Stock returns prediction using kernel adaptive filtering within a stock market interdependence approach

DOI:
10.1016/j.eswa.2020.113668

Citation for published version (APA):
Garcia-vega, S., Zeng, X., & Keane, J. (2020). Stock returns prediction using kernel adaptive filtering within a stock market interdependence approach. *Expert Systems with Applications*, *160*, 113668. https://doi.org/10.1016/j.eswa.2020.113668

Published in:
Expert Systems with Applications

Stock returns prediction using kernel adaptive filtering within a stock market interdependence approach

Sergio Garcia-Vega[a,*], Xiao-Jun Zeng[a], John Keane[a]

[a]*School of Computer Science, University of Manchester, Kilburn Building, Oxford Road, Manchester, M13 9PL, United Kingdom*

Abstract

Stock returns are continuously generated by different data sources and depend on various factors such as financial policies and national economic growths. Stock returns prediction, unlike traditional regression, requires consideration of both the sequential and interdependent nature of financial time-series. This work uses a two-stage approach, using kernel adaptive filtering (KAF) within a stock market interdependence approach to sequentially predict stock returns. Thus, unlike traditional KAF formulations, prediction uses not only their local models but also the individual local models learned from other stocks, enhancing prediction accuracy. The enhanced *KAF plus market interdependence* framework has been tested on 24 different stocks from major economies. The enhanced approach obtains higher sharpe ratio when compared with KAF-based methods, long short-term memory, and autoregressive-based models.

Keywords: Stock returns prediction; Sequential learning; Interdependence between markets; Kernel adaptive filtering.

1. Introduction

Stock returns are continuously generated by different and separate data sources, such as banks, corporations, and individuals. Thus, unlike traditional

[*] Corresponding author

Email addresses: sergio.garcia-vega@postgrad.manchester.ac.uk (Sergio Garcia-Vega), x.zeng@manchester.ac.uk (Xiao-Jun Zeng), john.keane@manchester.ac.uk (John Keane)

classification and regression problems, predicting these time-series requires consideration of: *i)* a continuous sequence of data records (Cui et al., 2016); *ii)* the interdependency between stock markets (Rejeb & Arfaoui, 2016). The stock returns are characterized by nonlinearities and high noise, which imposes additional challenges to obtaining reliable prediction (Ballings et al., 2015).

We introduce a two-phase framework for stock returns prediction using sequential learning within a stock market interdependence approach. Thus, the underlying models of each stock are learned separately using a kernel-based adaptive filter that encodes different patterns of the input space. Further, stock returns are predicted using not only their local models, but also the individual local models learned from other stocks, providing a natural way to incorporate these interdependencies. The proposed framework is a distributed learning paradigm rather than a centralized one in the sense that individual prediction models are learned based solely on a local data store, thus avoiding expensive and time-consuming data transportation into an integrated, central data store. Such a distributed learning paradigm is especially critical for big data analysis and real-time learning (Scardapane et al., 2016; Sohangir et al., 2018).

The framework is validated on 24 different stocks from three major economies, i.e., Germany, United States, and United Kingdom. Simulation obtains low values of mean absolute error (MAE) and mean square error (MSE) when compared with kernel-based adaptive filter, long short-term memory, and autoregressive-based methods. In addition, local models learned from stocks in the UK and the US were used to predict stocks traded in Germany, maximizing returns and reducing volatility.

The remainder of the paper is structured as follows: Section 2 shows the relevant background and related work; Section 3 introduces the proposed framework; the experimental settings are described in Section 4; Section 5 presents simulation results and analysis; and Section 6 concludes.

2. Background and Related Work

2.1. Sequential Learning

Several prediction methods that consider the sequential and interdependent nature of financial time series have been proposed. *Sequential learning* is usually addressed using the following methods: *i) statistical models*, such as autoregressive integrated moving average (Lee & Tong, 2011; Ramos et al., 2015). However, such linear models may not be well suited for characterizing the stochastic nature and uncertain dynamics of financial time series (Khashei & Hajirahimi, 2018; Suhermi et al., 2018); *ii) feedforward neural networks* (FFNNs) (Goodfellow et al., 2016), where sequential data is modeled by adding a set of delays to the input, e.g., time-delay NNs (Huang et al., 2017), that deals with the shortest path problem by means of time-dependent neurons; *iii) recurrent NNs* (Pascanu et al., 2013), that model sequence structure with recurrent lateral connections and process the data sequentially one record at a time, e.g., long short-term memory (LSTM) (Fischer & Krauss, 2018); *iv) kernel adaptive filters (KAFs)*, that combine the universal approximation property of NNs and the convex optimization of linear adaptive filters, e.g., kernel least-mean-square (KLMS) (Liu et al., 2008).

The prediction of financial time series has been addressed with a variety of NN approaches (Rather et al., 2015; Liu et al., 2015; Doucoure et al., 2016; Hosaka, 2019). However, these strategies may fail, as they can get stuck in local minima during the training stage (Principe, 2010). In addition, these methods require all the training data to be fully prepared in advance, restricting their application in sequence prediction (Gu et al., 2014). In practice, as NN models are retrained at regular intervals they require both significant computing and storage resources (Cui et al., 2016). Note that non-parametric kernel approaches have proven useful in identifying non-linear systems (Orabona et al., 2009; Zhao et al., 2011; Tang et al., 2017), showing that their convex optimization helps to reduce the computational complexity in sequential learning environments (Liu et al., 2011). The KAFs-based approaches can start learning the model without having the entire training set in advance, as their learning scheme is a combination of memory-based learning and error-correction, meaning that the model is updated sequentially in real-time while predictions are obtained. Hence, in this work, we use a KAF-based sequential learning method to model nonlinearities in financial time series.

2.2. Interdependency between stock markets

In addition to the sequential nature of financial time series, stock markets are themselves highly complex systems depending on various factors such as financial policies, national economic growths and sector performance (Zhang et al., 2019). It has became clear that all agents involved in a given stock market may exhibit interconnections and correlations, representing important internal forces of the market (Collins & Biekpe, 2003; Jizba et al., 2012) - that is, the movement of a stock market in a country is likely to be affected by movement of other stocks in both that country and in other regions (Masih & Masih, 2001). The following strategies have been proposed to identify and quantify interactions on this type of complex system (Greenblatt et al., 2012): *i) space-time*, such as covariance (Wang & Ye, 2016; De Ketelaere et al., 2018), correlation (Kenett et al., 2015), Granger causality (Papana et al., 2017), Shannon entropy (Sulthan et al., 2016), mutual information (Wang & Hui, 2017), and Renyi entropy (Brody et al., 2007); *ii) space-frequency and space-time-frequency*, such as Fourier transform (Fang & Chang, 2017; Saia et al., 2017), coherence (Vacha & Barunik, 2012), phase synchronization (Radhakrishnan et al., 2016), directed transfer function (Kamiński et al., 2001), wavelet transform (Joo & Kim, 2015; Saia, 2017), and cross-time frequency measures (Loh, 2013). The previous works study how the price of one stock is influenced by the economic factors of other markets. However, their models do not consider changes in network structure over time, meaning that the conditions for which the models were optimized may disappear (Olsen et al., 2018). Hence, how to incorporate these interdependencies into an analytical model, such as sequential learning, to predict stock returns in real-time remains an open issue.

3. Proposed sequential prediction and stock market interdependence framework

We develop a two-stage framework for stock returns prediction: (1) sequential learning, where the underlying models of each stock are learned separately using KAFs (Section 3.1); (2) interdependence between stocks, where local models learned from different stock markets are used to improve prediction (Section 3.2).

3.1. Sequential learning based on kernel adaptive filtering

Given a set of training data $\mathcal{T}=\{\boldsymbol{u}_t, y_t : t\in[1, N]\}$, where $\boldsymbol{u}_t\in\mathbb{R}^M$ is an input vector and $y_t\in\mathbb{R}$ is the desired output (see Figure 1). The task is to infer the underlying function $y=f(\boldsymbol{u})$ from the given data $\mathcal{T}$ and, for a new input vector $\boldsymbol{u}'\in\mathbb{R}^M$, predict the value of a new observation $y'\in\mathbb{R}$.

