DEEP TEMPORAL LOGISTIC BAG-OF-FEATURES FOR FORECASTING HIGH FREQUENCY LIMIT ORDER BOOK TIME SERIES

Nikolaos Passalis[1,2]*, Anastasios Tefas*[1]*, Juho Kanniainen*[2]*, Moncef Gabbouj*[2]
and Alexandros Iosifidis[3]

[1]School of Informatics, Aristotle University of Thessaloniki, Greece
[2]Faculty of Information Technology and Communication Sciences, Tampere University, Finland
[3]Dept. of Engineering, Electrical and Computer Engineering, Aarhus University, Denmark
nikolaos.passalis@tuni.fi, tefas@csd.auth.gr,
{juho.kanniainen, moncef.gabbouj}@tuni.fi, alexandros.iosifidis@eng.au.dk

ABSTRACT

Forecasting time series has several applications in various domains. The vast amount of data that are available nowadays provide the opportunity to use powerful deep learning approaches, but at the same time pose significant challenges of high-dimensionality, velocity and variety. In this paper, a novel logistic formulation of the well-known Bag-of-Features model is proposed to tackle these challenges. The proposed method is combined with deep convolutional feature extractors and is capable of accurately modeling the temporal behavior of time series, forming powerful forecasting models that can be trained in an end-to-end fashion. The proposed method was extensively evaluated using a large-scale financial time series dataset, that consists of more than 4 million limit orders, outperforming other competitive methods.

Index Terms— Temporal Bag-of-Features, Limit Order Book, Time series forecasting

1. INTRODUCTION

Forecasting time series has several applications in various domains, ranging from predicting the behavior of financial markets [1], to energy load prediction [2]. The large amount of data that are continuously generated in many of these domains provide the opportunity to employ powerful deep learning methods, but at the same time pose significant challenges of high-dimensionality, velocity and variety. This led to the development of various methods for time series analysis and forecasting. Early approaches employed adaptive distance metrics, such as Dynamic Time Wrapping [3], while with the advent of deep learning, neural network-based methods, such as recurrent and convolutional models [4, 5], are increasingly used. Apart from these methods, other models, such as the Bag-of-Features model (BoF), was recently adapted toward efficiently processing large amounts of complex and high-dimensional time series [6, 7, 8], due its ability to hand objects consisting of a varying number of features, as well as withstanding distribution shifts better than competitive methods [9, 10].

The Bag-of-Features model (BoF) was initially proposed for handling images and involves the following pipeline [11]: First, a number of feature vectors are extracted from each input object (this is called *feature extraction* step). This way the *feature space* is formed. Then, a set of representative features, that can be used to quantize the extracted feature vectors, are learned. This step is called *dictionary learning*, while the representative features are usually called *codewords*. The codewords form the *dictionary* (also called *codebook*) of the BoF model. Finally, the quantized feature vectors are aggregated compiling a constant length histogram vector for each input object. The ability of the BoF model to handle objects of various lengths provides one important advantage over other methods, i.e., allows the BoF model to efficiently extract a constant length representation of time series regardless its actual length.

The main contribution of this work is the proposal of a novel deep learning formulation of the Bag-of-Features model adapted toward the needs of time series forecasting. The proposed method combines the aforementioned advantages of the BoF model with the enormous learning capacity of deep learning models, leading to the development of powerful forecasting models. However, using existing temporal BoF formulations, such as [9, 10], in complex deep learning architectures is not straightforward. First, the existing formulations usually require the use of sophisticated and computationally intensive initialization schemes, e.g., k-means. To overcome this limitation, a novel logistic formulation of the BoF model is proposed, allowing for directly training the resulting model without using sophisticated initialization schemes or tuning any hyper-parameter. The proposed method can be combined with additional deep information extraction layers, e.g., convolutional layers, demonstrating the ability to use the proposed logistic BoF formulation with deep neural networks.

