fourier representation of decision trees and quadratic programming. It also presents several experimental results documenting the resource consumption characteristics of the system. The proposed system is designed to

handle multiple data sources; however this paper restricts the discussion to applications with single data source and multiple mobile clients.
Section 2 describes the architecture of the MobiMine system. Section 3 presents a novel Fourier analysis and decision tree-based system for mining the data stream. Section 4 presents the experimental results. Section 5 concludes this paper.

2. THE MOBIMINE SYSTEM

The MobiMine is a PDA-based application for managing stock portfolios and monitoring the continuous stream of stock market data. This section presents an overview of the system architecture and the functionalities of the different modules.
The MobiMine is a client-server application. The clients, running on mobile devices like hand-held PDAs and cell-phones, monitor a stream of financial data coming through the MobiMine server. The system is designed for currently available low-bandwidth wireless connections between the client and the server. In addition to different standard portfolio management operations, the MobiMine server and client apply several advanced data mining techniques to offer the user a variety of different tools to monitor the stock market at any time from any where.
The main functionalities of the MobiMine are listed in the following:

1. Portfolio Management and Stock Tickers: Standard book-keeping operations on stock portfolios including stock tickers to keep an eye on the performance of the stocks in the portfolio.

2. FocusArea: Stock market data is often overwhelming. It is very difficult to keep track of all the developments in the market. Even for a full-time professional following the developments all the time is challenging. It is undoubtedly more difficult for a mobile user who is likely to be busy with other things. The MobiMine system offers a unique way to monitor changes in the market data by selecting a subset of the events that is more "interesting" to the user. This is called the FocusArea of the user. It is a time varying feature and it is currently designed to support the following functionalities:

   (a) WatchList: The system applies different measures to assign a score to every stock under observation. The score is an indication of the "interestingness" of the stock. A relatively higher score corresponds to a more interesting score. A selected bunch of "interesting" stocks goes through a personalization module in the client device before it is presented to the user in the form of a WatchList. Further details of this module will be discussed later.

   (b) Context Module: This module offers a collection of different services for better understanding of the time-critical dynamics of the market. The main interesting components are,

      i. StockConnection Module: This module allows the user to graphically visualize the "influence" of the currently "active" stocks on the user's portfolio. This module detects the highly active stocks in the market and presents the causal relationship between these and the stocks in user's portfolio, if any. The objective is to give the user a high level qualitative idea about the possible influence on the portfolio stocks by the emerging market dynamics.

      ii. StockNuggets Module: The MobiMine Server continuously processes a data stream defined by a large number of stock features (fundamentals, technical features, evaluation of a large number of well-known portfolio managers). This module applies online clustering algorithms on the active stocks and the stocks that are usually influenced by them (excluding the stocks in the user's portfolio) in order to identify similarly behaving stocks in a specific sector.

   The StockConnection module tries to detect the effect of the market activity on user's portfolio. On the other hand, the StockNuggets module offers an advanced stock-screener-like service that is restricted to only time-critical emerging behavior of stocks.

   (c) Reporting Module: This module supports a multi-media based reporting system. It can be invoked from all the interfaces of the system. It allows the user to watch different visualization modules and record audio clips. The interface can also invoke the e-mail system for enclosing the audio clips and reports.

2.1 MobiMine: What It is Not

A large body of work exists that addresses different aspects of stock forecasting [1; 2; 7; 18; 20; 32] and selection [6; 16] problem. The MobiMine is fundamentally different from the existing systems for stock forecasting and selection. First of all, it is different on the basis of philosophical point of view. In a traditional stock selection or portfolio management system the user initiates the session. User outlines some preferences and then the system looks for a set of stocks that satisfy the constraints and maximizes some objective function (e.g. maximizing return, minimizing risk). The MobiMine does not do that. Instead it initiates an action, triggered by some activities in the market. The goal is to draw user's attention to possibly time-critical information. For example, if the Intel stock is under-priced but its long time outlook looks very good then a good stock selection system is likely to detect Intel as a good buy. However, the MobiMine is unlikely to pick Intel in the WatchList unless Intel stock happens to be highly active in the market and it fits with user's personal style of investment. The Context detection module is also unlikely to show Intel in its radar screen unless Intel happens to be highly influenced by some of the highly active stocks in the market. This difference in the design objective is mainly based on our belief that mobile data mining systems are likely to be appropriate only for time-critical data. If the data is not changing right now, probably you can wait and you do not need to keep an eye on the stock price while you are having a lunch with your colleagues.
Most of the stock selection systems are based on predictive models. There exists a large body of work that approaches

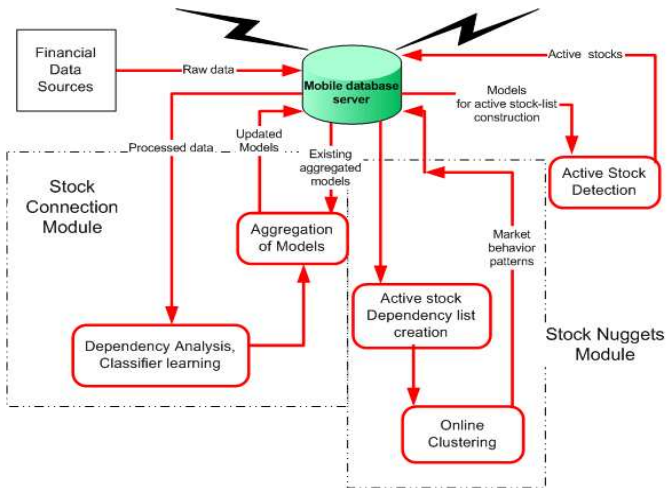

Figure 1: The architecture of the MobiMine Server.