In practice, a KAF sequentially estimates f by using the current input-output pair $\{\boldsymbol{u}_t, y_t\}$ and updating the previous estimate f_{t-1} as follows (Liu et al., 2008):

$$\begin{cases} f_0 = 0 \\ e_t = y_t - f_{t-1}\left(\boldsymbol{u}_t\right) \\ f_t = f_{t-1} + \eta e_t \kappa_\sigma(\boldsymbol{u}_t, \cdot) \end{cases} \tag{1}$$

being $\eta\in\mathbb{R}^+$ the step-size, f_t the learned mapping, $e_t\in\mathbb{R}$ the prediction error, while $\kappa_\sigma(\cdot,\cdot)\in\mathbb{R}^+$ is a Mercer kernel with a bandwidth $\sigma\in\mathbb{R}^+$ that controls the mapping smoothness (Scholkopf & Smola, 2001). Note that Equation (1) creates a kernel unit for every new sample, where $\boldsymbol{u}_t$ is the center and ηe_t its coefficient, posing additional issues for continuous adaptation scenarios. A challenge is to curb the network growth by either eliminating redundant information or minimizing information loss - that is, only using input data with high information content as the new centers.

We propose to reduce the network size by partitioning the centers into distinct regions that encode different patterns of the input space. Thus, as it has shown

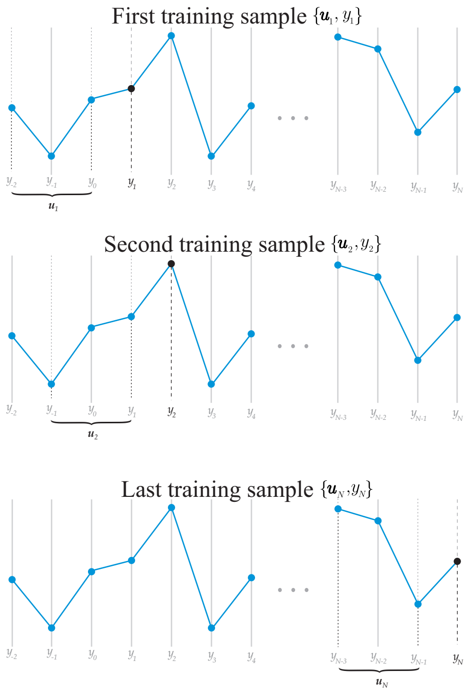

First training sample $\{\boldsymbol{u}_1, y_1\}$
y_{-2} y_{-1} y_0 y_1 y_2 y_3 y_4 y_{N-3} y_{N-2} y_{N-1} y_N
$\boldsymbol{u}_1$
Second training sample $\{\boldsymbol{u}_2, y_2\}$
y_{-2} y_{-1} y_0 y_1 y_2 y_3 y_4 y_{N-3} y_{N-2} y_{N-1} y_N
$\boldsymbol{u}_2$
Last training sample $\{\boldsymbol{u}_N, y_N\}$
y_{-2} y_{-1} y_0 y_1 y_2 y_3 y_4 y_{N-3} y_{N-2} y_{N-1} y_N
$\boldsymbol{u}_N$

Figure 1: Three representative samples of the training set $\mathcal{T}$ when M=3. The blue line represents stock returns of a given stock from which the training samples are selected. The upper, middle and lower graphs show the first, second, and last training samples, respectively.

stable performance in non-stationary environments, these patterns are identified using a previously proposed change point detection method (Yamanishi & Takeuchi, 2002). This method, at each iteration t, determines whether a change in the distribution has occurred within the sequence $y_1, \ldots, y_t$. This is done by measuring how large the probability density function p_t has moved from p_{t-1} after learning from y_t. Particularly, it is stated that a change point has taken place at iteration t when the following inequality holds:

$$\epsilon(y_t, p_{t-1}) \geq \delta \tag{2}$$

where $\epsilon(y_t, p_{t-1})$=$-\log p_{t-1}(y_t)$ denotes a prediction loss for y_t relative to a probability density function p_{t-1}, while δ∈$\mathbb{R}^+$ is a predefined threshold. Thus, when a change-point is detected within the sequence $y_1, \ldots, y_t$, we form a new set of centers or dictionary. Additionally, to avoid large discontinuities in learning (Li & Príncipe, 2017), all the centers and coefficients of the closest dictionary are copied or transferred to the newly formed dictionaries as follows:

(i) $\epsilon(y_t, p_{t-1}) \geq \delta$: This indicates a change in data distribution; hence, the following three dictionaries are formed,

- $\mathcal{C}_{|\boldsymbol{\mathcal{C}}|+1}=\{\mathcal{C}_{i^*}, \boldsymbol{u}_t\}$, with $\boldsymbol{\mathcal{C}}=\{\mathcal{C}_i : i\in[1, |\boldsymbol{\mathcal{C}}|]\}$;
- $\mathcal{Y}_{|\boldsymbol{\mathcal{Y}}|+1}=\{\mathcal{Y}_{i^*}, y_{t-1}\}$, with $\boldsymbol{\mathcal{Y}}=\{\mathcal{Y}_i : i\in[1, |\boldsymbol{\mathcal{Y}}|]\}$;
- $\mathcal{W}_{|\boldsymbol{\mathcal{W}}|+1}=\{\mathcal{W}_{i^*}, \eta e_t\}$, with $\boldsymbol{\mathcal{W}}=\{\mathcal{W}_i : i\in[1, |\boldsymbol{\mathcal{W}}|]\}$;

where $|\boldsymbol{\mathcal{C}}|=|\boldsymbol{\mathcal{Y}}|=|\boldsymbol{\mathcal{W}}|$ denotes the number of elements in each dictionary. In addition, $\mathcal{C}_{i^*}=\{\boldsymbol{u}_j : j\in[1, L]\}$, $\mathcal{Y}_{i^*}=\{y_j : j\in[1, L]\}$, and $\mathcal{W}_{i^*}=\{\eta e_j : j\in[1, L]\}$ are the closest dictionaries to y_{t-1}. Note, $\mathcal{C}_{i^*}$, $\mathcal{Y}_{i^*}$, and $\mathcal{W}_{i^*}$ are found using y_{t-1} rather than y_t. This is because, during real-time prediction tasks, only the input vectors $\boldsymbol{u}_t$ are known and the desired output y_t is the value to be predicted. As seen in Figure 1, y_{t-1} is always the last element of the input vector $\boldsymbol{u}_t$. We find the closest dictionary i^* using the Kullback-Leibler divergence as:

$$i^* = \underset{\forall i}{\arg\min}\, p_{t-1}^{\mathcal{Y}_i} \ln\left(p_{t-1}^{\mathcal{Y}_i}/p_t^{\mathcal{Y}_i}\right) \tag{3}$$

where $p_{t-1}^{\mathcal{Y}_i}$ and $p_t^{\mathcal{Y}_i}$ are the probability density functions of $\mathcal{Y}_i$ before and after learning y_{t-1}, respectively. The primary rationale behind the suggested strategy in Equation (3) is to quantify the information content that y_{t-1} will provide to each dictionary $\mathcal{Y}_i$. That is, when y_{t-1} does not provides high information content to a dictionary $\mathcal{Y}_i$, the Kullback-Leibler divergence will tend to zero, meaning that the two distributions $p_{t-1}^{\mathcal{Y}_i}$ and $p_t^{\mathcal{Y}_i}$ are identical. Thus, the dictionary $\mathcal{Y}_i$ that minimizes Equation (3) will be the closest dictionary to y_{t-1}.

(ii) $\epsilon(y_t, p_{t-1}) < \delta$: This means that the data distribution has not changed and, therefore, it is unnecessary to divide the centers into a new region. In addition, to curb the growth of the radial-basis-function structure, we incorporate an online vector quantization technique as follows,

- $\min\limits_{\forall i} p_{t-1}^{\mathcal{Y}_i} \ln\left(p_{t-1}^{\mathcal{Y}_i}/p_t^{\mathcal{Y}_i}\right) \leq \varepsilon$: The dictionary sizes remain the same and only the closest coefficient to $\boldsymbol{u}_t$ is updated using the following expression,

  $$\mathcal{W}_{i^*}^{(j^*)}{=}\mathcal{W}_{i^*}^{(j^*)} + \eta e_t, \tag{4}$$

  where the closest coefficient j^* is computed as follows,

  $$j^*{=}\arg\min_{\forall j} \left\| \boldsymbol{u}_t - \mathcal{C}_{i^*}^{(j)} \right\|, \tag{5}$$

  being $\|\cdot\|$ the ℓ_2 norm and $\varepsilon{\in}\mathbb{R}^+$ a predefined threshold. The previously imposed restraint aims to assign a new center $\boldsymbol{u}_t$ into the dictionary $\mathcal{C}_{i^*}$ only when y_{t-1} provides high information content to the dictionary $\mathcal{Y}_{i^*}$.

