2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Stock Volatility Prediction using Multi-Kernel Learning based Extreme Learning Machine

Feng Wang[1,*], *Member, IEEE*, Zhiyong Zhao[1], Xiaodong Li[2], Fei Yu[1], and Hao Zhang[1]
[1] State Key Laboratory of Software Engineering, Computer School, Wuhan University, Wuhan, China.
[2]Department of Computer Science, City University of Hong Kong, Hong Kong
fengwang@whu.edu.cn

***Abstract*—Stock price volatility prediction is regarded as one of the most attractive and meaningful research issues in financial market. Some existing researches have pointed out that both the prediction accuracy and the prediction speed are the most important facts in the process of stock prediction. In this paper, we focus on the problem of how to design a methodology which can improve prediction accuracy as well speed up prediction process, and propose a multi-kernel learning based extreme learning machine (MKL-ELM) model to enhance the prediction performance. ELM is a fast learning model and has been successfully applied in many research fields. Based on ELM, this MKL-ELM has the benefits of both multiple kernel learning and ELM, which can well balanced the requirements of both prediction accuracy and prediction speed. To validate the performance of the proposed MKL-ELM, we take experiments on HKEx 2001 stock market datasets. The market historical price and the market news are implemented in our MKL-ELM. We Compare our proposed MKL-ELM with Back-Propagation Neural Network(BP-NN), Support Vector Machine(SVM), Basic ELM and K-ELM. Experimental results show that, 1) MKL-ELM, K-ELM and SVM get higher prediction accuracy than BP-NN and B-ELM; 2) Both MKL-ELM and K-ELM can achieve faster prediction speed than SVM in most cases; 3) MKL-ELM has higher prediction accuracy in some cases than K-ELM and SVM.**

Keywords*—stock prices volatility prediction; multiple kernel learning; extreme learning machine; multiple data source integration

I. Introduction

Stock market plays an important role in nowadays financial markets. And it has attracted more and more attention in the past few decades. Many researchers from different fields pay more and more attention on it. With the development of computer science technology, some methodologies based on data mining and machine learning have been introduced to predict the stock price movement automatically.

With the development of algorithmic trading and electronic trading, more and more investors noticed that the stock volatility prediction should not only focus on the prediction accuracy but also the prediction speed. The prediction accuracy is the first requirement for the designed prediction model. Higher prediction accuracy can help the market traders make a better decision. And the fast speed of real-time market transaction asks for a faster prediction speed.

In the past few decades, several approaches, such as neural network and support vector machine, have been widely applied to classification and regression problems. Due to their good performance in the classification and regression fields, they have been applied to predict the stock price volatility successfully. Sureshkumar and Elango[1] use neural works to predict the future stock prices and evaluate their performance statistics. L.Cao and F.Tay[2] make financial forecasting with SVM. In recent years, some improved algorithms or strategies[3][4][5][6] have been applied to improve the performance of the stock price prediction. However, these approaches mainly focus on the prediction accuracy and rarely care about the prediction speed. The prediction speed is also an important fact in stock price volatility prediction, and sometimes even one mini-second lag may cause the market trading strategy invalid in algorithmic trading. As a result of this, how to design a model of stock price volatility prediction which can have good performance on both prediction accuracy and prediction speed is a meaningful and challenging issue in financial market.

On one hand, some works have argued that using multiple market data sources may promote the prediction accuracy rather than using one data source[7]. In recent years, several researchers have engaged in improving the performance of the stock price prediction by combining the historical price with other useful data[1][2]. And some researchers proved that market news can help make better prediction on the accuracy [8][9]. For example, Schumaker and Chen[8] have proposed the stock price prediction by combining the prices with news. Li and Wang[7] have taken the market news and the market prices into account to improve the prediction accuracy. On the other hand, different learning algorithms have been proposed to speed up the learning speed. Extreme learning machine(ELM), which proposed by Huang[10], has been reported to have a great improvement on speed for traditional neural networks, especially for Single-hidden Layer Feedforward Networks(SLFNs). As shown in[10], ELM randomly assigns the input weights and the hidden layer biases instead of fully tuning the parameters in the iteration that the traditional algorithms undertake. ELM could analytically determine the output weights and guarantee the norm of the output weights being the smallest. ELM provides not only the fast speed, but also the accuracy with the smallest output weights. With the development of ELM, several improved methods based on the basic ELM[11][12][13] have been studied and further improved the performance of basic ELM.

In this paper, we propose a multi-kernel learning based extreme learning machine(MKL-ELM) model to enhance the performance of the stock price volatility prediction with some different data sources. Then we evaluate our proposed MKL-

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

ELM method on the HKEx 2001 stock market price dataset and news articles of year 2001 and compare the performance of MKL-ELM with Back-Propagation Neural Network(BP-NN), Support Vector Machine(SVM), basic ELM(B-ELM) and kernel ELM(K-ELM) on the testing dataset. Experimental results show that, compared with the current existing methodologies, MKL-ELM can achieve better performance on the consideration of both prediction accuracy and prediction speed in most cases.

The rest of this paper is organized as follows. Section II gives an introduction of our preliminary work about data preparation. Section III presents the detailed information of our multi kernel learning based extreme learning machine (MKL-ELM) model. Experimental results and discussion are reported in Section IV. The conclusion and future work are given in Section V.

II. Preliminary Work

There are many factors that may have impacts on the stock price volatility, such as historical prices, news about big mergers, bankruptcy of some companies and economic turmoil. The market news and stock market historical prices are widely concerned by both academic researchers and market practitioners. In order to make these data ready for the prediction tasks, some work including the data preprocessing and the data alignment must be done at first. In this section, we will introduce the processes of data preprocessing and the data alignment, at the end of this section, we introduce the models how to combine these multiple data sources.