this problem using statistical techniques, neural networks, genetic algorithms, and other techniques. The MobiMine in its current design does not perform any kind of prediction of future behavior. The StockConnection module presents the influence diagram based on the current data. It simply presents how the different companies have been influencing the portfolio stocks in the recent past. As we noted earlier, the StockNuggets module is a clustering-based screener for the active stocks and those (excluding the portfolio stocks) strongly influenced by the active stocks. This is another major distinction between the traditional stock selection systems and the MobiMine. In short, the MobiMine system is designed based on a minimalist principle—anything that can wait should wait and therefore the MobiMine does not need to support it. It should only support functionalities required for time-critical events.
The following sections discuss the architectures of the server and the client.

2.2 The MobiMine Server Architecture

The MobiMine server collects the stock market data from different sources in the web and processes it on a regular basis. It employs several data mining techniques to sift through the data. It interacts with multiple mobile clients in order to offer personalized services supported by advanced data mining services. Figure 1 shows the general structure of the server. Different conceptual aspects of the server functionalities are discussed in the following.

2.2.1 Data Collection and Storage

This module of the MobiMine server is comprised of the mobile database server and financial data sources in the web. It collects publicly available stock market data from different financial web sites and stores some critical subsets needed for maintaining the data mining models. The system is designed to handle data feeds in XML format. However the experiments reported in this paper downloads the data in flat files and stores the critical data in a mobile database server.
The data collection module is a discrete event system that downloads financial data at a certain frequency. The frequency can be controlled by setting the corresponding parameter. Usually the data is stored for a window period. At every time interval this module downloads a new block of data. Each block of data contains about one hundred different features (e.g. the prices of the stocks, yearly range, day's range, change percentage, pics and pans by different reputed portfolio managers; various ratios like P/E ratio, sales/earnings ratio, book value, and so on).

2.2.2 Selection of Active Stocks

The MobiMine server detects a set of currently "active stocks". A stock is called active in this application by computing different common statistics (e.g. percentage change, absolute change, volume of transactions and multiple rule driven scores based on quarterly report disclosure, relevant breaking news). The rule driven score does not always reflect the current behavior of the stock; instead it tries to assign a score of activity based on common anticipations.
A set of stocks is selected based on this score value. This list is sent to the client in order to create its personalized version—the WatchList.

2.2.3 Data Mining

The MobiMine makes use of a collection of online mining techniques including several statistical algorithms, clustering [15], Bayesian nets [10], and decision trees [4; 25; 26]. These techniques are primarily used in the Context component. The StockConnection module uses online statistical, Fourier spectrum-based decision tree and Bayesian learning techniques for detecting the interaction among the active stocks and the portfolio. The StockNuggets module applies a collection of different online clustering algorithms for identifying "interesting" stocks that are influenced by the currently active stocks. In this paper we consider only the Fourier representation-based decision tree learning approach and its unique appeal for mobile applications. We discuss this aspect in details later in this paper.

2.3 The MobiMine Client Architecture

Figure 2 shows the client architecture of the MobiMine. This section describes the client architecture and the user interface. The main components of the client system are described in the following.

2.3.1 Portfolio Management

The MobiMine client offers several utilities to manage stock portfolios and monitor the trend of the stock market. Portfolio data (e.g. stocks, purchased price, quantity of the stocks purchased, stock buying date) and the user's preference related information (e.g. personal acceptable level of risk) are stored in a Micro version of Point Base mobile database. Users can open one of his/her portfolios to review, edit or delete the stored information. Access to this information is controlled by a password. Figure 3 shows the main screen of the client of the MobiMine. It also shows the portfolio editing interface.
The interface has two tickers. One of them is for keeping an eye on the stocks in the user's portfolio. The other one is the WatchList. The latter will be discussed in the next section. The tickers are fairly standard. They show the current stock price and the net change in stock price since the beginning of the trading on that day.

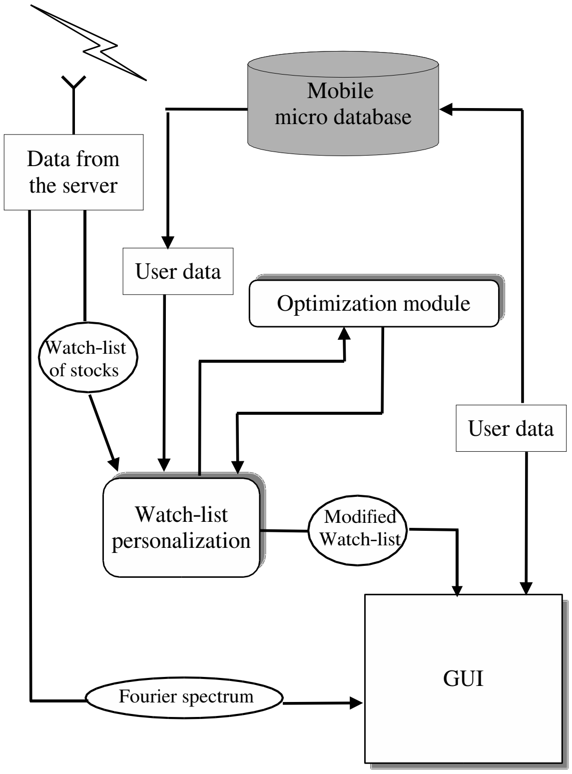

Figure 2: Data Flow in the client system.