- $\min\limits_{\forall i} p_{t-1}^{\mathcal{Y}_i} \ln\left(p_{t-1}^{\mathcal{Y}_i}/p_t^{\mathcal{Y}_i}\right) > \varepsilon$: The sample $\boldsymbol{u}_t$ is assigned as a new center to the closest dictionary $\mathcal{C}_{i^*}$, the previous desired output y_{t-1} is stored in the dictionary $\mathcal{Y}_{i^*}$, while the set of coefficients $\mathcal{W}_{i^*}$ is updated using e_t, i.e., $\mathcal{C}_{i^*}{=}\{\mathcal{C}_{i^*}, \boldsymbol{u}_t\}$, $\mathcal{Y}_{i^*}{=}\{\mathcal{Y}_{i^*}, y_{t-1}\}$, $\mathcal{W}_{i^*}{=}\{\mathcal{W}_{i^*}, \eta e_t\}$.

The above quantization technique has similarities to the method in Chen et al. (2012). The key difference between the two strategies is that our proposal is not based on the distance measure in the input space. Rather, we use the data distribution as the criterion to update the network. This enhances utilization efficiency of the closest center, which may yield better prediction accuracy and in some circumstances a more compact network. The proposed strategy for sequential learning, when applied to a single stock, is summarized in Algorithm 1.

Algorithm 1: Strategy for sequential learning.

```
input : $\mathcal{T}$– training data
Parameter setting: $\eta$- step-size, $\sigma$- kernel parameter, $\delta$- change point
  threshold, $\varepsilon$- quantization threshold
Initial dictionaries: $\mathcal{C}_1=\{\boldsymbol{u}_1\}$, $\mathcal{Y}_1=y_0$, $\mathcal{W}_1=\eta y_1$
Sets of initial dictionaries: $\boldsymbol{\mathcal{C}}=\{\mathcal{C}_1\}$, $\boldsymbol{\mathcal{Y}}=\{\mathcal{Y}_1\}$, $\boldsymbol{\mathcal{W}}=\{\mathcal{W}_1\}$
Computation:
while $\{\boldsymbol{u}_t, y_t\}$ available do
    Select the closest dictionary: $i^*=\arg\min_{\forall i} p_{t-1}^{\mathcal{Y}_i} \ln\left(p_{t-1}^{\mathcal{Y}_i}/p_t^{\mathcal{Y}_i}\right)$
    Compute the filter output: $\hat{y}_t=\sum_{j=1}^{L} \mathcal{W}_{i^*}^{(j)} \kappa_\sigma\left(\mathcal{C}_{i^*}^{(j)}, \boldsymbol{u}_t\right)$
    Compute the error: $e_t=y_t-\hat{y}_t$
    if $\epsilon(y_t, p_{t-1}) \geq \delta$ then
        Form new dictionaries:
        $\mathcal{C}_{|\boldsymbol{\mathcal{C}}|+1}=\{\mathcal{C}_{i^*}, \boldsymbol{u}_t\}$, $\mathcal{Y}_{|\boldsymbol{\mathcal{Y}}|+1}=\{\mathcal{Y}_{i^*}, y_{t-1}\}$, $\mathcal{W}_{|\boldsymbol{\mathcal{W}}|+1}=\{\mathcal{W}_{i^*}, \eta e_t\}$
        Update set of dictionaries:
        $\boldsymbol{\mathcal{C}}=\{\boldsymbol{\mathcal{C}}, \mathcal{C}_{|\boldsymbol{\mathcal{C}}|+1}\}$, $\boldsymbol{\mathcal{Y}}=\{\boldsymbol{\mathcal{Y}}, \mathcal{Y}_{|\boldsymbol{\mathcal{Y}}|+1}\}$, $\boldsymbol{\mathcal{W}}=\{\boldsymbol{\mathcal{W}}, \mathcal{W}_{|\boldsymbol{\mathcal{W}}|+1}\}$
    else
        if $\min_{\forall i} p_{t-1}^{\mathcal{Y}_i} \ln\left(p_{t-1}^{\mathcal{Y}_i}/p_t^{\mathcal{Y}_i}\right) \leq \varepsilon$ then
            Select the closest center: $j^*=\arg\min_{\forall j} \left\|\boldsymbol{u}_t-\mathcal{C}_{i^*}^{(j)}\right\|$
            Update coefficient of closest center: $\mathcal{W}_{i^*}^{(j^*)}=\mathcal{W}_{i^*}^{(j^*)}+\eta e_t$
        else
            Assign a new center: $\mathcal{C}_{i^*}=\{\mathcal{C}_{i^*}, \boldsymbol{u}_t\}$
            Assign a new desired output: $\mathcal{Y}_{i^*}=\{\mathcal{Y}_{i^*}, y_{t-1}\}$
            Assign a new coefficient: $\mathcal{W}_{i^*}=\{\mathcal{W}_{i^*}, \eta e_t\}$
output: $\boldsymbol{\mathcal{C}}$, $\boldsymbol{\mathcal{Y}}$, $\boldsymbol{\mathcal{W}}$
```

3.2. Stock returns prediction within a stock market interdependence approach

With the aim of enhancing stock returns prediction, we consider interdependencies between stock markets. More formally, let $\mathcal{D}=\{\mathcal{T}_r : r \in [1, S]\}$ be the set of training samples of S stocks, where $\mathcal{T}_r=\{\boldsymbol{u}_{t,r}, y_{t,r} : t\in[1, N]\}$. The underlying models of each $\mathcal{T}_r$, as seen in Figure 2, are learned separately using Algorithm 1, giving three sets of dictionaries per stock, i.e., $\boldsymbol{\mathcal{C}}_r$=$\{\mathcal{C}_{i,r} : i\in[1, |\boldsymbol{\mathcal{C}}_r|]\}$, $\boldsymbol{\mathcal{Y}}_r$=$\{\mathcal{Y}_{i,r} : i\in[1, |\boldsymbol{\mathcal{Y}}_r|]\}$, and $\boldsymbol{\mathcal{W}}_r$=$\{\mathcal{W}_{i,r} : i\in[1, |\boldsymbol{\mathcal{W}}_r|]\}$.

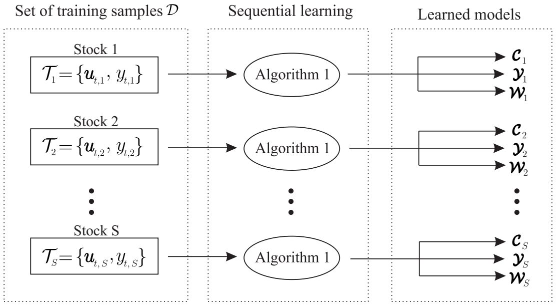

Set of training samples $\mathcal{D}$
Sequential learning
Learned models
Stock 1
$\mathcal{T}_1 = \{\boldsymbol{u}_{t,1}, y_{t,1}\}$
Algorithm 1
$\boldsymbol{\mathcal{C}}_1$
$\boldsymbol{\mathcal{Y}}_1$
$\boldsymbol{\mathcal{W}}_1$
Stock 2
$\mathcal{T}_2 = \{\boldsymbol{u}_{t,2}, y_{t,2}\}$
Algorithm 1
$\boldsymbol{\mathcal{C}}_2$
$\boldsymbol{\mathcal{Y}}_2$
$\boldsymbol{\mathcal{W}}_2$
Stock S
$\mathcal{T}_S = \{\boldsymbol{u}_{t,S}, y_{t,S}\}$
Algorithm 1
$\boldsymbol{\mathcal{C}}_S$
$\boldsymbol{\mathcal{Y}}_S$
$\boldsymbol{\mathcal{W}}_S$

Figure 2: Sequential learning within a stock market interdependence approach.