A. Data Preprocessing

1) The Market Prices: Stock tick data is widely used for market volatility prediction, but the stock tick data has its disadvantages during the prediction process. Since these tick data are often unordered, variant interval and incomplete, it is very necessary to do the data preprocessing before prediction. The raw tick price data is preprocessed through the following two steps:

- **Sorting**. Because the tick price data is not recorded by the order of their time stamps, we must sort the entire list of the records by their time stamps.
- **Interpolation**. Time intervals between consecutive transactions are not the same, and sometimes there exists no record for some time periods. In order to perform time-aligned correlation analysis, we need to determine what price value should be filled in during that time period. In this paper, we choose the nearest closing price. This method splits tick data in a given time unit interval and samples the closing price in each time unit.

2) The Market News: News articles need to be preprocessed before aligning them with the time series of historical stock price data. Given a set of news articles, our objective is to extract some useful information from the words of the articles. For text document based information sources such as news articles[8][14], we need perform the following four steps in the data preprocessing phase: language-specific term segmentation and refinement, term normalization and filtering, feature selection and term weighting. Here we use Chinese segmentation software ICTCLAS to segment the sentences in the news. We keep the relatively more representative words, such as adjectives, nouns and verbs etc., and remove the less important words, i.e. the stop words, in the second step. Then, we use commonly adopted weighting scheme - $tf \cdot idf$ - to calculate the weights of each word. In this way, each news article is projected onto the vector space and could be simply represented as a vector. For feature selection, we use the chi-square (χ^2) method that compares the difference of one vector with another vector, and gives a score for the difference, as shown in Formula (1). To be specific, the χ^2 method calculates the difference score feature-by-feature by comparing the vector of the feature with the vector of the labels. For example, let $p(t_k)$ denote the percentage that word t_k occurs in the articles, and denote $1 - p(t_k)$ as $p(\overline{t_k})$, and $p(+)$ and $p(-)$ are the probability of class labels. Then,

$$\chi^2 = \frac{(ad - bc)^2 N}{(a+b)(a+c)(b+d)(c+d)} \tag{1}$$

where

$$\begin{aligned} a &= N \cdot p(\overline{t_k}, -) \\ b &= N \cdot p(\overline{t_k}, +) \\ c &= N \cdot p(t_k, -) \\ d &= N \cdot p(t_k, +) \end{aligned} \tag{2}$$

The higher the χ^2 score is, the more informative for prediction t_k will be. In this paper, we choose top 1000 χ^2 scored words as features.

B. Data Alignment

In order to predict the stock price volatility, we need to resample the time series to a common length. In addition, we also need to correlate one information source with another, such as correlating news articles with historical price time series.

1) The Market Price: Our goal for market volatility analysis is to predict the upcoming trend of market price, this can be done using the rate of change for a given period n on a time series P, and often referred to as relative difference in percentage (RDP):

$$RDP - k\,(P_k) = 100 \times \frac{P_t - P_{t-k}}{P_{t-k}} \tag{3}$$

where k is the prediction horizon.

From both the trading and the prediction perspective, the prediction horizon should not be too long or too short. Moving average (MA) is commonly used with time series data to smooth out short-term fluctuations and highlight longer term trends or cycles, and MA-5 is mostly common used in the prediction over time series. In this paper, we set the value k as 5 and the formulae of RDPs are listed in Table I.

In addition to RDPs, we can also adopt some other popular market indicators from stock technical analysis. The formulae of some of such market indicators are listed in Table II, where p_i is the price at time point i, q refers to the order counted in minute, $RSI(q)$ is relative strength index, $RSV(q)$ is raw

TABLE I. THE FORMULAE OF RDPS

Indicator	Formula
RDP-5	$100 * (p_i - p_{i-5})/p_{i-5}$
RDP-10	$100 * (p_i - p_{i-10})/p_{i-10}$
RDP-15	$100 * (p_i - p_{i-15})/p_{i-15}$
RDP-20	$100 * (p_i - p_{i-20})/p_{i-20}$
RDP-25	$100 * (p_i - p_{i-25})/p_{i-25}$
RDP-30	$100 * (p_i - p_{i-30})/p_{i-30}$

stochastic value, $R(q)$ is William index, $BIAS(q)$ represents the bias of the stock price, and $PSY(q)$ means psychological line.

2) The Market News: In the context of news articles, we need to extract news features that can correlate with the stock price volatility. Given a set of stocks, $S = S_1, S_2, \cdots$, where a stock $S_i = s_{i1}, s_{i2}, \cdots, s_{iT}$ is a sequence of stock prices in the time interval T (for example at 1 minute sampling rate). In the same interval there exists a set of news articles $A = A_1, A_2, \cdots$, where a news article A_i contains a set of features (such as terms) f_{ij} $(j = 1, \cdots, m)$. We assume that it is known which stock S_i an article is related to. We represent a feature f related to a stock S_i in the time interval T as a time series, $f(i) = f^i(1), f^i(2), \cdots, f^i(T)$, where $f^i(t)$ is defined as follows:

$$f^i(t) = \frac{AF_{i,f}(t)}{N_i(t)} \times log(\frac{N_i(A)}{AF_{i,f}(A)}) \tag{4}$$

where $AF_{i,f}(t)$ denotes the number of related articles in A containing the feature f for the stock S_i at time t and $N_i(t)$ denotes the total number of articles in A which are related to stock S_i at time t. Here t is a time unit defined by user. We call f_i the adjusted TF-IDF value of the news feature f related to stock S_i at time t. Similarly, $AF_{i,f}(A)$ denotes the number of articles in A that are related to stock S_i and containing the feature f during the interval T and $N_i(A)$ denotes the total number of articles in A which are related to stock S_i during the interval T.