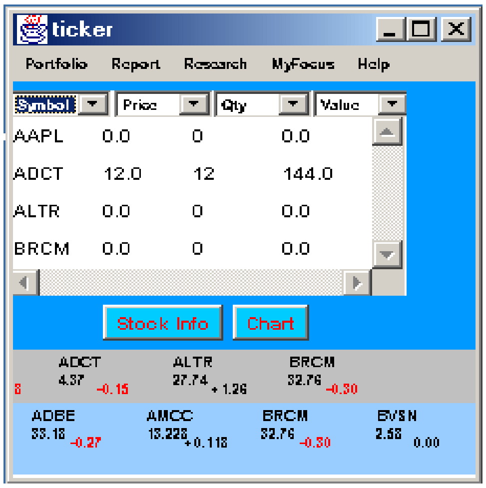

Figure 3: The main screen for Stock Miner

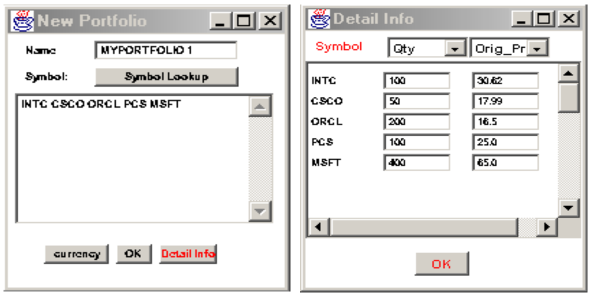

Figure 4: (Left) Screen for creating a new portfolio. (Right) Screen for putting the details of the stocks in the portfolio.

The user can also create a number of portfolios. The interface for creating a new portfolio is shown in Figure 4(Left). Details about the stocks can be added by clicking on the *Detail Info* button which takes the user to the screen shown in Figure 4(Right).

2.3.2 Construction of the WatchList

The bottom-most ticker in Figure 3 shows the WatchList of the MobiMine. It is constructed based on two levels of processing. The first level of processing makes use of a scoring function computed based on several stock features that identify the most active stocks in the market. This is performed at the server. However, the user may not be interested in every active stock in the market. For example, if the user is only interested in low risk investment, a flurry of activity in the high risk volatile stocks may not be interesting to the user. Therefore, such stocks should not be in the WatchList of this user. This calls for personalization and the second level of WatchList processing addresses that. The MobiMine uses a quadratic programming-based optimization algorithm for selecting a subset of the WatchList constructed by the server. This subset of active stocks defines the final WatchList at the client. We should also note that the user can always edit the WatchList. Using this feature the user can explicitly ask the system to keep an eye on a specific stock. Similarly the user can also delete a specific stock from the list. This step is performed in the client because the personalization is likely to vary from user to user and therefore a large number of users can easily overwhelm the server if the personalization module is run on the server.

The current implementation of MobiMine offers two kind of personalization. One is based on straightforward input from the user regarding his/her desire to watch one or more specific stocks. The other is based on user's acceptable level of risk. Different users may have different investing styles. User's acceptable level of risk and the desired level of return of investment are some of the key distinguishing features. The interestingness of a stock depends on these parameters. The MobiMine Client performs a Markowitz-style [22] optimization in order to personalize the list of interesting stocks for every user. Let r_i be the return of investment for the i-th stock estimated over the period of observation and α_i be its normalized weight that represent the degree of importance. So the expected return from investing user's attention (equating attention to virtual investment) to a set of stocks $z = \mathbf{r}^T\alpha$ where $\mathbf{r}$ is the return vector for N stocks and α is the normalized weight vector comprised of the α_i-s. Let Q be the corresponding covariance matrix of the return. So minimizing the risk and maximizing the reward imply minimization of the following objective function:

$$\mathbf{r}^T\alpha - k\alpha^TQ\alpha$$

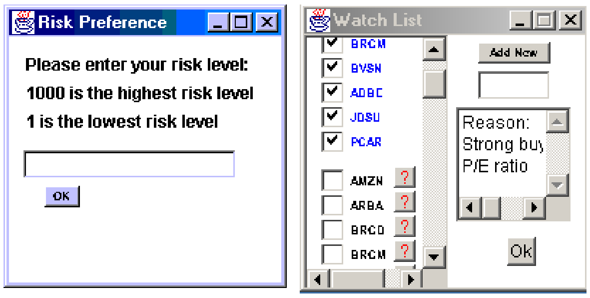

Figure 5: (Left) Screen for risk level acceptable by the user. (Right) The screen for editing the WatchList.

subjected to the following constraints,

$$\sum_{i=1}^{N} \alpha_i = 1$$

$$\alpha_i \geq 0$$

where k is a user-chosen parameter that reflects user's degree of aversion from risk and N is the total number of stocks under observation. We can obtain the α_i-s by solving the above quadratic programming problem. The α-s give us the weights of the stocks. The final watch list is comprised of the stocks that receive high weights (α_i-s) from this optimization module.

Figure 5(Left) shows the module for editing k; it can be set in the range of 1—1000, with 1 corresponding to the lowest risk level while 1000 corresponding to the highest risk level. The user is allowed to input his/her level of risk tolerance.