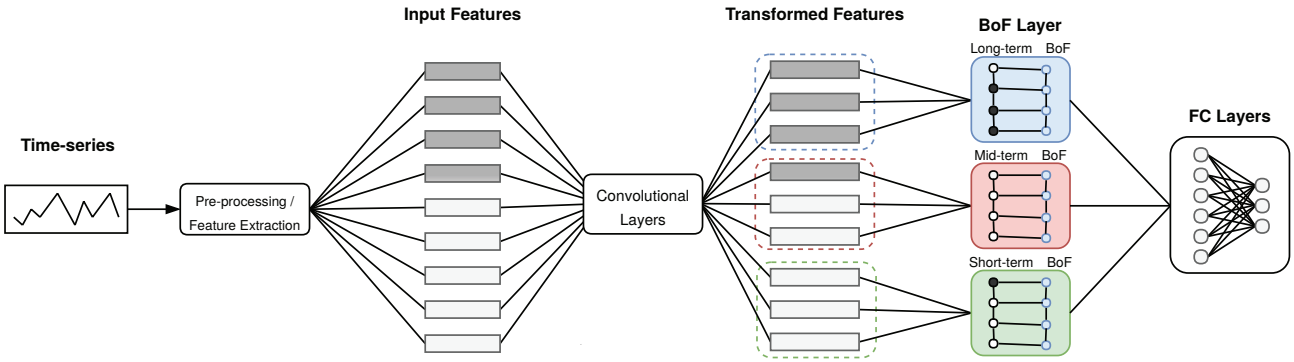

Fig. 1. The proposed deep temporal logistic BoF architecture for time series forecasting.

Furthermore, the proposed method introduces the ability to perform fine-grained temporal modeling, as shown in Figure 1, where the short-term, mid-term, and long-term behavior of time series are modeled. The proposed method is evaluated, for two different forecasting tasks, using a large-scale financial time series dataset that consists of more than 4 million limit orders.

The rest of the paper is structured as follows. First, the related work is briefly introduced and compared to the proposed approach in Section 2. Then, the proposed method is introduced in Section 3, while the experimental evaluation is provided in Section 4. Finally, conclusions are drawn in Section 5.

2. RELATED WORK

There is an increasing number of recent works in the literature that employ variants of the Bag-of-Features model to perform time series analysis, e.g., forecasting, retrieval, etc. In [12], a BoF-based method was proposed for extracting discriminative representations by employing a discriminative objective for the optimizing the codebook. A dictionary learning methods for the BoF model was also utilized in [13], in order to learn retrieval-oriented representations. In more advanced approaches, time series segments of various lengths were used, as in [6], to allow for efficiently handling warping, while an approach that employs temporal modeling was proposed in [7]. Quite recently, a neural formulation of the BoF model was used to perform time series analysis [14], while an extension of this method, that allows for better capturing the temporal dynamics of time series, was introduced in [8].

In contrast with [8], in this work we use a logistic BoF formulation that allows for training temporal BoF models without using any sophisticated initialization schemes. Also, the proposed method does not require the carefully tuning of any hyper-parameter, e.g., the initial scaling factor of the kernel function that was employed in [8]. Furthermore, the proposed BoF formulation was appropriately designed to allow for the smooth flow of information in deep architectures. To the best of our knowledge, this is the first work in which a deep temporal formulation of the BoF model is combined with convolutional feature extraction layers, demonstrating that it is indeed possible to learn powerful deep learning models for time series analysis.

3. PROPOSED METHOD

Let $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$ be a collection of N time series, where x_i denotes the i-th time series. Also, let $\mathbf{x}_{ij} \in \mathbb{R}^D$ be the j-th feature vector extracted from the i-th time series, where D is the dimensionality of the extracted feature vectors. Several different choices exist for extracting feature vectors from time series. The most straightforward one is to directly consider the set of measurements for each time step as a separate vector (in this case D denotes the dimensionality of the time series) [12]. However, more sophisticated methods do exist, e.g., using domain knowledge to design and extract features describing various aspects of the time series, e.g., as proposed in [15] for modeling high frequency limit order book data. Also, let N_i denote the length of i-th time series. Note that it is possible for different time series to have different lengths, since the BoF model can directly handle objects from which a varying number of feature vectors are extracted.