C. Information Sources in Stock Market

Stock market is a complex system which has a lot of information, and these information would have more or less impacts on the stock market volatility. How to set up an effect model to detect these information and the inter correlations among them to make a good prediction is a difficult research problem. Traditional models usually use the historical prices to make the prediction. But our previous finds showed that use multiple data sources could help make better prediction[15].

Suppose there are m kinds of data sources in the stock market, for single data source model, the prediction problem can be represented as,

$$\mathbf{DS} \mapsto \mathbb{L} \tag{5}$$

where **DS** indicates the data source used for prediction. For multiple data sources model, there are mainly two methods to combine different kinds of data sources to make prediction. The first one is naive combine, the prediction problem can be represented as,

$$\{\mathbf{DS}_1,\ \mathbf{DS}_2, \cdots, \mathbf{DS}_m\} \mapsto \mathbb{L} \tag{6}$$

The naive combine method cares about all the data sources together, but these data sources have their own specifications and it's hard to find a good learning algorithm for all of these data sources. Another method is to use multi-kernel learning algorithm to deal with these different data sources with different kernels. In a multi-kernel prediction model, we train a specific sub-kernel $subkernel^{(i)}$ for each data source DS_i, and the prediction problem can be represented as,

$$\left.\begin{array}{r} subkernel^{(1)} : \mathbf{DS}_1 \\ subkernel^{(2)} : \mathbf{DS}_2 \\ \cdots \\ subkernel^{(m)} : \mathbf{DS}_m \end{array}\right\} kernel \mapsto \mathbb{L} \tag{7}$$

III. MULTI-KERNEL LEARNING BASED EXTREME LEARNING MACHINE FOR STOCK PREDICTION

In this section, we firstly introduce the basic extreme learning machine(B-ELM) and then give detailed description of the main functional components in the MKL-ELM model, including applying appropriate kernel method to train data from different data sources and integrating the selected sub-kernels for classification based prediction.

A. Extreme Learning Machine(ELM)

Extreme learning machine(ELM) is a new learning algorithm which can easily achieve good generalization performance at extremely fast learning speed by randomly choosing hidden nodes and analytically determining the output weights of single-hidden layer feedforward neural networks(SLFNs).

For the SLFNs with random hidden nodes, suppose we have N arbitrary distinct samples (X_i, t_i), where $X_i = [x_{i1}, x_{i2}, \cdots, x_{in}]^T \in R^n$ and $t_i = [t_{i1}, t_{i2}, \cdots, t_{im}]^T \in R^m$. Then the standard SLFNs with $\tilde{N}$ hidden nodes can be represented as:

$$\sum_{i=1}^{\tilde{N}} \beta_i g(\mathbf{W}_i \cdot \mathbf{X}_j + b_i) = \mathbf{o}_j, j = 1, \cdots, N \tag{8}$$

where g is the activation function, $\mathbf{W}_i = [\omega_{i1}, \omega_{i2}, \cdots, \omega_{in}]^T$ is the input weight, β_i is the output weight, and b_i is the bias of ith hidden neuron. $\mathbf{W}_i \cdot \mathbf{X}_j$ denotes the inner product of $\mathbf{W}_i$ and $\mathbf{X}_j$.

The optimization goal of SLFNs is to minimize the error of the outputs, which can be represented as,

$$\sum_{j=1}^{\tilde{N}} \|\mathbf{o}_j - \mathbf{t}_j\| = 0 \tag{9}$$

and there exist β_i, $\mathbf{W}_i$ and b_i such that

$$\sum_{i=1}^{\tilde{N}} \beta_i g(\mathbf{W}_i \cdot \mathbf{X}_j + b_i) = \mathbf{t}_j, j = 1, \cdots, N \tag{10}$$

TABLE II. OTHER MARKET INDICATORS

RDP	Formula
RSI(q)	$100 * Up/(Up + Down)$
RSV(q)	$100 * (p_0 - min_q(p_i))/(max_q(p_i) - min_q(p_i))$
R(q)	$100 * (max_q(p_i) - p_0)/(max_q(p_i) - min_q(p_i))$
BIAS(q)	$100 * (p_0 - (\sum_i p_i)/q)/((\sum_i p_i)/q)$
PSY(q)	$100 * (\sum l\{p_i > p_{i-1}\})/q$

The above equation can be transformed as,

$$\mathbf{H}\beta = \mathbf{T} \tag{11}$$

where $\mathbf{H}$ is the output matrix of the neural network.

$$\begin{gathered}\mathbf{H}(W_1, \cdots, W_{\tilde{N}}, b_1, \cdots, b_{\tilde{N}}, X_1, \cdots, X_N) = \\ \begin{bmatrix} g(W_1 \cdot X_1 + b_1) & \cdots & g(W_{\tilde{N}} \cdot X_1 + b_{\tilde{N}}) \\ \vdots & \cdots & \vdots \\ g(W_1 \cdot X_N + b_1) & \cdots & g(W_{\tilde{N}} \cdot X_N + b_{\tilde{N}}) \end{bmatrix}_{N \times \tilde{N}}\end{gathered} \tag{12}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_{\tilde{N}}^T \end{bmatrix}_{\tilde{N} \times m} \quad and \quad \mathbf{T} = \begin{bmatrix} \mathbf{t}_1^T \\ \vdots \\ \mathbf{t}_{\tilde{N}}^T \end{bmatrix}_{N \times m} \tag{13}$$

In order to train the SLFN, the specific $\hat{\mathbf{W}}_i$, $\hat{b}_i$ and $\hat{\beta}$ should be trained such that

$$\left\| \mathbf{H}(\hat{\mathbf{W}}_i, \hat{b}_i)\hat{\beta} - \mathbf{T} \right\| = \min_{\mathbf{W}, b, \beta} \|\mathbf{H}(\mathbf{W}_i, b_i)\beta - \mathbf{T}\| \tag{14}$$

where $i = 1, \cdots, \tilde{N}$, which is the equivalent to the minimization of the cost function.

$$E = \sum_{j=1}^{N} \left(\sum_{i=1}^{\tilde{N}} \beta_i g\left(\mathbf{W}_i \cdot \mathbf{X}_j + b_i\right) - \mathbf{t}_j \right)^2 \tag{15}$$

In order to solve this problem, some algorithms based on the gradient were presented, such as BP learning algorithm and its variants[16][17], but the basic gradient-based learning algorithm need to tune all the parameters within the network. As a new learning algorithm, ELM does not need to tune the parameters in the iteration, which has also been used to deal with this problem.