Figure 5(Right) shows the WatchList interface. There are two groups of stocks listed on this interface. The top part of the list is explicitly created by the user. The bottom part is created by the automated personalization module. As mentioned earlier, the user can delete, add, or edit any member of the WatchList and make it further personalized. The Context module allows the user to explore the underlying dynamics of the market. The following section describes this module.

3. PORTFOLIO AND MARKET ACTIVITIES

Understanding the interaction between a portfolio and the market is a non-trivial problem. It often requires extensive data analysis, in depth understanding of the driving factors in the market, and years of experience in the stock market. In fact PDA may not be right kind of platform to perform extensive user mediated market and portfolio analysis. The MobiMine does not make any attempt to do that. Its StockConnection module offers short-term dependency among a pair of stocks based on the data from the recent past. This does not tell the user about what will happen to the dependent stock in the near future. It simply suggests the user to keep an eye on a stock since it is statistically correlated or influenced by a currently active stock. The MobiMine employs statistical, Bayesian, and decision tree-based online techniques to detect the dependencies.

In the rest of this paper we explore a decision tree-based approach to develop this understanding from the observed stock market data. We particularly focus on a novel approach to mine data streams that is suitable for mobile platforms connected over a wireless network.

There exist several known techniques to construct incremental decision tree-based models. Some of the earlier efforts include ID4[27], ID5[29], ID5R[30] and ITI[31]. All these systems work using the ID3 style "information gain" measure to select attributes (or, equivalently, decision nodes). They are all designed to incrementally build a decision tree using one training instance at a time by keeping necessary statistics (measure for information gain) at each decision node.

Some of the recent techniques designed for large-scale applications include the Bootstrapping-based BOAT [11] and the Hoeffding (additive Chernoff bound) [13] tree-based VFDT [8] and the CVFDT [14].

An ensemble-based approach is proposed in [9]. It works using a Boosting-based approach to create ensemble of models. Different trees are generated from different blocks of data observed at different time intervals. The ensemble classifier for the stream is defined by a weighted average of the outputs of these trees. Breiman proposed an arcing method as the means to learn from large data sets and stream data [3; 5]. It is also based on adaptive re-sampling. However, it uses unweighted average to build the ensemble classifier.

Recently Street and Kim [28] proposed Streaming Ensemble Algorithm (SEA) that learns an ensemble of decision trees for large-scale classification. SEA maintains a fixed number of classifiers. Once the ensemble becomes full, the k-th classifier C_k is added only if it outperforms any previous C_i in the ensemble. In that case, C_i is removed from the ensemble. Performance is measured using the most current data block.

Both the ensemble-based and incremental techniques in practice generate a collection of decision trees. In order to provide the user with the explanatory market models we need to send the trees to the client devices over the wireless network and present them in a user-friendly format. This poses the following problems:

1. Communication of a large number of decision trees over the low-bandwidth wireless network.

2. Visualization of a large number of decision trees in a small screen mobile device like a PDA.

We need an approach that provides a compressed representation of decision trees and offers a compact but meaningful way to visualize them. An early investigation of these issues and a solution based on Fourier analysis of decision trees was reported elsewhere [17]. This paper adopts that approach and reports the experimental results for a wireless network. The following section presents a brief overview of this approach to represent, aggregate, and visualize decision trees.

3.1 Fourier Transform of Decision Trees

A decision tree is a function that maps the domain members to a range of class labels. Sometimes, it is a symbolic function where features take symbolic (non-numeric) values. However, a symbolic function can be easily converted to a numeric function by simply replacing the symbols with numeric values in a consistent manner. A numeric function-representation of a decision tree may be quite useful. For example, we may be able to aggregate a collection of trees (often produced by ensemble learning techniques) by simply performing basic arithmetic operations (e.g. adding two decision trees, weighted average) in their numeric representations. This numeric representation also helps in visualizing and updating of the decision trees on a regular basis.

Once the tree is converted to a numeric discrete function, we can also apply any appropriate analytical transformation that we want. Fourier transformation is one such possibility and it is an interesting one. Fourier basis offers an additively decomposable representation of a function. In other words, the Fourier representation of a function is a weighted linear combination of the Fourier basis functions. The weights are called Fourier coefficients. The coefficients completely define the representation. Each coefficient is associated with a Fourier basis function that depends on a certain subset of features defining the domain of the data set to be mined. The entire process of converting decision trees to a Fourier Spectrum is detailed in [23]. The next section offers a short introduction to the multi-variate Fourier basis representation.