Then, when a new input vector $\boldsymbol{u}'\in\mathbb{R}^M$ arrives, the task is to predict a value $y'\in\mathbb{R}$. Thus, the first step is to find the closest dictionary i^*, as in Section 3.1, using the following expression:

$$i^* = \arg\min_{\forall i,r} p_{t-1}^{\mathcal{Y}_{i,r}} \ln \left(p_{t-1}^{\mathcal{Y}_{i,r}} / p_t^{\mathcal{Y}_{i,r}} \right), \tag{6}$$

from Equation (6), it can be seen that several stocks are considered in the selection of the closest dictionaries $\mathcal{C}_{i^*}$=$\{\boldsymbol{u}_j : j\in[1, L]\}$ and $\mathcal{W}_{i^*}$=$\{\eta e_j : j\in[1, L]\}$. In practice, these dictionaries are used to predict y' as follows:

$$y'=\sum_{j=1}^{L} \mathcal{W}_{i^*}^{(j)} \kappa_\sigma \left(\mathcal{C}_{i^*}^{(j)}, \boldsymbol{u}_t \right) \tag{7}$$

Note that, when a new sample $\boldsymbol{u}'$ comes from the r-th stock, its prediction is usually calculated using the model learned on that stock. Here, we predict $\boldsymbol{u}'$ using not only the local model but also the individual local models learned from other stocks (see Equations (6) and (7)). This strategy has similarities to previously proposed methods such as ensemble learning (Dietterich et al., 2002; Krawczyk et al., 2017) and forecast combination (Newbold & Harvey, 2002; Baumeister & Kilian, 2015). The ensemble learning framework is constructed in two steps (Zhou, 2015): *1)* a number of base learners are produced, which can be generated in a parallel or sequentially; *2)* the base learners are combined using majority voting for classification or weighted averaging for regression.

However, the combination of multiple classifiers does not always outperform the best individual classifier (Polikar, 2009). In addition, better results may be obtained when some base learners are selected instead of ensemble them (Zhou et al., 2002). In contrast to ensemble learning methods, our approach does not combine the base learners; rather, here, the prediction tasks are performed only by the *best* learner. This better utilises the internal forces of the market, provides a natural way to incorporate interdependencies between stock markets, and, in some circumstances, may enhance accuracy in real-time prediction tasks (Zhou & Tang, 2003). Finally, the proposed real-time prediction framework for stock returns is shown in Algorithm 2.

Algorithm 2: Predicting stock returns within a stock market interdependence approach.

```
input  : $\boldsymbol{u}'$; $\boldsymbol{\mathcal{C}}_1,\ldots,\boldsymbol{\mathcal{C}}_S$; $\boldsymbol{\mathcal{Y}}_1,\ldots,\boldsymbol{\mathcal{Y}}_S$; $\boldsymbol{\mathcal{W}}_1,\ldots,\boldsymbol{\mathcal{W}}_S$
Parameter setting: $\sigma$- kernel parameter
Computation:
Select the closest dictionary
$i^* = \underset{\forall i,r}{\arg\min}\, p_{t-1}^{\mathcal{Y}_{i,r}} \ln\left(p_{t-1}^{\mathcal{Y}_{i,r}} / p_t^{\mathcal{Y}_{i,r}}\right)$
Compute the output of the filter
$y' = \sum_{j=1}^{L} \mathcal{W}_{i^*}^{(j)} \kappa_\sigma\left(\mathcal{C}_{i^*}^{(j)}, \boldsymbol{u}_t\right)$
output: $y'$
```

4. Experimental Design

The aim is to use the last ten stock returns to predict the current day price change. The learned filter, as in Liu et al. (2011), is used to compute the performance values on each test set. We validate the proposed framework for stock returns prediction using the following performance measures:

- *Mean Absolute Error (MAE)*, a negatively-oriented score that measures the average of absolute errors - lower MAE values indicates better prediction performance (Baek & Kim, 2018);

- *Mean Squared Error (MSE)*, measures the average of the squares of the errors, meaning that values closer to zero are better (Kiran et al., 2020);
- *Sharpe Ratio (SR)*, quantifies the average return earned in excess of the risk-free rate per unit of volatility or total risk (Wang et al., 2020). Here, as suggested in Almahdi & Yang (2019), the Sharpe Ratio does not consider any risk-free rate.

The above metrics have been widely used to measure models' predictive power and their trading performance France & Ghose (2019); Portugal et al. (2018); Kalayci et al. (2019). The first two performance measures are regression-oriented metrics, while the last one is considered the industry standard for measuring risk-adjusted return (Jalota et al., 2017). In addition, for comparison purposes, the proposed framework is contrasted with the following online prediction methods:

1. Long Short-Term Memory (LSTM), representing the state-of-the-art RNN model for sequence learning tasks (Nweke et al., 2018). Here, as it has shown competitive performance, the implementation known as vanilla LSTM is used (Greff et al., 2017);
2. Nearest Instance Centroid-Estimation (NICE) (Li & Príncipe, 2017), a recently proposed method that outperforms traditional KAF-based algorithms in prediction of chaotic time-series;
3. Quantized Kernel Least-Mean-Square (QKLMS) (Chen et al., 2012), a well-known adaptive filtering method that uses an online vector quantization strategy;
4. Vector Autoregression (VAR) (Jang, 2020), a forecasting method used to identify relationships among multiple time-series;
5. Vector Error Correction Model (VECM) (Liang & Schienle, 2019), which is considered the standard tool to handle multivariate non-stationary time-series.

The Python implementation of the above methods can be downloaded from `https://github.com/segarciave/ESwA-2020`.

4.1. Dataset

The daily closing prices data used to calculate the returns are collected from Yahoo Finance[1]. Here, as suggested in Siikanen et al. (2018), we calculate daily log returns using the adjusted closing prices of each day. The publicly available dataset can be downloaded using Python libraries such as yfinance (please visit the following GitHub page for more details `https://segarciave.github.io/stock_returns_prediction`). Testing has been carried out on 24 different stocks from three major economies over 12 years (see Table 1). The considered training set ranges from January 17, 2006, to November 30, 2016, while the test set covers January 3, 2017, to February 28, 2018. In the simulations, as suggested in Chen (2019), the dataset is standardized so all stocks have a mean estimation of zero and a standard deviation of one.

4.2. Parameter settings

Table 2 summarizes the set-up of compared methods in the tested stocks. The parameters were heuristically adjusted to provide the best possible accuracy in this dataset. In particular, to ensure consistency in the results, both η and σ remain the same for all KAF methods. Thus, as long as this condition is met, different η and σ values from those shown in Table 2 will not offer an advantage to any particular algorithm.

We train the LSTM method using a single hidden layer with 20 neurons, as suggested in Cui et al. (2016). The LSTM activation function, as suggested in Tian et al. (2018), is the sigmoid, while the optimization is performed by Adam algorithm with MSE as the loss function (Kingma & Ba, 2014). The LSTM method was implemented using TensorFlow (version 1.4.0) [2] and Keras

[1] https://finance.yahoo.com/

[2] https://www.tensorflow.org/

Table 1: Stocks considered in the experimental design.

Market	Ticker	Stock
DE	ADS	Adidas AG
	ALV	Allianz SE
	DPW	Deutsche Post AG
	DTE	Deutsche Telekom AG
	HEI	Heidelberg Cement AG
	LIN	Linde AG
	MRK	Merck KGaA
	SAP	SAP AG
UK	ADM	Admiral Group PLC
	AHT	Ashtead Group PLC
	BA	BAE Systems PLC
	BP	BP PLC
	CCL	Carnival PLC
	IAG	International Consolidated Airlines Group
	SKY	SKY PLC
	VOD	Vodafone Group PLC
US	AAL	American Airlines Group Inc
	AAPL	Apple Inc
	AMZN	Amazon Inc
	C	Citigroup Inc
	GOOGL	Alphabet In-CL A
	MSFT	Microsoft Corp
	SPY	SPDR S&P 500 Etf
	T	AT&T

(version 2.1.2) [3]. The free parameters of autoregressive-based methods (VAR and VECM) were selected using heuristic approaches and following strategies proposed in the literature (Lütkepohl, 2013; Kuo, 2016). These parameters were chosen to achieve the best MSE on each tested data set.