In ELM algorithm, the hidden layer output matrix $\mathbf{H}$ remains unchanged once the input weights $\mathbf{W}_i$ and the hidden layer biases b_i are randomly assigned. Training an SLFN can be transformed to solve a linear system $\mathbf{H}\beta = \mathbf{T}$. And the output weight β can be analytically determined by

$$\hat{\beta} = \mathbf{H}^{\dagger}\mathbf{T} \tag{16}$$

where $\mathbf{H}^{\dagger}$ is the *Moore-Penrose generalized inverse* of the matrix $\mathbf{H}$. Huang[10] has proved that the solution $\hat{\beta}$ is the smallest norm least-squares one and the solution $\hat{\beta}$ is unique.

B. Multi-kernel Learning based ELM (MKL-ELM)

The basic ELM can linearly be extended to kernel-based methods. For RBF network, the Equation (10) could be formulated as,

$$\sum_{i=1}^{\tilde{N}} \beta_i \phi_i(X_j) = \mathbf{t}_j, j = 1, \cdots, N \tag{17}$$

where β_i is the output weights connecting the ith kernel and the output neurons, $\phi_i(X)$ is the output of the ith kernel. The output weight β could be solved by Equation (16).

In order to take full advantages of the diversity of multiple sources data and make a fast prediction, we propose a multiple kernel learning based extreme learning machine (MKL-ELM) mode. We train different kernels for different data sources separately to get the best sub-kernels, then combine the sub-kernels into a new kernel, and train the new model to make the final prediction.

Suppose we have m kinds of different data sources, and called DS_1, DS_2, $\cdots$, DS_m separately, $\mathbf{k}_{DS_1}$ is the best selected kernel for the DS_1, $\mathbf{k}_{DS_2}$ is the best selected kernel for the DS_2, $\cdots$, $\mathbf{k}_{DS_m}$ is the best selected kernel for the DS_m. The kernel combined both $\mathbf{k}_{DS_1}$, $\mathbf{k}_{DS_2}$, $\cdot$, $\mathbf{k}_{DS_m}$ can be represented as,

$$\begin{aligned}\mathbf{k}(X_i^{(1)}, X_i^{(2)}, \cdot, X_i^{(m)}) = d_{DS_1}\mathbf{k}_{DS_1}(X_i^{(1)}) \\ + d_{DS_2}\mathbf{k}_{DS_2}(X_i^{(2)}) \\ + \cdots \\ + d_{DS_m}\mathbf{k}_{DS_m}(X_i^{(m)})\end{aligned} \tag{18}$$

where $d_{DS_1}, d_{DS_2}, \cdots, d_{DS_m}$ are the sub-kernel's weights, and $d_{DS_i} \in (0, 1)$. $X^{(1)}$ are the instances of the DS_1, $X^{(2)}$ are the instances of the DS_2, $\cdots$, $X^{(m)}$ are the instances of the DS_m.

Suppose we have N_{DS_1} arbitrary distinct samples $\left(X_i^{(1)}, t_i\right)$ for the DS_1, N_{DS_2} arbitrary distinct samples $\left(X_i^{(2)}, t_i\right)$ for the DS_2, $\cdots$, and N_{DS_m} arbitrary distinct samples $\left(X_i^{(m)}, t_i\right)$ for the DS_m, where $X_i^{(1)} = \left[x_{i1}^{(1)}, x_{i2}^{(1)}, \cdots, x_{in_1}^{(1)}\right]^T \in R^{n_1}$, $X_i^{(2)} = \left[x_{i1}^{(2)}, x_{i2}^{(2)}, \cdots, x_{in_2}^{(2)}\right]^T \in R^{n_2}$, $\cdots$, $X_i^{(m)} = \left[x_{i1}^{(m)}, x_{i2}^{(m)}, \cdots, x_{in_m}^{(m)}\right]^T \in R^{n_m}$ and $t_i = [t_{i1}, t_{i2}, \cdots, t_{im}]^T \in R^m$. And the N_{DS_1} arbitrary distinct samples $\left(X_i^{(1)}, t_i\right)$ for the DS_1, the N_{DS_2} distinct

samples $\left(X_i^{(2)}, t_i\right)$ for the DS_2, $\cdots$, the N_{DS_m} distinct samples $\left(X_i^{(m)}, t_i\right)$ for the DS_m have the same time stamp.

The process of our proposed MKL-ELM has three steps, the first one is training the sub-kernels, i.e. training a K-ELM just for the market prices and the market new separately. After selecting the two best kernels, the second step is to find the best combination of the two sub-kernels, and find the best weights for them. The third step is to test the proposed MKL-ELM on the testing set.