3.2 A Brief Review of the Fourier Basis

Consider the function space over the set of all ℓ-bit Boolean feature vectors. The Fourier basis set that spans this space is comprised of 2^ℓ basis functions; for the time being let us consider only discrete Boolean Fourier basis. Each Fourier basis function is defined as $\psi_{\mathbf{j}}(\mathbf{x}) = (-1)^{(\mathbf{x}\cdot\mathbf{j})}$. Where $\mathbf{j}$ and $\mathbf{x}$ are binary strings of length ℓ. In other words $\mathbf{j} = (j_1, j_2, \cdots j_\ell)$, $\mathbf{x} = (x_1, x_2, \cdots x_\ell)$ and $\mathbf{j}, \mathbf{x} \in \{0,1\}^\ell$; $\mathbf{x}\cdot\mathbf{j}$ denotes the inner product of $\mathbf{x}$ and $\mathbf{j}$. $\psi_{\mathbf{j}}(\mathbf{x})$ can either be 1 or -1. The string $\mathbf{j}$ is called a *partition*. The *order* of a partition $\mathbf{j}$ is the number of 1-s in $\mathbf{j}$. A Fourier basis function depends on some x_i only when $j_i = 1$. Therefore, a partition can also be viewed as a representation of a certain subset of x_i-s; every unique partition corresponds to a unique subset of x_i-s. If a partition $\mathbf{j}$ has exactly β number of 1-s then we say the partition is of order β since the corresponding Fourier function depends on only those β number of features corresponding to the 1-s in the partition $\mathbf{j}$. A function $f : \mathbf{X}^\ell \rightarrow \Re$, that maps an ℓ-dimensional space of binary strings to a real-valued range, can be written using the Fourier basis functions: $f(\mathbf{x}) = \sum_{\mathbf{j}} w_{\mathbf{j}}\psi_{\mathbf{j}}(\mathbf{x})$; where $w_{\mathbf{j}}$ is the Fourier coefficient corresponding to the partition $\mathbf{j}$; $w_{\mathbf{j}} = \frac{1}{2^\ell}\sum_{\mathbf{x}} f(\mathbf{x})\psi_{\mathbf{j}}(\mathbf{x})$. The Fourier coefficient $w_{\mathbf{j}}$ can be viewed as the relative contribution of the partition $\mathbf{j}$ to the function value of $f(\mathbf{x})$. Therefore, the absolute value of $w_{\mathbf{j}}$ can be used as the "significance" of the corresponding partition $\mathbf{j}$. If the magnitude of some $w_{\mathbf{j}}$ is very small compared to other coefficients then we may consider the $\mathbf{j}$-th partition to be insignificant and neglect its contribution.

As pointed out elsewhere [17; 23; 24], the Fourier representation can be easily extended to the domains where features are non-Boolean. Consider a domain defined by ℓ possibly non-Boolean features where the i-th feature can take λ_i distinct values. Let $\overline{\lambda} = \lambda_1, \lambda_2 \cdots \lambda_\ell$. The generalized Fourier basis function over an $\overline{\lambda}$-ary feature space is defined as,

$$\psi_{\mathbf{j}}^{(\overline{\lambda})}(\mathbf{x}) = \Pi_{m=1}^{l} \exp^{\frac{2\pi i}{\lambda_m} x_m j_m} \tag{1}$$

When $\lambda_1 = \lambda_2 = \cdots = \lambda_\ell = \lambda$ we can write, $\psi_{\mathbf{j}}^{(\lambda)}(\mathbf{x}) = \exp^{\frac{2\pi i}{\lambda}(\mathbf{x}\cdot\mathbf{j})}$. Where $\mathbf{j}$ and $\mathbf{x}$ are λ-ary strings of length ℓ. The energy of a set of Fourier coefficients is the sum of the square of the coefficients in that set.

The Fourier coefficients for non-Boolean domain can be defined as follows:

$$w_{\mathbf{j}} = \Pi_{i=1}^{l} \frac{1}{\lambda_i} \sum_{\mathbf{x}} f(\mathbf{x}) \overline{\psi}_{\mathbf{j}}^{(\overline{\lambda})}(\mathbf{x}) \tag{2}$$

where $\overline{\psi}_{\mathbf{j}}^{(\overline{\lambda})}(\mathbf{x})$ is the complex conjugate of $\psi_{\mathbf{j}}^{(\overline{\lambda})}(\mathbf{x})$.

Since a decision tree is a function defined over an inherently discrete or some discretization of a continuous space, we can compute its Fourier transformation. It turns out that the Fourier representation of a decision tree with bounded depth has some very interesting properties [19; 21; 23]. These observations are discussed in the following section.

3.3 Fourier Spectrum of a Bounded Depth Decision Tree

For most practical applications decision trees have bounded depths. The Fourier spectrum of a bounded depth decision tree has some interesting properties.

1. The Fourier representation of a bounded depth (say k) Boolean decision tree only has a polynomial number of non-zero coefficients; all coefficients corresponding to partitions involving more than k feature variables are zero. The proof is relatively straight forward.

2. If the order of a partition is the number of its defining features then the magnitude of the Fourier coefficients decay exponentially with the order of the corresponding partition; in other words low order coefficients are exponentially more significant than the higher order coefficients. This was proved in [21] for Boolean decision trees. Its counterpart for trees with non-Boolean features can be found elsewhere [23].

These observations suggest that the spectrum of a decision tree can be approximated by computing only a small number of low-order[1] coefficients. So Fourier basis offers an efficient numeric representation of a decision tree in the form of an algebraic function that can be easily stored, communicated, and manipulated. This does not necessarily mean that the Fourier spectrum of a single small tree is always a more compact representation of the tree. However, it offers a more efficient representation particularly for large ensemble of trees.

3.4 Visualization of Decision Trees

The Context module offers an interface to visualize decision trees through their Fourier spectrum. The interface is interactive and it does not require any prior knowledge of Fourier analysis. Figure 6 shows a typical session of this interface. It shows the Fourier coefficients of an ensemble of decision trees. Each rectangular cell on the interface corresponds to each coefficient. Each cell is color-coded according to the magnitude (or significance) of the corresponding Fourier coefficient. When the user clicks on a significant (or brighter) cell, all the features associated with the corresponding Fourier coefficient are displayed, which are essentially the dominant features.

[1]Order of a coefficient is the number of features defining the corresponding partition. Low-order coefficients are the ones for which the orders of the partitions are relatively small.