The performance of our proposal is sensitive to the selection of δ and ε; however, values for these parameters can be selected as follows: *i)* δ-threshold, based on our experimentation, an appropriate value is in the interval $[5, 15]$; *ii)* ε-quantization value, where a value of 0.0001 has shown stable performance on all tested stocks.

5. Simulation Results and Analysis

The simulation results for the compared methods are shown in Tables 3 to 5, where the last row displays the average performance of each algorithm. The best overall method is in bold notation and marked with an asterisk. Table 3

[3] https://keras.io/

Table 2: Parameter setting of compared methods. M-input vector size, η-step size, σ-bandwidth, λ-quantization value, β-centroid distance, δ-threshold, ε-quantization value, $\mathcal{L}$-layers, $\mathcal{N}$-neurons per layer, $\mathcal{G}$-maximum number of lags, $\mathcal{A}$-number of lagged differences in the model, $\mathcal{I}$-cointegration rank.

Method	**Parameter**											
	M	η	σ	λ	β	δ	ε	$\mathcal{L}$	$\mathcal{N}$	$\mathcal{G}$	$\mathcal{A}$	$\mathcal{I}$
LSTM	10	-	-	-	-	-	-	1	20	-	-	-
NICE	10	0.05	0.5	0.06	2σ	-	-	-	-	-	-	-
Proposal	10	0.05	0.5	-	-	10	0.0001	-	-	-	-	-
QKLMS	10	0.05	0.5	0.4	-	-	-	-	-	-	-	-
VAR	10	-	-	-	-	-	-	-	-	15	-	-
VECM	10	-	-	-	-	-	-	-	-	-	3	0

shows the MAE values of each method in the considered stocks. The LSTM method outperforms the other algorithms, converging to the lowest average MAE value. This means that an error no greater than 0.012 can be expected during the prediction task on average. Although LSTM shows the best performance, the compared methods also converge to relatively low values of MAE. Note, a method that minimises MAE will lead to forecasts of the median (Chai & Draxler, 2014), meaning that this scale-dependent metric may be unable to quantify the prediction of abrupt changes in stock returns.

Table 4 summarizes the MSE prediction performance, where lower values are better. The compared methods show similar MSE values, suggesting a relatively good prediction performance on all considered stocks under this metric. However, NN-based methods such as LSTM requires significant computing resources, as they need to be retrained regularly in sequence prediction tasks (Cui et al., 2016). In practice, this means that several epochs have to be re-performed each time a new sample arrives in the system, which allows finding the best possible performance when the training set is updated. The learning scheme of our proposal, unlike LSTM and autoregressive-based methods, does not require the entire training set in advance to start learning the model. In contrast, the model is updated sequentially while predictions are obtained at the same time, providing an alternative to sequence prediction tasks. Additionally, the proposed interdependence strategy allows predicting each stock using not only the local model but also the models learned from other stock markets, supporting the

Table 3: Testing MAE at final iteration in stock returns prediction.

	Stock	**Method**					
		LSTM	*NICE*	*Proposal*	*QKLMS*	*VAR*	*VECM*
DE	**ADS**	0.0109	0.0109	0.0137	0.0109	0.0115	0.0136
	ALV	0.01	0.0101	0.0122	0.01	0.0101	0.0122
	DPW	0.0611	0.0608	0.0645	0.0609	0.0601	0.0711
	DTE	0.0063	0.0063	0.0142	0.0063	0.0065	0.0073
	HEI	0.0099	0.0097	0.0161	0.0098	0.0102	0.0115
	LIN	0.0077	0.0075	0.01	0.0075	0.0077	0.0093
	MRK	0.0071	0.0071	0.0112	0.0071	0.0073	0.0087
	SAP	0.007	0.0078	0.013	0.0078	0.0072	0.0086
UK	**ADM**	0.0084	0.0083	0.0154	0.0083	0.0087	0.0105
	AHT	0.0153	0.0153	0.0215	0.0152	0.0162	0.0196
	BA	0.0089	0.0087	0.0134	0.0088	0.0093	0.0106
	BP	0.0076	0.0077	0.0194	0.0077	0.0076	0.009
	CCL	0.0081	0.0082	0.0156	0.0082	0.0085	0.0106
	IAG	0.021	0.0213	0.0322	0.0212	0.0214	0.0246
	SKY	0.0323	0.0323	0.039	0.0322	0.0325	0.0398
	VOD	0.0072	0.0083	0.0166	0.0081	0.0074	0.0088
US	**AAL**	0.0139	0.0149	0.0228	0.0148	0.0155	0.0181
	AAPL	0.0086	0.0086	0.0224	0.0085	0.0088	0.0095
	AMZN	0.0095	0.0102	0.0232	0.01	0.0101	0.0115
	C	0.0087	0.0094	0.0417	0.0094	0.0097	0.0123
	GOOGL	0.008	0.0082	0.0098	0.0081	0.0082	0.0091
	MSFT	0.0077	0.0074	0.01	0.0075	0.0078	0.009
	SPY	0.0039	0.0038	0.0057	0.0038	0.0041	0.005
	T	0.0083	0.0081	0.0105	0.0082	0.0081	0.0096
	Mean	**0.012***	0.013	0.02	0.013	0.013	0.015
Standard Deviation		**0.012**	0.012	0.013	0.012	0.011	0.014

learning of arbitrary long-term dependencies in the input sequences.

The previous metrics attempt to measure the predictive power of models. However, they do not provide any insight into the performance of the compared methods from a trading point of view. Table 5 shows the SR values of each method in the considered stocks, where a higher value means greater returns for the portfolio relative to the inherent risk (Kaplanski et al., 2016). We see that our proposal outperforms the other algorithms, converging to higher values of SR, which suggests that the proposed interdependence strategy helps maximize returns while reducing volatility. This strategy selects a model based on each new data point that arrives. For example, it may be possible to obtain some predictions on ADS (a stock traded in Germany) using the model learned from VOD (a stock traded in the UK). However, the converging speed may be adversely affected if the kernel bandwidth parameter is inappropriately chosen. Consequently, a suitable value can be selected using a method like Silverman's

Table 4: Testing MSE at final iteration in stock returns prediction.

	Stock	**Method**					
		LSTM	*NICE*	*Proposal*	*QKLMS*	*VAR*	*VECM*
DE	**ADS**	0.0003	0.0003	0.0004	0.0003	0.0003	0.0004
	ALV	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003
	DPW	0.0135	0.0135	0.014	0.0134	0.0132	0.0172
	DTE	0.0001	0.0001	0.0003	0.0001	0.0001	0.0001
	HEI	0.0002	0.0002	0.0004	0.0002	0.0002	0.0002
	LIN	0.0001	0.0001	0.0002	0.0001	0.0001	0.0001
	MRK	0.0001	0.0001	0.0002	0.0001	0.0001	0.0002
	SAP	0.0001	0.0001	0.0003	0.0001	0.0001	0.0001
UK	**ADM**	0.0002	0.0002	0.0004	0.0002	0.0002	0.0002
	AHT	0.0005	0.0005	0.0008	0.0005	0.0005	0.0007
	BA	0.0002	0.0002	0.0003	0.0002	0.0002	0.0002
	BP	0.0001	0.0001	0.0005	0.0001	0.0001	0.0001
	CCL	0.0001	0.0001	0.0004	0.0001	0.0001	0.0002
	IAG	0.0009	0.0009	0.0016	0.0009	0.0009	0.0011
	SKY	0.0028	0.0028	0.0034	0.0028	0.0028	0.0037
	VOD	0.0001	0.0001	0.0004	0.0001	0.0001	0.0002
US	**AAL**	0.0004	0.0004	0.0008	0.0004	0.0004	0.0005
	AAPL	0.0001	0.0002	0.0006	0.0002	0.0002	0.0002
	AMZN	0.0002	0.0002	0.0007	0.0002	0.0002	0.0003
	C	0.0001	0.0002	0.0019	0.0002	0.0002	0.0002
	GOOGL	0.0001	0.0001	0.0002	0.0001	0.0001	0.0002
	MSFT	0.0001	0.0001	0.0002	0.0001	0.0001	0.0002
	SPY	0.000	0.000	0.0001	0.000	0.000	0.0001
	T	0.0001	0.0001	0.0002	0.0001	0.0001	0.0002
	Mean	**0.001**	**0.001**	**0.001**	**0.001**	**0.001**	**0.001**
Standard Deviation		**0.003**	**0.003**	**0.003**	**0.003**	**0.003**	**0.003**

rule of thumb (Liu et al., 2011). In addition, the proposed method uses linear adaptive structures in reproducing kernel Hilbert spaces (RKHS) to obtain nonlinear filters in the input space, enhancing prediction tasks while preserving the simplicity of linear adaptive filters.