- **Step1 : Train sub − kernels separately**
  Followed Equation (17), the optimization of sub-kernel for Data Source DS_k can be represented as:

$$\sum_{i=1}^{\tilde{N}} \beta_i^{(k)} \phi_i^{(k)}(X_j^{(k)}) = \mathbf{t}_j, \quad j = 1, \cdots, N \tag{19}$$

  where $\beta_i^{(k)}$ is the output weights connecting the ith kernel and the output neurons for the market news, $\phi_i^{(k)}$ is the output of the ith kernel for the market news. And it can be rewritten as:

$$\mathbf{H}^{(k)}\beta^{(k)} = \mathbf{T} \tag{20}$$

  where $\mathbf{H}^{(k)}$ is the output matrix of the RBF network for the market news.

$$\mathbf{H}^{(k)}(\mu_1^{(k)}, \cdots, \mu_{\tilde{N}}^{(k)}, \sigma_1^{(k)}, \cdots, \sigma_{\tilde{N}}^{(k)}, X_1^{(k)}, \cdots, X_N^{(k)}) = \begin{bmatrix} \phi^{(k)}(\mu_1^{(k)}, \sigma_1^{(k)}, X_1^{(k)}) & \cdots & \phi^{(k)}(\mu_{\tilde{N}}^{(k)}, \sigma_{\tilde{N}}^{(k)}, X_1^{(k)}) \\ \vdots & \cdots & \vdots \\ \phi^{(k)}(\mu_1^{(k)}, \sigma_1^{(k)}, X_N^{(k)}) & \cdots & \phi^{(k)}(\mu_{\tilde{N}}^{(k)}, \sigma_{\tilde{N}}^{(k)}, X_N^{(k)}) \end{bmatrix}_{N \times \tilde{N}} \tag{21}$$

$$\beta^{(k)} = \begin{bmatrix} \beta_1^{(k)T} \\ \vdots \\ \beta_{\tilde{N}}^{(k)T} \end{bmatrix}_{\tilde{N} \times m} \quad and \quad \mathbf{T} = \begin{bmatrix} \mathbf{t}_1^T \\ \vdots \\ \mathbf{t}_{\tilde{N}}^T \end{bmatrix}_{N \times m} \tag{22}$$

  The model with m kinds of different data sources DS_1, DS_2, $\cdots$, DS_m can be represented as:

$$\begin{aligned} \mathbf{H}^{(1)}\beta^{(1)} &= \mathbf{T} \\ \mathbf{H}^{(2)}\beta^{(2)} &= \mathbf{T} \\ &\cdots \\ \mathbf{H}^{(m)}\beta^{(m)} &= \mathbf{T} \end{aligned} \tag{23}$$

  Once the kernel function is selected, the output weights $\hat{\beta}^{(1)}$, $\hat{\beta}^{(2)}$, $\cdots$, $\hat{\beta}^{(m)}$ could be solved by Equation (16). Our purpose is to find the m best kernel functions $\mathbf{k}_{DS_1}$, $\mathbf{k}_{DS_2}$, $\cdots$, $\mathbf{k}_{DS_m}$.

- **Step2 : Combine sub − kernelsbyMKL**
  How to combine these sub-kernels into a new kernel is an important but difficult issue. There are many learning mechanisms in multi-kernel learning, such as the composite kernels, the multi-scale kernels and the infinite kernels [18][19][20]. In this paper, we employ a simple method, which combines the sub-kernels linearly to combine the sub-kernels.

$$\begin{aligned} \phi_i(X_j^{(1)}, X_j^{(2)}) = d_{DS_1}\phi_i^{(1)}(X_j^{(1)}) \\ +d_{DS_2}\phi_i^{(2)}(X_j^{(2)}) \\ +\cdots \\ +d_{DS_m}\phi_i^{(m)}(X_j^{(m)}) \end{aligned} \tag{24}$$

$$\sum_{i=1}^{\tilde{N}} \beta_i \phi_i(X_j^{(1)}, X_j^{(2)}) = \mathbf{t}_j, \quad j = 1, \cdots, N \tag{25}$$

  As a result of this, the optimization problem can be reformulated as,

$$\mathbf{H}\beta = \mathbf{T} \tag{26}$$

  where $\mathbf{H} = d_{DS_1}\mathbf{H}^{(1)} + d_{DS_2}\mathbf{H}^{(2)} + \cdots + d_{DS_m}\mathbf{H}^{(m)}$ is the output matrix of the RBF network for the MKL-ELM. Then the optimization goal is to train the combining kernel to get the best combination of d_{DS_1}, d_{DS_2}, $\cdots$, d_{DS_m} and get the best $\hat{\beta}$ by Equation (16).

- **Step3 : Test MKL − ELMonDatasets**
  After the previous two steps, we can get a well-trained MKL-ELM model. In this step, we will test the MKL-ELM model on the testing set and to calculate the testing accuracy. First, we calculate the real output $\mathbf{o}_{test}$,

$$\mathbf{O}_{test} = \mathbf{H}_{test}\hat{\beta} \tag{27}$$

  where $\mathbf{H}_{test} = d_{DS_1}\mathbf{H}_{test}^{(1)} + d_{DS_2}\mathbf{H}_{test}^{(2)} + \cdots + d_{DS_m}\mathbf{H}_{test}^{(m)}$. Then we compare the real output $\mathbf{O}_{test}$ with the expected output $\mathbf{T}_{test}$ and calculate the prediction accuracy.

The bottleneck for any kernel method is the definition of a kernel $\mathbf{k}$ that accurately reflects the similarity among data samples. However, not all metric distances are permitted. In fact, valid kernels are only those fulfilling the Mercers Theorem[21] and the most common ones are the following: (1) The linear kernel, $k(x_i, x_j) = \Phi(x_i) * \Phi(x_j)$. The value of the kernel equals to the inner product of two vectors x_i and x_j in the kernel space $\Phi(x_i)$ and $\Phi(x_j)$. (2) The polynomial kernel, $k(x_i, x_j) = (\langle x_i, x_j \rangle + 1)^d$, d is an integer. (3) The radial basis function(RBF), $k(x_i, x_j) = exp(\frac{-\|x_i - x_j\|^2}{2\sigma^2}), \sigma \in R$. It is also called the Gaussian kernel. The kernel parameter should be carefully chosen as it implicitly defines the structure of the high dimensional feature space $\Phi(x)$ and thus controls the complexity of the final solution.