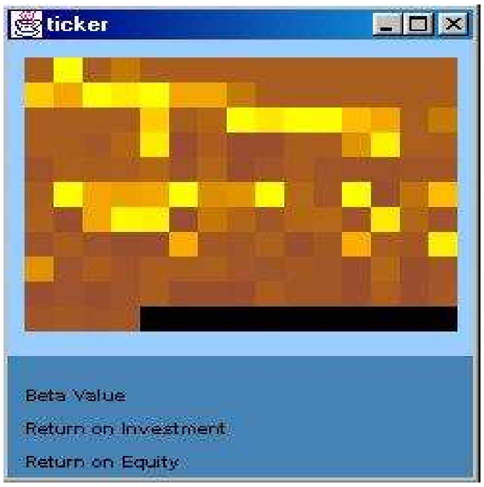

Figure 6: Screen showing influence of the control features. This interface plots the coefficients of the Fourier Spectrum of the decision tree ensemble. Each cell corresponds to a unique coefficient and the cells are color-coded based on their magnitudes. Every cell corresponds to a unique subset of features. Brighter cells correspond to influencing features. The bottom window lists these features.

4. EFFICIENT TRANSMISSION OF DECISION TREES

This section reports the experimental results that demonstrate the benefits of using the Fourier basis for representing decision trees in mobile data mining applications. It notes the storage and communication cost of the conventional ensemble of decision trees and compares it with that of aggregated ensemble in the Fourier representation.

The Fourier spectrum of a tree reflects the properties of the underlying function. The partitions associated with significant coefficients (with relatively large magnitude) identify the different subsets of features that strongly interact with the predicted feature. A weighted-average ensemble of trees in its aggregated Fourier spectrum-form tells us about the most influencing set of features averaged over all the trees. Since the MobiMine system generates these trees from different data blocks observed at different times, the aggregated spectrum identifies the influencing subsets of features over a period of time. The tree-ensembles for different stocks belonging to the same sector (e.g. semiconductor industry) can also be aggregated for finding out the most influencing subsets of features across the sector. We do not necessarily have to use the weighted average scheme. We can simply count the number of significant partitions or just use the magnitude of the coefficients instead of the signed values of the coefficients. In this paper we report results that simply average the coefficients.

The experiments use a stream of Nasdaq 100 stock data. At every time interval the data collection module gathers 100 different rows for 100 different companies. Data is collected every five minutes. For these experiments we used 45 numeric feature values. These features are pre-processed and discretized based on the percentages of difference between two consecutive instances. If the percentage of difference is greater than a certain threshold, the feature value is set to one, otherwise to zero.

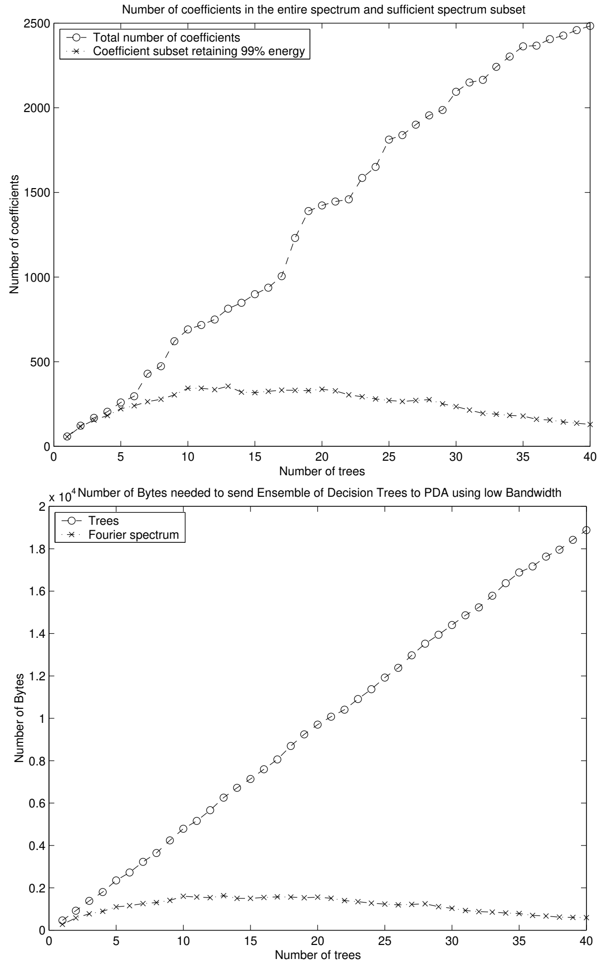

Figure 7: (Top) Variation of the total number of coefficients and significant coefficients in the Fourier spectrum. (Bottom) Byte size of the tree ensemble and its spectrum.

For this experiment, we create an ensemble of decision trees after 65 data updates. The experiments are performed using the "computer technology sector"-stocks in the Nasdaq 100 index. The sector includes 40 companies. We mapped each decision tree to its corresponding Fourier spectrum and measured the following items:

- The total size of both the aggregated Fourier spectrum and the subset of the spectrum that holds most of the significant coefficients.
- The number of bytes required to represent the Fourier spectrum and the original ensemble.
- The time required to send the Fourier spectrum and the original ensemble to a mobile client.

The experiments are conducted in order to find out how the Fourier spectrum-based approach scales up compared to the regular ensemble-based approach as the number of trees increases.