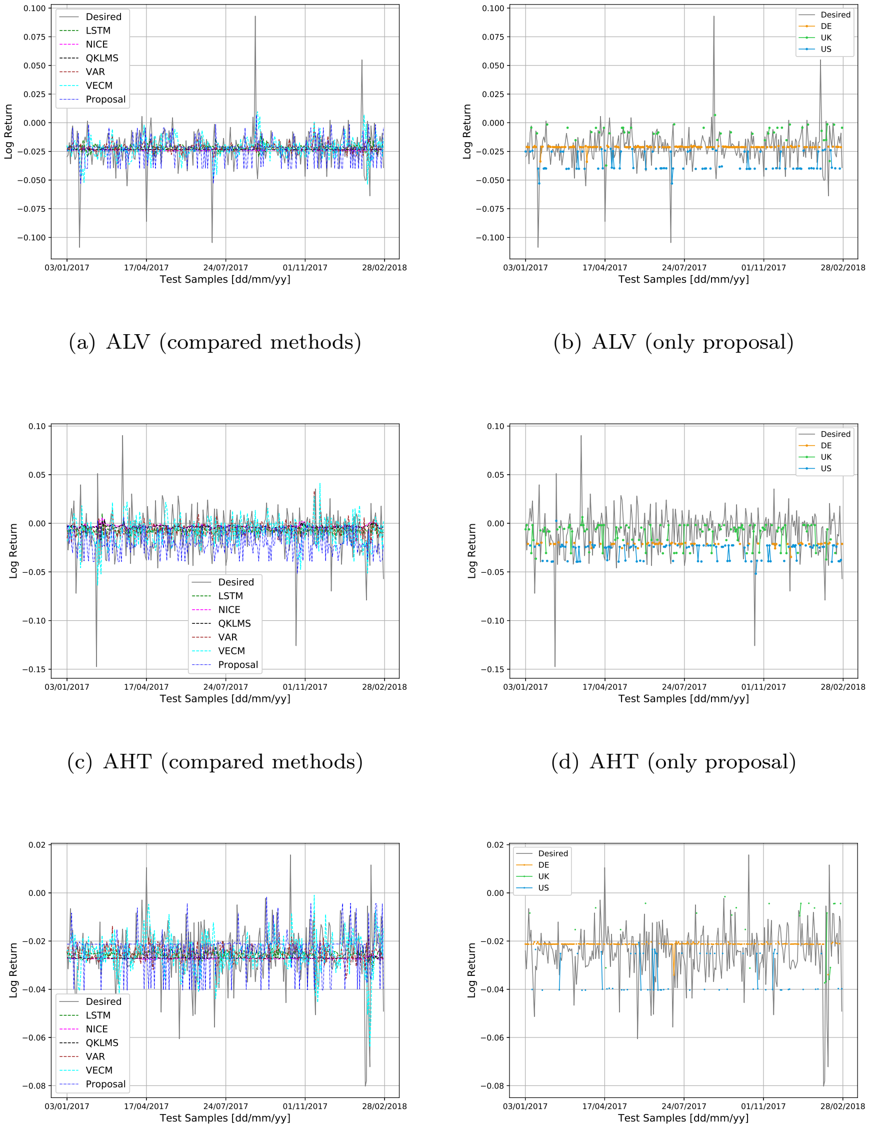
The true and predicted stock returns are shown in Figure 3 for three representative stocks, namely Allianz SE (ALV), Ashtead Group PLC (AHT), and Alphabet In-CL A (GOOGL). The left column graphs (Figures 3(a), 3(c) and 3(e)) display predictions for all tested methods. The right column graphs (Figures 3(b), 3(d) and 3(f)) show the predictions of our proposal, where each color represents a different stock market. Note that, unlike the other methods, the proposed framework obtains more accurate predictions when predicting abrupt stock return changes. Figure 3(b) provides an explanation for this behavior, where it can be seen that predictions were calculated using models learned from stocks in the US and UK. Thus, the interdependence strategy of

Table 5: Testing SR at final iteration in stock returns prediction.

	Stock	Method					
		LSTM	*NICE*	*Proposal*	*QKLMS*	*VAR*	*VECM*
DE	**ADS**	0.9017	0.2162	3.6583	0.0051	2.2682	0.9369
	ALV	1.2965	0.2089	3.3982	0.0729	1.0817	0.0021
	DPW	-0.7522	-1.0027	2.13	-1.1187	-0.8717	0.7948
	DTE	0.2688	-0.0372	1.9498	-0.1236	0.7665	1.2571
	HEI	0.8574	0.2089	2.0087	-0.0287	1.3908	2.0077
	LIN	0.5044	0.1126	3.1285	0.0646	1.2623	2.5074
	MRK	0.7296	0.0983	3.627	-0.0257	1.4655	0.6756
	SAP	0.4705	0.0497	4.39	-0.0371	1.8604	2.4806
UK	**ADM**	0.4663	0.334	5.9617	0.0065	2.4326	1.0164
	AHT	2.2794	0.936	5.3711	-1.2048	-1.3246	-0.3778
	BA	0.7246	0.0998	2.1991	-0.0579	1.2097	1.2656
	BP	1.0361	1.5645	6.549	0.1094	-0.0275	-1.3034
	CCL	1.2433	0.5456	5.9871	0.0216	-0.7435	-0.5154
	IAG	1.9193	-1.123	4.5652	1.5662	0.3662	-1.3497
	SKY	-0.2469	0.3637	1.4274	0.318	-1.2898	-1.3334
	VOD	0.6955	0.222	5.2585	0.1688	0.3891	1.4227
US	**AAL**	1.1319	-0.5774	5.8926	0.4946	-1.144	-1.1616
	AAPL	0.6432	0.1279	2.6692	-0.1274	1.0422	1.648
	AMZN	0.5934	0.1681	1.5527	-0.0205	1.0327	1.7307
	C	0.975	0.2737	3.6621	0.0879	1.1656	-0.3031
	GOOGL	0.5118	0.1185	2.9763	0.0459	1.3334	1.2835
	MSFT	0.5291	0.1301	2.8933	0.0355	1.2268	2.1107
	SPY	0.4121	0.0827	1.7878	-0.0977	1.0711	1.7259
	T	1.2947	0.1438	3.3456	-0.0684	1.3681	1.1828
	Mean	0.77	0.136	**3.6***	0.004	0.722	0.738
Standard Deviation		0.606	0.517	**1.532**	0.484	1.061	1.207

our proposal provides an advantage over competition algorithms, as they do not consider the interconnections between stock markets. Figure 3(d) shows a similar situation, where predictions are improved using models learned from the US. Figure 3(f) further suggests that the predictions of a particular stock can be enhanced by incorporating models learned from other stock markets.