Here we select the Gaussian kernel as the original kernel for each data source.

$$\phi_i(X) = \phi(\mu_i, \sigma_i, X) = exp(-\frac{\|X - \mu_i\|^2}{2\sigma_i^2}) \tag{28}$$

μ_i is the ith kernel's center and σ_i is its impact width. And we also select the radial basis function (RBF) in SVM, K-ELM and our proposed MKL-ELM.

TABLE III. THE NUMBERS OF INSTANCES AFTER PREPROCESSING

	5m	10m	15m	20m	25m	30m
Total	1721	1953	2035	1965	1963	1906

IV. EXPERIMENTAL RESULTS AND DISCUSSION

In order to evaluate the performance of our proposed MKL-ELM, we tested it on HK exchange datasets with two important market information sources: the stock market historical prices and the market news. We compare it with some traditional methods, such as Back-Propagation Neural Networks(BP-NN) and Support Vector Machine(SVM). We also compare it with Basic ELM(B-ELM) and Kernel ELM(K-ELM).

A. Data Sets

In this paper, we use the following two kinds of data sources: 1) the stock market prices and 2) the market news.

- **Stock Prices**. The market prices contain all the stocks' prices of HK exchange datasets in year 2001.
- **Market news**. The market news of year 2001 used in our experiment are bought from Caihua. Each piece of news is attached with a time stamp, and the time represents the release time of the news. All the market news are written in Traditional Chinese.

Time stamps of news articles and prices are tick based.

HK exchange has thousands of stocks, but not all the stocks are active in the market. We mainly focus on the constituents of Hang Seng Index(HSI) which, according to the change log, includes 33 stocks in year 2001. However, the constituents of HSI changed twice in year 2001, which was on June 1st and July 31th. Due to the tyranny of indexing, price movement of newly added constituent is not rational and usually will be mispriced during the first few months. We only select the constituents that had been constituents through the whole year. Thus, the number of stocks left becomes 23.

The numbers of instances after preprocessing are listed in Table III.

B. Model Setup and Parameter Selection

The process of each model and the parameters selection are as follows:

- $\mathbf{BP-NN}$. Back-propagation Neural Network(BP-NN) is a traditional method to deal with the classification problem, and there are many variants for it. In this paper, we take a fast BP algorithm named Levenberg-Marquardt in our experiments as a comparison to our proposed MKL-ELM. The parameter to be tuned for BP-NN is only the number of hidden nodes. In consideration of the limit of the hardware, especially the limit of the memory, we set the number of the hidden nodes as 10.
- $\mathbf{SVM}$. Support Vector Machine(SVM) is widely applied due to its good performance in the classification problem. As mentioned above, there are some different kinds of kernels. In this paper, we take RBF kernel in our experiments. The parameters to be tuned for RBF-SVM are the kernel parameter γ and the penalty parameter C. We split the datasets into three types: training set, validation set and testing set. The size of the testing set is fixed as 100. And we apply 5-fold cross validation method in our experiments, which splits the remaining data to 5 parts, 4 parts of them are used for training and the remaining is used for validating. The grid search is used to find the best combination of the kernel's parameter γ and the penalty parameter C. The γ searches $\left\{2^{-17}, 2^{-16}, \cdots, 2^{2}\right\}$ and the C searches $\left\{2^{-5}, 2^{-4}, \cdots, 2^{14}\right\}$, and the best combination will be selected based on the accuracy on the validation set.
- $\mathbf{B-ELM}$. Basic Extreme Learning Machine(B-ELM) is designed especially for the SLFNs, the parameter to be tuned is similar to the BP-NN, just the number of hidden nodes should be tuned. But different from the BP-NN, the B-ELM does not need too much memory. In consideration of the number of input nodes and the number of instances, in this paper, we set the number of hidden nodes as 1100.
- $\mathbf{K-ELM}$. The data for single kernel ELM(K-ELM) is organized as formula (6), a naive combine of the market prices and the market news. The tuning process of K-ELM is similar to SVM, and in the process of K-ELM, the RBF kernel is also adopted. The γ of the RBF kernel searches $\left\{2^{-17}, 2^{-16}, \cdots, 2^{2}\right\}$ and regulation term C searches $\left\{2^{-5}, 2^{-4}, \cdots, 2^{14}\right\}$. We take 5-fold cross validation to get the best combination of parameters.
- $\mathbf{MKL-ELM}$. Multiple Kernel Learning based Extreme Learning Machine(MKL-ELM) has three steps in the whole tuning process, the first step is training the sub-kernels separately for the market prices and the market news. In the process of the sub-kernels training, similar to the process of the K-ELM, for example, suppose a sub-kernel training for the market prices, the tuning process of this sub-kernel, The γ_{price} of the RBF kernel searches $\left\{2^{-17}, 2^{-16}, \cdots, 2^{2}\right\}$ and regulation term C_{price} searches $\left\{2^{-5}, 2^{-4}, \cdots, 2^{14}\right\}$. And then take 5-fold cross validation to get the best combination of parameters for the sub-kernel. The second step is training for the whole two kernels system to get the best combination of the weights d_{news}, $d_{indicator}$ and the regulation term C of the multi-kernel system. And regulation term C searches $\left\{2^{-5}, 2^{-4}, \cdots, 2^{14}\right\}$.

C. Experimental Results

We evaluate the models from both the prediction accuracy and the prediction speed. The accuracy of the prediction is obtained by checking whether the direction of the predicted volatility is the same as the actual trend. The prediction accuracy is measured by,

$$accuracy = \frac{tp + tn}{tp + fp + tn + fn} \tag{29}$$

where tp and tn refer to the number of true positives and true negatives respectively. fp and fn denote the number of false

positives and false negatives. The prediction speed of each model is measured by CPU time.