In order to identify the subset of the spectrum that captures the most significant coefficients in each ensemble, we sort the

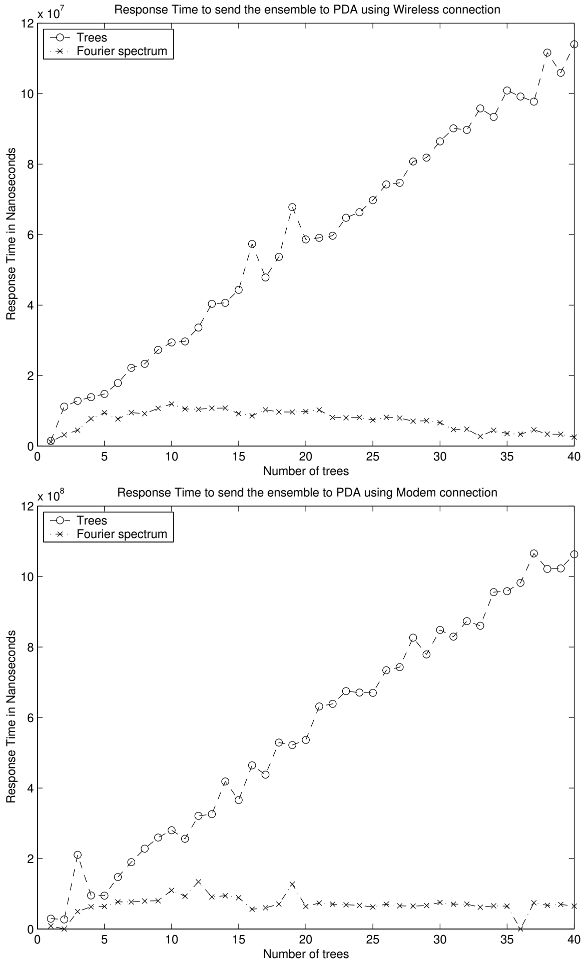

Figure 8: Variation of the response time for transmitting the trees and that of transmitting the *average* Fourier spectrum of the ensemble with respect to an increasing number of trees in the ensemble. (Top) Results using an IEEE 802.11b wireless channel. (Bottom) Results using a 56Kbps modem channel.

Fourier coefficients according to their magnitudes and pick up the subset that contains 99% of the spectrum-energy. As Figure 7(Top) indicates, a small number of Fourier coefficients is sufficient to accurately represent an ensemble regardless of the number of trees it contains. In other words, even for a large ensemble, its Fourier spectrum is very compact.

In order to compare the overheads in storing a Fourier spectrum in the system with that of the original ensemble-based approach, we computed the number of bytes required to represent each model. As clearly represented in Figure 7(Bottom), while the number of bytes for representing original ensemble model increases linearly as the size of the ensemble increases, it increases very little for the Fourier representation.

We also compare the response time needed to communicate these models to the client. We do that by measuring the time required to send the Fourier spectrum and the original ensemble to a Compaq iPaq H3650 over an IEEE 802.11b wireless channel. Figure 8(Top) shows the average result of 50 runs. The graph indicates that we also achieve a significant reduction in transmission time with the Fourier representation. The same set of experiments is conducted over a low bandwidth (56Kbps) dial-up connection. The results are reported in Figure 8(Bottom). The Fourier spectrum-based approach appears to outperform the regular ensemble-approach and the benefit is more pronounced because of the low bandwidth.

5. FUTURE WORK AND CONCLUSIONS

Although mobile computing devices are becoming more accessible and computationally powerful, their usage is still restricted to simple operations like web surfing, checking emails, maintaining personal schedules and taking notes. The limited battery power, restricted visual display area, and the low-bandwidth communication channel are hindering more sophisticated desktop-style applications. However, the ubiquitous aspect of these devices makes them very attractive for many time-critical applications. So we need a new breed of applications for time-critical domains that can live with these resource restrictions.

We believe that data mining can play a significant role in mobile applications. The ubiquitous aspect of mobile devices inherently makes them unsuitable for desktop-style applications that demand time consuming operations, undivided attention from the user, and complicated user interaction involving large amount of information. The small mobile devices need semi-automated programs that the user can rely on for sifting through the data, performing necessary operations, and reporting only the time-critical information. Data mining systems perform similar tasks in desktop environments. They should also be able to help mobile applications. However, there are several issues that we need to address. Mobile environment is fundamentally different from the desktop environment in many aspects. The critical resources are different here. A data mining algorithm that works by minimizing disk I/O may not work well in disk-less mobile devices. An algorithm that requires a single data table downloaded from different sites may not scale in a mobile environment because of limited bandwidth wireless networks. We need data mining techniques that pay careful attention to these different kinds of resources like bandwidth, battery power, and limited display.

This paper took a small step exploring the largely unchartered water of mobile data mining. It presented a high level description of the MobiMine, a personalized mobile data mining system for monitoring the financial stock market. The system sifts through the voluminous stock data and detects "active" stocks that are interesting to the user. It also tries to capture the context of user's stock portfolio by identifying its interaction with the current active stocks in the market. It is not a predictive system for stock forecasting. Rather it is a system that summarizes the market data in some dimensions and helps the user in keeping an eye on the "most interesting" part of the voluminous data stream.

The MobiMine system is designed to scale up for serving a large number of users and the scalability tests are ongoing. To overcome communication bottleneck and difficulty in rendering complex information regarding financial market in a small display area, the system uses several specialized techniques. This paper discusses a novel technique to represent decision trees in Fourier basis. The empirical results strongly indicate that the Fourier spectrum-based representation is appropriate for transmitting the tree-ensembles over a wireless channel. It also appears to be an interesting approach for presenting the information in a small display area. However, the human-computer-interaction aspect needs further studies. The current approach to construct the Fourier spectrum of a decision tree works for categorical features and real-valued features that are discretized before constructing the tree. We are exploring extension of this technique that can handle real-valued data that is dynamically categorized during the construction of the tree. We are also currently exploring different issues such as battery power consumption of the modules, scalability of the server, and different multi-media-based enhancements of the MobiMine system.