Next, the extracted vectors are fed to a sequence of 1-D convolutional layers, as show in Figure 1. The convolutional layers are employed to better model the temporal relationships between succeeding feature vectors. Let $f_{conv}(\cdot)$ denote the employed convolutional feature extractor that receives the input feature vectors $\mathbf{X}_i = [\mathbf{x}_{i,1}, \mathbf{x}_{i,2}, \ldots, \mathbf{x}_{i,N_i}]$ and extracts the k-th output feature vector after performing various convolutional operations. The resulting feature vector is denoted by $\mathbf{x}^c_{ik} = f_{conv}(\mathbf{X}_i, k) \in \mathbb{R}^{N_f}$, where N_f the number of filters used in the last convolutional layer. Therefore, after feeding the extracted feature vectors into the convolutional layers each time series is represented by the transformed feature set $\mathbf{X}^c_i = [\mathbf{x}^c_{i,1}, \mathbf{x}^c_{i,2}, \ldots, \mathbf{x}^c_{i,N_i}]$, assuming that the appropriate padding is used to ensure that N_i transformed feature vectors will be extracted. These transformed feature vectors also cap-

ture part of the temporal relationships between the original vectors $\mathbf{x}_{ij}$.

Then, the transformed feature vectors are aggregated using the proposed Temporal Logistic Bag-of-Features (abbreviated as "T-LoBoF"). First, the similarity between each transformed feature vector and each codeword $\mathbf{v}_k \in \mathbb{R}^{N_f}$ is measured as:

$$u_{ijk} = {\mathbf{x}_{ij}^c}^T \mathbf{v}_k \in \mathbb{R}. \quad (1)$$

To ensure that (1) encodes a proper similarity metric that can be used to quantize the extracted vectors into the employed codewords, an appropriate transformation function must be used. Several methods have been proposed to this end, e.g., using the absolute value of the inner product [16], or appropriately transforming the Euclidean distance between the feature vectors and the codewords using the Gaussian kernel [9]. However, the former approach leads to non-bounded similarity metrics, while the latter depends on the use of a Gaussian kernel for the similarity calculations that implies that a) the codewords must be carefully initialized (usually using the k-means algorithm) and b) the scaling factor of the kernel must be manually tuned. To avoid these drawbacks, as well as derive a method that works with the minimal amount of tuning and avoids the need for complicated initialization schemes, we propose transforming the inner product appearing in (1) using the *logistic* sigmoid ($\sigma(\cdot)$) function as follows:

$$u_{ijk} = \sigma({\mathbf{x}_{ij}^c}^T \mathbf{v}_k) \in (0, 1), \quad (2)$$

where $\sigma(x) = \frac{e^x}{e^x+1}$. The employed measure u_{ijk} is always bounded between 0 and 1 and expresses the similarity between the feature vector $\mathbf{x}_{ij}^c$ and the codeword $\mathbf{v}_k$. The normalized membership is obtained for each of these similarity values, allowing for quantizing the extracted feature vectors into the used codewords, using the following equation:

$$d_{ijk} = \frac{u_{ijk}}{\sum_{l=1}^{N_K} u_{ijl}} \cdot N_K, \quad (3)$$

where N_K is the number of used codewords. Note that the values of the normalized membership vector $\mathbf{d}_{ij} = [d_{ij1}, d_{ij2}, \ldots, d_{ijN_K}] \in \mathbb{R}^{N_K}$ sum to N_K instead of 1. This formulation is equivalent to the traditional l^1 normalization performed in such soft BoF formulations , e.g., [9, 8] (the values are simply scaled by a constant factor), while - at the same time - ensures the smooth flow of gradients in the resulting model. Indeed, scaling this vector by N_K ensures the proper information flow through the model [17]. Finally, the histogram vector that describes the distribution of the transformed feature vectors $\mathbf{x}_{ij}^c$ of the i-th time series is calculated as:

$$\mathbf{s}_i = N_S \cdot \frac{1}{N_i} \sum_{l=1}^{N_i} \mathbf{d}_{ij} \in \mathbb{R}^{N_K}, \quad (4)$$

where N_S is the number of feature vectors fed to the model during the training process. Again, we avoid using the usual l^1 normalization, that can cause instabilities during the training process, by scaling the resulting histogram by N_S. Note that the l^1 norm of $\mathbf{s}_i$ is kept constant, regardless the number of feature vectors fed to model during the training/inference, ensuring that the resulting formulation keeps the *length-invariance* property of BoF.