We run the experiments with MATLAB R2010a in a PC machine with Core i5 and CPU 2.8GHZ. Table IV lists the cross validation results and Table V lists the results of independent testing. Table VI gives the prediction speed (running time) of each model at each time point (numbers in bold font indicate the best results at the given time point and the second best results are underlined). Each value is the mean of 50 random runs.

On the prediction accuracy side, from Table IV and Table V, we can find that SVM, K-ELM and MKL-ELM achieve better performance than BP-NN and B-ELM on both the validation set and the testing set. Among the 6 time points in Table IV, SVM gets the best validation accuracy on the 10m, 15m, 20m, 25m and 30m time points. MKL-ELM is little lower than SVM and K-ELM on the 10m, 15m, 20m, 25m and 30m, but has the best validation accuracy on the 5m time point. The results of BP-NN shows that BP-NN is not suitable for the large-scale computation. And for B-ELM, the results that B-ELM can not achieve good validation accuracy and testing accuracy. One possible reason might be that B-ELM is not suit for the nonlinear problem.

From the results at 6 time points shown in Table V, we can also find that SVM achieves the best testing accuracy on 15m, 20m and 25m time points, and MKL-ELM achieves the best testing accuracy on 5m, 10m and 30m. Especially, SVM and MKL-ELM have a better result than K-ELM on the prediction accuracy. One possible reason why MKL-ELM is better than K-ELM could be the data organizing way. In the MKL-ELM, since there are multiple data sources, and how to combine those data might be different which will affect prediction accuracy. Different data sources have its own specifications, and as the naive combination does not considers the data specifications of different sources, it can not get the best results. If there are some other information data that may affect the result of prediction, it is easy to add them into the MKL-ELM while not affecting the training processes of the foregoing data.

On the prediction speed side, the results in Table VI reveals that the learning methods based on ELM have a faster prediction speed than BP-NN and SVM. The prediction speed of MKL-ELM is more faster than SVM. The reason is that while transferring from training phase to testing phase, the SVM's support vector keeps more nodes than the methods based on ELM.

It is obviously that K-ELM model has the fastest prediction speed in all mentioned methods in this paper. Since MKL-ELM has to train multiple kernels than K-ELM in the optimization process, K-ELM is a little bit slower than MKL-ELM. Actually, in the MKL-ELM model, there is no direct relation among the kernels, so the independent kernel learning could be performed concurrently. After applying the technology of performing the multiple kernels concurrently the speed of MKL-ELM can be improved.

In general, from the above results, if we take the prediction accuracy and the prediction speed into account at the same time, we can see that MKL-ELM model performs much better than the other methodologies in most cases, and it is indeed a good learning model which is exactly what the investors or traders required in the real trading market.

TABLE IV. RESULTS OF VALIDATION ACCURACY

Validation	5m		10m	
	avg.	dev.	avg.	dev.
BP-NN	0.521	0.045	0.538	0.054
SVM	0.625	0.024	**0.661**	0.023
B-ELM	0.499	0.054	0.492	0.049
K-ELM	0.605	0.021	0.650	0.029
MKL-ELM	**0.629**	0.015	0.643	0.017

Validation	15m		20m	
	avg.	dev.	avg.	dev.
BP-NN	0.548	0.059	0.551	0.051
SVM	**0.694**	0.029	**0.686**	0.029
B-ELM	0.503	0.047	0.505	0.049
K-ELM	0.680	0.027	0.667	0.029
MKL-ELM	0.636	0.019	0.662	0.019

Validation	25m		30m	
	avg.	dev.	avg.	dev.
BP-NN	0.556	0.053	0.520	0.043
SVM	**0.659**	0.022	**0.643**	0.031
B-ELM	0.513	0.048	0.513	0.048
K-ELM	0.649	0.034	0.633	0.050
MKL-ELM	0.626	0.015	0.600	0.013

TABLE V. RESULTS OF TESTING ACCURACY

Testing	5m		10m	
	avg.	dev.	avg.	dev.
BP-NN	0.499	0.046	0.526	0.055
SVM	0.544	0.060	0.573	0.047
B-ELM	0.493	0.051	0.506	0.047
K-ELM	0.570	0.049	0.584	0.033
MKL-ELM	**0.583**	0.026	**0.604**	0.028

Testing	15m		20m	
	avg.	dev.	avg.	dev.
BP-NN	0.537	0.058	0.557	0.061
SVM	**0.644**	0.031	**0.625**	0.028
B-ELM	0.499	0.052	0.500	0.045
K-ELM	0.607	0.044	0.597	0.024
MKL-ELM	0.563	0.026	0.579	0.022

Testing	25m		30m	
	avg.	dev.	avg.	dev.
BP-NN	0.553	0.052	0.522	0.049
SVM	**0.593**	0.040	0.535	0.036
B-ELM	0.522	0.046	0.506	0.049
K-ELM	0.551	0.043	0.536	0.037
MKL-ELM	0.561	0.018	**0.562**	0.019

V. CONCLUSION AND FUTURE WORK

In this paper, we have proposed a new learning model named MKL-ELM, which combines the market prices and the market news to predict the market price volatility. And we compared it with some traditional methods, such as BP-NN and SVM, and the B-ELM and K-ELM. We conduct the experiments on the one whole year stock market tick data of Hong Kong Exchange. Results have shown that, under the consideration of both prediction accuracy and prediction speed, MKL-ELM outperforms than the the others in most cases. And further more, since this MKL-ELM can be regarded as a multiple data source integration framework, other data sources (such as trading volume) can also be filled into this framework and help achieve much better prediction results. But if the data sources become more complicated, MKL-ELM will become more time-consuming. In our future work, we will try to improve the efficiency of MKL-ELM with more data sources.