Acknowledgments

The authors acknowledge supports from the United States National Science Foundation CAREER award IIS-0093353 and TEDCO, Maryland Technology Development Center. The authors would like to thank Sohel Merchant for his help in developing the system.

6. REFERENCES

[1] A. Azoff. *Neural Network Time Series Forecasting of Financial Markets*. Wiley, New York, 1994.

[2] N. Baba and M. Kozaki. An intelligent forecasting system of stock price using neural networks. In *Proceedings of IJCNN, Baltimore, Maryland*, pages 652–657, Los Alamitos, 1992. IEEE Press.

[3] L. Breiman. Pasting bites together for prediction in large data sets and on-line. Technical report, Statistics Department, University of California at Berkeley, 1997.

[4] L. Breiman, J. H. Freidman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, Belmont, CA, 1984.

[5] L. Breiman. Pasting small votes for classification in large databases and on-line. *Machine Learning*, 36(1–2):85–103, 1999.

[6] J. Campbell, A. Lo, and A. MacKinley. *The Econometrics of Financial Markets*. Princeton University Press, USA, 1997.

[7] S. Cheng. A neural network approach for forecasting and analyzing the price-volume relationship in the taiwan stock market. Master's thesis, National Jow-Tung University, Taiwan, R.O.C, 1994.

[8] P. Domings and G. Hulten. Mining high-speed data streams. In *Proceedings of Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Boston, MA, August, 2000.

[9] W. Fan, S. Stolfo, and J. Zhang. The application of adaboost for distributed, scalable and on-line learning. In *Proceedings of Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Diego, California, 1999.

[10] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2-3):131–163, 1997.

[11] J. Gehrke, V. Ganti, R. Ramakrishnan, and W. Loh. BOAT — optimistic decision tree construction. In *Proceedings of SIGMOD, ACM*, pages 169–180, 1999.

[12] J. Himberg, K. Korpiaho, H. Mannila, J. Tikanmaki, and Toivonen. Time series segmentation for context recognition in mobile devices. In *Proceedings of the IEEE International Conference on Data Mining*, pages 203–210, 2001.

[13] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963.

[14] G. Hulten, Spencer L., and P. Domingos. Mining time-changing data streams. In *Proceedings of the Seventh International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, 2001. ACM Press.

[15] A. Jain and R. Dubes. *Algorithms for clustering data*. Printice Hall, 1988.

[16] G. Jang, F. Lsi, and T. Parng. Intelligent stock trading decision support system using dual adaptive-structure neural networks. *Journal of Information Science Engineering*, 9:271–297, 1993.

[17] H. Kargupta and B. Park. Mining time-critical data stream using the Fourier spectrum of decision trees. In *Proceedings of the IEEE International Conference on Data Mining*, pages 281–288, 2001.

[18] R. Kuo, L. Lee, and C. Lee. Intelligent stock market forecasting system through artificial neural networks and fuzzy delphi. In *Proceedings of World Congress on Neural Networks*, pages 345–350, San Deigo, 1996. INNS Press.

[19] S. Kushilevitz and Y. Mansour. Learning decision rees using Fourier spectrum. In *Proceedings of 23rd Annual ACM Symp. on Theory of Computing*, pages 455–464, 1991.

[20] C. Lee. Intelligent stock market forecasting system through artificial neural networks and fuzzy delphi. Master's thesis, Kaohsiung Polytechnic Institute, Taiwan, R.O.C, 1996.

[21] N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, Fourier transform, and learnability. *Journal of the ACM*, 40:607–620, 1993.

[22] H. Markowitz. *Portfolio Selection: Efficient Diversification of Investments*. Basil Blackwell, Cambridge, 1991.

[23] B. Park. *Knowledge Discovery from Heterogeneous Data Streams Using Fourier Spectrum of Decision Trees*. PhD thesis, Washington State University, 2001.

[24] B. Park, Ayyagari R., and H. Kargupta. A Fourier analysis-based approach to learn classifier from distributed heterogeneous data. In *Proceedings of the First SIAM Internation Conference on Data Mining*, Chicago, US, 2001.

[25] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[26] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kauffman, 1993.

[27] J. C. Schlimmer and R. Granger Jr. Beyond incremental processing: Tracking concept drift. In *AAAI, Vol. 1*, pages 502–507, 1986.

[28] N. W. Street and Y. Kim. A streaming ensemble algorithm (sea) for large-scale classificaiton. In *Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, pages 377–382, 2001.

[29] P. E. Utgoff. ID5: an incremental ID3. In J. Laird, editor, *Proceedings of the Fifth International Conference on Machine Learning*, pages 107–120, San Mateo, CA, 1988. Morgan Kaufmann.

[30] P. E. Utgoff. Incremental induction of decision trees. *Machine Learning*, 4:161–186, 1989.

[31] P. E. Utgoff. An improved algorithm for incremental induction of decision trees. In *Proc. 11th International Conference on Machine Learning*, pages 318–325. Morgan Kaufmann, 1994.

[32] J. Zirilli. *Financial Prediction Using Neural Networks*. International Thomson Computer Press, 1997.