Figure 4 shows the models used by the proposed framework to predict the considered stocks, where each color represents a different stock market. Figure 4(a) displays the percentage of models from Germany, the UK and the US that were used to predict stock returns. For example, 18% of the models used to predict the 8 stocks traded in Germany are from the UK, while 60% of the models used to predict the 8 stocks traded in the US are from Germany. The models from Germany, such as SAP AG (SAP) (see Figure 4(b)), are widely used to predict stock returns in the UK and the US (see Figure 4(a)), suggesting that the patterns encoded by German stocks are more appropriate to predict



(a) ALV (compared methods)

(b) ALV (only proposal)

(c) AHT (compared methods)

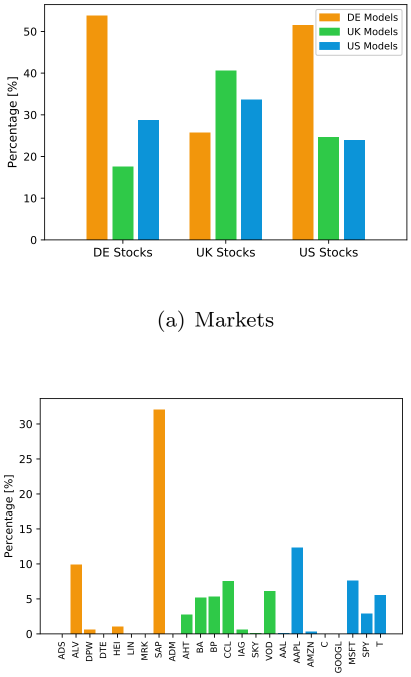
(d) AHT (only proposal)

(e) GOOGL (compared methods)

(f) GOOGL (only proposal)

Figure 3: Stock return predictions in the test sets of three representative stocks.

other stock markets. Figure 4(b) also shows that there are some stocks with 0% such as ADS, DTE, LIN, SKY, AAL, etc. This means that their models were not used to obtain their own predictions, nor were they used to predict other stocks. However, their performance is still competitive (see Tables 3 to 5).



(a) Markets

(b) Stocks

Figure 4: Percentage of models used by the proposed framework to predict the 24 stocks.

Note: The Python implementation can be downloaded from `https://segarciave.github.io/stock_returns_prediction`.

6. Conclusions and Future Work

This study introduces a framework for stock returns prediction using KAF within a stock market interdependence approach. The framework sequentially predicts stock returns by considering interconnections between stock markets. The proposed sequential learning strategy uses the data distribution as the criterion to encode different patterns of the input space, which in some circumstances may yield a more compact network. This strategy helps maximize returns and reduces volatility while maintaining the robustness and simplicity of kernel-based adaptive filters. The proposed framework for stock returns prediction comprises an $\mathcal{O}(N^3)$ computational complexity, which results in an expensive execution time when compared to conventional algorithms. Thus, more elaborate optimization algorithms or GPU-based implementations must be considered to reduce the computational burden.

The framework has been tested on 24 different stocks from three major economies. Simulation results demonstrate that the interdependence strategy used by our proposal enhances prediction accuracy, representing an advantage over compared methods. In addition, as the simulation results show, the models learned from the German market are more suitable for making predictions in other stock markets. This suggests that the US market is more influenced by the European and not vice versa, which is in line with previous empirical findings (Jizba et al., 2012; Rezayat & Yavas, 2006).

We are in the process of expanding our research in the following areas: *i)* considering other stock markets and portfolios of financial assets such as commodities and bonds; *ii)* considering additional Mercer kernels, i.e., not restricted to the Gaussian kernel; *iii)* introducing a hyper-parameter tuning procedure into the proposed method.

Acknowledgment

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 675044.

References

Almahdi, S., & Yang, S. Y. (2019). A constrained portfolio trading system using particle swarm algorithm and recurrent reinforcement learning. *Expert Systems with Applications*, *130*, 145–156.

Baek, Y., & Kim, H. Y. (2018). Modaugnet: A new forecasting framework for stock market index value with an overfitting prevention lstm module and a prediction lstm module. *Expert Systems with Applications*, *113*, 457–480.

Ballings, M., Van den Poel, D., Hespeels, N., & Gryp, R. (2015). Evaluating multiple classifiers for stock price direction prediction. *Expert Systems with Applications*, *42*, 7046–7056.

Baumeister, C., & Kilian, L. (2015). Forecasting the real price of oil in a changing world: a forecast combination approach. *Journal of Business & Economic Statistics*, *33*, 338–351.

Brody, D. C., Buckley, I. R., & Constantinou, I. C. (2007). Option price calibration from rényi entropy. *Physics Letters A*, *366*, 298–307.

Chai, T., & Draxler, R. R. (2014). Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature. *Geoscientific model development*, *7*, 1247–1250.

Chen, B., Zhao, S., Zhu, P., & Príncipe, J. C. (2012). Quantized kernel least mean square algorithm. *IEEE Transactions on Neural Networks and Learning Systems*, *23*, 22–32.

Chen, J. (2019). *Text-classification methods and the mathematical theory of Principal Components*. Ph.D. thesis Georgia Institute of Technology.

Collins, D., & Biekpe, N. (2003). Contagion and interdependence in african stock markets. *South African Journal of Economics*, *71*, 181–194.

Cui, Y., Ahmad, S., & Hawkins, J. (2016). Continuous online sequence learning with an unsupervised neural network model. *Neural computation*, *28*, 2474–2504.

De Ketelaere, B., Hubert, M., Rousseeuw, P., & Vranckx, I. (2018). Real-time outlier detection based on detmcd. In *Book of Abstracts* (p. 99).

Dietterich, T. G. et al. (2002). Ensemble learning. *The handbook of brain theory and neural networks*, *2*, 110–125.

Doucoure, B., Agbossou, K., & Cardenas, A. (2016). Time series prediction using artificial wavelet neural network and multi-resolution analysis: Application to wind speed data. *Renewable Energy*, *92*, 202–211.

Fang, M., & Chang, C.-L. (2017). Options pricing efficiency with fractional fast fourier transform. *DEStech Transactions on Environment, Energy and Earth Sciences*, .

Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, *270*, 654–669.

France, S. L., & Ghose, S. (2019). Marketing analytics: Methods, practice, implementation, and links to other fields. *Expert Systems with Applications*, *119*, 456–475.

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* volume 1. MIT press Cambridge.

Greenblatt, R. E., Pflieger, M., & Ossadtchi, A. (2012). Connectivity measures applied to human brain electrophysiological data. *Journal of neuroscience methods*, *207*, 1–16.

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. (2017). Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, *28*, 2222–2232.

Gu, Y., Liu, J., Chen, Y., Jiang, X., & Yu, H. (2014). Toselm: timeliness online sequential extreme learning machine. *Neurocomputing*, *128*, 119–127.

Hosaka, T. (2019). Bankruptcy prediction using imaged financial ratios and convolutional neural networks. *Expert Systems with Applications*, *117*, 287–299.

Huang, W., Yan, C., Wang, J., & Wang, W. (2017). A time-delay neural network for solving time-dependent shortest path problem. *Neural Networks*, *90*, 21–28.

Jalota, H., Thakur, M., & Mittal, G. (2017). Modelling and constructing membership function for uncertain portfolio parameters: A credibilistic framework. *Expert Systems with Applications*, *71*, 40–56.

Jang, W. W. (2020). Risk aversion, uncertainty, and monetary policy: Structural vector autoregressions identified with high-frequency external instruments. *Economics Letters*, *186*, 108675.

Jizba, P., Kleinert, H., & Shefaat, M. (2012). Rényi's information transfer between financial time series. *Physica A: Statistical Mechanics and its Applications*, *391*, 2971–2989.

Joo, T. W., & Kim, S. B. (2015). Time series forecasting based on wavelet filtering. *Expert Systems with Applications*, *42*, 3868–3874.

Kalayci, C. B., Ertenlice, O., & Akbay, M. A. (2019). A comprehensive review of deterministic models and applications for mean-variance portfolio optimization. *Expert Systems with Applications*, .

Kamiński, M., Ding, M., Truccolo, W. A., & Bressler, S. L. (2001). Evaluating causal relations in neural systems: Granger causality, directed transfer function and statistical assessment of significance. *Biological cybernetics*, *85*, 145–157.

Kaplanski, G., Levy, H., Veld, C., & Veld-Merkoulova, Y. (2016). Past returns and the perceived sharpe ratio. *Journal of Economic Behavior & Organization*, *123*, 149–167.

Kenett, D. Y., Huang, X., Vodenska, I., Havlin, S., & Stanley, H. E. (2015). Partial correlation analysis: Applications for financial markets. *Quantitative Finance*, *15*, 569–578.

Khashei, M., & Hajirahimi, Z. (2018). A comparative study of series arima/mlp hybrid models for stock price forecasting. *Communications in Statistics-Simulation and Computation*, (pp. 1–16).