However, the histogram vector $\mathbf{s}_i$ describes the overall behavior of the time series through the time. To capture the fine-grained temporal dynamics, we propose segmenting the transformed feature vectors into N_T temporal regions. In Figure 1, $N_T = 3$ temporal regions are used corresponding to the short-term, mid-term, and long-term behavior of the time series. Therefore, the most recent $\lfloor \frac{N_i}{N_T} \rfloor$ feature vectors are employed for calculating the short-term histogram $\mathbf{s}_i^{short}$, the preceding $\lfloor \frac{N_i}{N_T} \rfloor$ feature vectors are used to calculate the mid-term histogram $\mathbf{s}_i^{mid}$, while the rest of the feature vectors are used for calculating the long-term histogram $\mathbf{s}_i^{long}$. Finally, the resulting concatenated vector $\mathbf{s}_i = [\mathbf{s}_i^{short}, \mathbf{s}_i^{mid}, \mathbf{s}_i^{long}] \in \mathbb{R}^{3N_K}$ is fed to the following fully connected layer, as shown in Figure 1.

The resulting architecture can be trained in an end-to-end fashion using gradient descent, i.e.,

$$\Delta(\mathbf{W}_{conv}, \mathbf{V}, \mathbf{W}_{fc}) = \eta(\frac{\partial \mathcal{L}}{\partial \mathbf{W}_{conv}}, \frac{\partial \mathcal{L}}{\partial \mathbf{V}}, \frac{\partial \mathcal{L}}{\partial \mathbf{W}_{fc}}), \quad (5)$$

where $\mathcal{L}$ is the employed loss function, $\mathbf{W}_{conv}$ denotes the parameters of the convolutional feature extractor $f_{conv}(\cdot)$, $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{N_K}]$ denotes the codebook used by the LoBoF model, while $\mathbf{W}_{fc}$ denotes the parameters of the fully connected layers. The cross-entropy loss is used for all the experiments conducted in this paper, while the same codebook is utilized for all the temporal regions. The Adam algorithm was used to perform the optimization [18]. The training time series were fed to the network in batches of 128 samples, where each time series was sampled with probability inversely proportional to the frequency of its class. Finally, the learning rate was set to $\eta = 10^{-4}$, while the networks were trained for 20 epochs.

4. EXPERIMENTAL EVALUATION

The proposed method was evaluated using a large-scale limit order book dataset [20]. The employed dataset consists of high frequency limit order book data collected from 5 Finish companies traded in the Helsinki Exchange (operated by Nasdaq Nordic). The 10 highest and lower ask order prices were collected for each time step, while data were collected over a period of 10 business days (1st June 2010 to 14th June 2010). A total of 4.5 million limit orders were gathered and processed according to the pre-processing and feature extraction pipeline proposed in [15]. Thus, a total number of 453,975 144-dimensional feature vectors were extracted.

The proposed algorithms were evaluated using an anchored evaluation setup [21]: The time series that were

Table 1. Evaluation results using the FI-2010 dataset (price direction prediction)