TABLE VI. RESULTS OF PREDICTION SPEED

Prediction Speed	5m		10m	
	avg.	dev.	avg.	dev.
BP-NN	0.200	0.055	0.165	0.063
SVM	0.266	0.025	0.261	0.026
B-ELM	0.149	0.072	0.149	0.084
K-ELM	**0.138**	0.084	**0.148**	0.092
MKL-ELM	0.164	0.024	0.173	0.031

Prediction Speed	15m		20m	
	avg.	dev.	avg.	dev.
BP-NN	0.190	0.114	0.178	0.068
SVM	0.266	0.030	0.252	0.026
B-ELM	0.155	0.094	0.160	0.090
K-ELM	**0.144**	0.089	**0.140**	0.070
MKL-ELM	0.188	0.021	0.179	0.015

Prediction Speed	25m		30m	
	avg.	dev.	avg.	dev.
BP-NN	0.197	0.059	0.192	0.055
SVM	0.261	0.046	0.297	0.032
B-ELM	0.160	0.094	0.151	0.077
K-ELM	**0.140**	0.090	**0.131**	0.060
MKL-ELM	0.199	0.030	0.188	0.019

VI. ACKNOWLEDGEMENT

This work was supported by National Nature Science Foundation of China [grant number 61103125]; the Doctoral Fund of Ministry of Education of China [grant number 20100141120046]; the Natural Science Foundation of Hubei Province of China [grant number 2010CDB08504]; the 111 Programme of Introducing Talents of Discipline to Universities [grant number B07037] and Wuhan University Academic Development Plan for Scholars after 1970s ("Research on Internet User Behavior").

REFERENCES

[1] K. Sureshkumar and N. Elango, "Performance analysis of stock price prediction using artificial neural network," *Global Journal of Computer Science and Technology*, vol. 12, no. 1, 2012.

[2] L.-J. Cao and F. E. H. Tay, "Support vector machine with adaptive parameters in financial time series forecasting," *IEEE Transactions on Neural Networks*, vol. 14, no. 6, pp. 1506–1518, 2003.

[3] J. L. Ticknor, "A bayesian regularized artificial neural network for stock market forecasting," *Expert Systems with Applications*, 2013.

[4] W. Dai, J.-Y. Wu, and C.-J. Lu, "Combining nonlinear independent component analysis and neural network for the prediction of asian stock market indexes," *Expert Systems with Applications*, vol. 39, no. 4, pp. 4444–4452, 2012.

[5] L. C. Martinez, D. N. da Hora, J. R. de M Palotti, W. Meira, and G. L. Pappa, "From an artificial neural network to a stock market day-trading system: A case study on the bm&f bovespa," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN'09)*. IEEE, 2009, pp. 2006–2013.

[6] N. N. Nguyen and C. Quek, "Stock price prediction using generic self-evolving takagi-sugeno-kang (gsetsk) fuzzy neural network," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN'10)*. IEEE, 2010, pp. 1–8.

[7] X. Li, C. Wang, J. Dong, F. Wang, X. Deng, and S. Zhu, "Improving stock market prediction by integrating both market news and stock prices," in *Database and Expert Systems Applications*. Springer, 2011, pp. 279–293.

[8] R. P. Schumaker and H. Chen, "Textual analysis of stock market prediction using breaking financial news: The azfin text system," *ACM Transactions on Information Systems (TOIS)*, vol. 27, no. 2, pp. 1–19, 2009.

[9] J. Bollen, H. Mao, and X. Zeng, "Twitter mood predicts the stock market," *Journal of Computational Science*, vol. 2, no. 1, pp. 1–8, 2011.

[10] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1, pp. 489–501, 2006.

[11] G.-B. Huang and L. Chen, "Convex incremental extreme learning machine," *Neurocomputing*, vol. 70, no. 16, pp. 3056–3062, 2007.

[12] G. Feng, G.-B. Huang, Q. Lin, and R. K. L. Gay, "Error minimized extreme learning machine with growth of hidden nodes and incremental learning," *IEEE Transactions on Neural Networks*, vol. 20, no. 8, pp. 1352–1357, 2009.

[13] H.-J. Rong, G.-B. Huang, and Y.-S. Ong, "Extreme learning machine for multi-categories classification applications," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN'08)*, 2008, pp. 1709–1713.

[14] R. P. Schumaker and H. Chen, "Textual analysis of stock market prediction using financial news," in *Americas Conference on Information Systems*, 2006.

[15] F. Wang, L. Liu, and C. Dou, "Stock market volatility prediction: A service-oriented multi-kernel learning approach," in *Proceedings of the 2012 IEEE Ninth International Conference on Services Computing (SCC)*. IEEE, 2012, pp. 49–56.

[16] C.-C. Cheung, S.-C. Ng, A. K. Lui, and S. S. Xu, "Enhanced two-phase method in fast learning algorithms," in *Proceedings of the 2010 International Joint Conference on Neural Networks (IJCNN'10)*. IEEE, 2010, pp. 1–7.

[17] R. Zhang, Z.-B. Xu, G.-B. Huang, and D. Wang, "Global convergence of online bp training with dynamic learning rate," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 2, pp. 330–341, 2012.

[18] S. Sonnenburg, G. Rätsch, and C. Schäfer, "A general and efficient multiple kernel learning algorithm," *NIPS*, pp. 1–8, 2006.

[19] F. R. Bach, G. R. Lanckriet, and M. I. Jordan, "Multiple kernel learning, conic duality, and the smo algorithm," in *Proceedings of the 21th International Conference on Machine learning*, 2004, pp. 1–8.

[20] A. Zien and C. S. Ong, "Multiclass multiple kernel learning," in *Proceedings of the 24th International Conference on Machine learning*, 2007, pp. 1191–1198.

[21] N. Cristianini and J. Shawe-Taylor, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge University Press, 2000.