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, .

Kiran, R., Kumar, P., & Bhasker, B. (2020). Dnnrec: A novel deep learning based hybrid recommender system. *Expert Systems with Applications*, *144*, 113054.

Krawczyk, B., Minku, L. L., Gama, J., Stefanowski, J., & Woźniak, M. (2017). Ensemble learning for data stream analysis: A survey. *Information Fusion*, *37*, 132–156.

Kuo, C.-Y. (2016). Does the vector error correction model perform better than others in forecasting stock price? an application of residual income valuation theory. *Economic Modelling*, *52*, 772–789.

Lee, Y.-S., & Tong, L.-I. (2011). Forecasting time series using a methodology based on autoregressive integrated moving average and genetic programming. *Knowledge-Based Systems*, *24*, 66–72.

Li, K., & Príncipe, J. C. (2017). Transfer learning in adaptive filters: The nearest instance centroid-estimation kernel least-mean-square algorithm. *IEEE Transactions on Signal Processing*, *65*, 6520–6535.

Liang, C., & Schienle, M. (2019). Determination of vector error correction models in high dimensions. *Journal of econometrics*, *208*, 418–441.

Liu, W., Pokharel, P. P., & Principe, J. C. (2008). The kernel least-mean-square algorithm. *IEEE Transactions on Signal Processing*, *56*, 543–554.

Liu, W., Principe, J. C., & Haykin, S. (2011). *Kernel adaptive filtering: a comprehensive introduction* volume 57. John Wiley & Sons.

Liu, Y.-k., Xie, F., Xie, C.-l., Peng, M.-j., Wu, G.-h., & Xia, H. (2015). Prediction of time series of npp operating parameters using dynamic model based on bp neural network. *Annals of Nuclear Energy*, *85*, 566–575.

Loh, L. (2013). Co-movement of asia-pacific with european and us stock market returns: A cross-time-frequency analysis. *Research in International Business and Finance*, *29*, 1–13.

Lütkepohl, H. (2013). Vector autoregressive models. In *Handbook of Research Methods and Applications in Empirical Macroeconomics*. Edward Elgar Publishing.

Masih, A. M., & Masih, R. (2001). Dynamic modeling of stock market interdependencies: an empirical investigation of australia and the asian nics. *Review of Pacific Basin Financial Markets and Policies*, *4*, 235–264.

Newbold, P., & Harvey, D. I. (2002). Forecast combination and encompassing. *A companion to economic forecasting*, (pp. 268–283).

Nweke, H. F., Teh, Y. W., Al-Garadi, M. A., & Alo, U. R. (2018). Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges. *Expert Systems with Applications*, *105*, 233–261.

Olsen, R. B., Glattfelder, J. B., & Golub, A. (2018). The alpha engine: Designing an automated trading algorithm. In *High-Performance Computing in Finance* (pp. 49–76). Chapman and Hall/CRC.

Orabona, F., Keshet, J., & Caputo, B. (2009). Bounded kernel-based online learning. *Journal of Machine Learning Research*, *10*, 2643–2666.

Papana, A., Kyrtsou, C., Kugiumtzis, D., & Diks, C. (2017). Financial networks based on granger causality: A case study. *Physica A: Statistical Mechanics and its Applications*, *482*, 65–73.

Pascanu, R., Mikolov, T., & Bengio, Y. (2013). On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning* (pp. 1310–1318).

Polikar, R. (2009). Ensemble learning. *Scholarpedia*, *4*, 2776. Revision #186077.

Portugal, I., Alencar, P., & Cowan, D. (2018). The use of machine learning algorithms in recommender systems: A systematic review. *Expert Systems with Applications*, *97*, 205–227.

Principe, J. C. (2010). *Information theoretic learning: Renyi's entropy and kernel perspectives*. Springer Science & Business Media.

Radhakrishnan, S., Duvvuru, A., Sultornsanee, S., & Kamarthi, S. (2016). Phase synchronization based minimum spanning trees for analysis of financial time series with nonlinear correlations. *Physica A: Statistical Mechanics and its Applications*, *444*, 259–270.

Ramos, P., Santos, N., & Rebelo, R. (2015). Performance of state space and arima models for consumer retail sales forecasting. *Robotics and computer-integrated manufacturing*, *34*, 151–163.

Rather, A. M., Agarwal, A., & Sastry, V. (2015). Recurrent neural network and a hybrid model for prediction of stock returns. *Expert Systems with Applications*, *42*, 3234–3241.

Rejeb, A. B., & Arfaoui, M. (2016). Financial market interdependencies: A quantile regression analysis of volatility spillover. *Research in International Business and Finance*, *36*, 140–157.

Rezayat, F., & Yavas, B. F. (2006). International portfolio diversification: A study of linkages among the us, european and japanese equity markets. *Journal of Multinational Financial Management*, *16*, 440–458.

Saia, R. (2017). A discrete wavelet transform approach to fraud detection. In *International Conference on Network and System Security* (pp. 464–474). Springer.

Saia, R., Carta, S. et al. (2017). A frequency-domain-based pattern mining for credit card fraud detection. In *IoTBDS* (pp. 386–391).

Scardapane, S., Wang, D., & Panella, M. (2016). A decentralized training algorithm for echo state networks in distributed big data applications. *Neural Networks*, *78*, 65–74.

Scholkopf, B., & Smola, A. J. (2001). *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press.

Siikanen, M., Baltakys, K., Kanniainen, J., Vatrapu, R., Mukkamala, R., & Hussain, A. (2018). Facebook drives behavior of passive households in stock markets. *Finance Research Letters*, *27*, 208–213.

Sohangir, S., Wang, D., Pomeranets, A., & Khoshgoftaar, T. M. (2018). Big data: Deep learning for financial sentiment analysis. *Journal of Big Data*, *5*, 3.

Suhermi, N., Prastyo, D. D., Ali, B. et al. (2018). Roll motion prediction using a hybrid deep learning and arima model. *Procedia computer science*, *144*, 251–258.

Sulthan, A., Jayakumar, S., & David, G. (2016). On the review and application of entropy in finance. *International Journal of Business Insights & Transformation*, *10*.

Tang, J., Zhang, J., Wu, Z., Liu, Z., Chai, T., & Yu, W. (2017). Modeling collinear data using double-layer ga-based selective ensemble kernel partial least squares algorithm. *Neurocomputing*, *219*, 248–262.

Tian, Y., Zhang, K., Li, J., Lin, X., & Yang, B. (2018). Lstm-based traffic flow prediction with missing data. *Neurocomputing*, *318*, 297–305.

Vacha, L., & Barunik, J. (2012). Co-movement of energy commodities revisited: Evidence from wavelet coherence analysis. *Energy Economics*, *34*, 241–247.

Wang, M.-C., & Ye, J.-K. (2016). The relationship between covariance risk and size effects in emerging equity markets. *Managerial Finance*, *42*, 174–190.

Wang, W., Li, W., Zhang, N., & Liu, K. (2020). Portfolio formation with preselection using deep learning from long-term financial data. *Expert Systems with Applications*, *143*, 113042.

Wang, X., & Hui, X. (2017). Mutual information based analysis for the distribution of financial contagion in stock markets. *Discrete Dynamics in Nature and Society*, *2017*.

Yamanishi, K., & Takeuchi, J.-i. (2002). A unifying framework for detecting outliers and change points from non-stationary time series data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 676–681). ACM.

Zhang, M., Jiang, X., Fang, Z., Zeng, Y., & Xu, K. (2019). High-order hidden markov model for trend prediction in financial time series. *Physica A: Statistical Mechanics and its Applications*, *517*, 1–12.

Zhao, P., Hoi, S. C., & Jin, R. (2011). Double updating online learning. *Journal of Machine Learning Research*, *12*, 1587–1615.

Zhou, Z.-H. (2015). Ensemble learning. *Encyclopedia of biometrics*, (pp. 411–416).

Zhou, Z.-H., & Tang, W. (2003). Selective ensemble of decision trees. In *International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing* (pp. 476–483). Springer.

Zhou, Z.-H., Wu, J., & Tang, W. (2002). Ensembling neural networks: many could be better than all. *Artificial intelligence*, *137*, 239–263.