Method	Prediction Target	Macro-Precision	Macro-Recall	Macro-F1	Cohen's κ
MLP [8]	10	40.20 ± 0.50	56.25 ± 2.20	36.91 ± 1.81	0.1281 ± 0.0137
BoF [8]	10	39.26 ± 0.94	51.44 ± 2.53	36.28 ± 2.85	0.1182 ± 0.0246
N-BoF [8]	10	42.28 ± 0.87	61.41 ± 3.68	41.63 ± 1.90	0.1724 ± 0.0212
T-BoF [8]	10	43.85 ± 1.11	66.66 ± 3.40	43.96 ± 1.59	0.1992 ± 0.0201
WMTR [19]	10	$46.25 \pm$ *N/A*	$51.29 \pm$ *N/A*	$47.87 \pm$ *N/A*	*N/A*
T-LoBoF (raw)	10	46.34 ± 1.49	$\mathbf{69.40 \pm 3.61}$	48.98 ± 2.30	0.2538 ± 0.0306
T-LoBoF (conv)	10	$\mathbf{47.80 \pm 1.64}$	68.25 ± 4.56	$\mathbf{51.58 \pm 2.15}$	$\mathbf{0.2814 \pm 0.0309}$
MLP [8]	50	44.03 ± 1.25	52.67 ± 1.56	41.91 ± 2.33	0.1787 ± 0.0229
BoF [8]	50	42.56 ± 1.26	49.57 ± 2.28	39.56 ± 2.36	0.1576 ± 0.0254
N-BoF [8]	50	47.20 ± 1.80	58.17 ± 2.61	46.15 ± 4.07	0.2285 ± 0.0419
T-BoF [8]	50	49.58 ± 2.10	63.50 ± 2.54	49.82 ± 3.18	0.2723 ± 0.0348
T-LoBoF (raw)	50	50.48 ± 1.50	65.46 ± 2.77	51.42 ± 2.16	0.2900 ± 0.0269
T-LoBoF (conv)	50	$\mathbf{51.56 \pm 2.29}$	$\mathbf{65.81 \pm 4.32}$	$\mathbf{53.73 \pm 2.85}$	$\mathbf{0.3116 \pm 0.0403}$

extracted from the first day were used to train the model, while the data from the second day were employed for the evaluation. Then, the first two days were used for the training and the next day was used for the evaluation, etc. This process was repeated 9 times. For all the evaluated metrics (macro-precision, macro-recall, macro-F1 and Cohen's κ), the mean and standard deviation are reported. The direction of the average mid price (up, stationary or down) after 10 and 50 time steps were predicted. A stock was considered stationary if the change in the mid price was less than to 0.01% (or 0.02% for the prediction horizon of 50 time steps).

For each time step a time series that consists of the 15 more recently extracted feature vectors was compiled. The time series was segmented into $N_T = 3$ temporal regions, each consisting of 5 feature vectors. The employed convolutional feature extractor was composed of 256 1D convolutional filters (the size of the kernel was set to 5 and the stride was set to 1). A codebook composed of 256 codewords shared across the temporal models and two fully connected layers (with 512 and 3 neurons respectively) were used. The ReLU function was used both for the convolutional feature extractor and the first fully connected layer. Two variants of the proposed method were evaluated: a) using the aforementioned architecture as described and shown in Figure 1 (denoted by "T-LoBoF (conv)") and b) directly using the raw feature vectors without employing a convolutional layer (denoted by "T-LoBoF (raw)") .

The experimental results are shown in Table 1. The proposed methods are compared to various competitive approaches for forecasting financial time series, including the plain Temporal BoF ("T-BoF") approach [8] and the Weighted Multichannel Time series Regression (WMTR) method [19], as well as to various baselines. Several conclusions can be draw from the results reported in Table 1. First, the plain logistic BoF formulation (without using any convolutional feature extraction layers) performs significantly better than the competitive T-BoF method. Note that the T-LoBoF method is significant easier to use than the T-BoF, since it doesn't require any sophisticated initialization scheme or extensive hyper-parameter tuning. For example, the Cohen's κ rises from approx. 0.20 (T-BoF) to over 0.25, when the prediction horizon was set to the next 10 time steps. Furthermore, the proposed approach can be combined with convolutional feature extraction layers, significantly increasing the learning capacity of the model. Indeed, the proposed deep T-LoBoF formulation ("T-LoBoF (conv)"), outperforms all the other evaluated methods when combined with convolutional feature extraction layers.

5. CONCLUSIONS

In this paper, a novel deep learning formulation of the Bag-of-Features model that was adapted toward the needs of time series forecasting was presented. The proposed method can effectively combine the advantages of the BoF model with the great learning capacity of DL models. The proposed method employs a fully differential logistic formulation of the BoF model, allowing for directly training the resulting architecture in an end-to-end fashion, leading to powerful DL methods for time series analysis. Furthermore, the proposed method is capable of modeling the behavior of time series at various temporal levels. The proposed method was extensively evaluated and compared to other competitive methods using a large-scale financial time series dataset that consists of more than 4 million limit orders.

6. REFERENCES

[1] Li-Juan Cao and Francis Eng Hock 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.

[2] Henrique Steinherz Hippert, Carlos Eduardo Pedreira, and Reinaldo Castro Souza, "Neural networks for short-term load forecasting: A review and evaluation," *IEEE Transactions on Power Systems*, vol. 16, no. 1, pp. 44–55, 2001.

[3] Xiaopeng Xi, Eamonn Keogh, Christian Shelton, Li Wei, and Chotirat Ann Ratanamahatana, "Fast time series classification using numerosity reduction," in *Proceedings of the International Conference on Machine Learning*, 2006, pp. 1033–1040.

[4] Zachary C Lipton, David C Kale, Charles Elkan, and Randall Wetzell, "Learning to diagnose with lstm recurrent neural networks," *arXiv preprint 1511.03677*, 2015.

[5] Zhicheng Cui, Wenlin Chen, and Yixin Chen, "Multi-scale convolutional neural networks for time series classification," *arXiv preprint arXiv:1603.06995*, 2016.

[6] Mustafa Gokce Baydogan, George Runger, and Eugene Tuv, "A bag-of-features framework to classify time series," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2796–2802, 2013.

[7] Adeline Bailly, Simon Malinowski, Romain Tavenard, Thomas Guyet, and Laetitia Chapel, "Bag-of-temporal-sift-words for time series classification," in *ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data*, 2015.

[8] Nikolaos Passalis, Anastasios Tefas, Juho Kanniainen, Moncef Gabbouj, and Alexandros Iosifidis, "Temporal bag-of-features learning for predicting mid price movements using high frequency limit order book data," *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2018.

[9] Nikolaos Passalis and Anastasios Tefas, "Learning bag-of-features pooling for deep convolutional neural networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[10] Nikolaos Passalis and Anastasios Tefas, "Training lightweight deep convolutional neural networks using bag-of-features pooling," *IEEE Transactions on Neural Networks and Learning Systems*, 2018.

[11] Josef Sivic, Andrew Zisserman, et al., "Video google: A text retrieval approach to object matching in videos.," in *Proceedings of the IEEE International Conference on Computer Vision*, 2003, vol. 2, pp. 1470–1477.

[12] Alexandros Iosifidis, Anastasios Tefas, and Ioannis Pitas, "Multidimensional sequence classification based on fuzzy distances and discriminant analysis," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 11, pp. 2564–2575, 2013.

[13] Nikolaos Passalis and Anastasios Tefas, "Entropy optimized feature-based bag-of-words representation for information retrieval," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 7, pp. 1664–1677, 2016.

[14] Nikolaos Passalis, Avraam Tsantekidis, Anastasios Tefas, Juho Kanniainen, Moncef Gabbouj, and Alexandros Iosifidis, "Time-series classification using neural bag-of-features," in *Proceedings of the European Signal Processing Conference*, 2017, pp. 301–305.

[15] Alec N. Kercheval and Yuan Zhang, "Modelling high-frequency limit order book dynamics with support vector machines," *Quantitative Finance*, vol. 15, no. 8, pp. 1315–1329, 2015.

[16] Nikolaos Passalis and Anastasios Tefas, "Concept detection and face pose estimation using lightweight convolutional neural networks for steering drone video shooting," in *Proceedings of the European Signal Processing Conference*, 2017, pp. 71–75.

[17] Xavier Glorot and Yoshua Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[18] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," in *Proceedings of the International Conference on Learning Representations*, 2015.

[19] Dat Thanh Tran, Martin Magris, Juho Kanniainen, Moncef Gabbouj, and Alexandros Iosifidis, "Tensor representation in high-frequency financial data for price change prediction," in *Proceedings of the IEEE Symposium Series on Computational Intelligence*, 2017, pp. 1–7.

[20] Adamantios Ntakaris, Juho Kanniainen, Moncef Gabbouj, and Alexandros Iosifidis, "Mid-price prediction based on machine learning methods with technical and quantitative indicators," *SSRN*, 2018.

[21] Emilio Tomasini and Urban Jaekle, *Trading Systems*, Harriman House Limited, 2011.