Neural Computing and Applications (2022) 34:19441–19452
https://doi.org/10.1007/s00521-022-07509-6

S.I.: DEEP LEARNING MODELLING IN REAL LIFE: (ANOMALY DETECTION, BIOMEDICAL, CONCEPT ANALYSIS, FINANCE, IMAGE ANALYSIS, RECOMMENDATION)

Multisource financial sentiment analysis for detecting Bitcoin price change indications using deep learning

Nikolaos Passalis[1] · Loukia Avramelou[1] · Solon Seficha[1] · Avraam Tsantekidis[1] · Stavros Doropoulos[2] · Giorgos Makris[2] · Anastasios Tefas[1]

Received: 2 December 2021 / Accepted: 1 June 2022 / Published online: 3 July 2022

Abstract

The success of deep learning (DL) in various areas, such as computer vision, fueled the interest in several novel DL-enabled applications, such as financial trading, which could potentially surpass the previously used approaches. Indeed, there has been a plethora of DL-based trading methods proposed in recent years. Despite the success of these methods, they typically rely on a very restricted set of information, usually employing only price-related information. As a result, they ignore sentiment-related information, which can have a profound impact and be a strong predictor of various assets, such as cryptocurencies. The contribution of this paper is multifold. First, we examine whether the use of sentiment information, as extracted by various online sources, including news articles, is beneficial when training DL agents for trading. Then, given the difficulty of training reliable sentiment extractors for financial applications, we evaluate the impact of using different DL models as sentiment extractors, as well as employ an unsupervised training pipeline for further improving their performance. Finally, we propose an effective multisource sentiment fusion approach that can improve the performance over the rest of the evaluated approaches. The conducted experiments have been performed using several different configurations and models, ranging from multilayer perceptrons (MLPs) to convolutional neural networks (CNNs) and recurrent neural networks (RNNs), to provide a reliable evaluation of sentiment-aware DL-based trading strategies providing evidence that sentiment information might be a stronger predictor compared to the information provided by the actual price time series for Bitcoin.

Keywords Financial trading · Sentiment analysis · Deep learning · Sentiment-aware trading

1 Introduction

The large number applications of deep learning (DL) in various areas [14, 20, 21] fueled the interest in developing intelligent agents for financial trading [34, 35, 37], which could lead to better performance in many cases, compared to traditional methods, such as rule-based strategies. As a result, a series of powerful DL formulations proposed in the literature led to models with enormous learning capacity. At the same time, their ability to seamlessly

✉ Nikolaos Passalis
passalis@csd.auth.gr

Loukia Avramelou
avramelou@csd.auth.gr

Solon Seficha
sseficha@csd.auth.gr

Avraam Tsantekidis
avraamt@csd.auth.gr

Stavros Doropoulos
doro@datascouting.com

Giorgos Makris
gmakris@datascouting.com

Anastasios Tefas
tefas@csd.auth.gr

[1] Computational Intelligence and Deep Learning Group, AIIA Lab, Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece

[2] DataScouting, Thessaloniki, Greece

integrate with reinforcement learning (RL) methodologies allowed for directly optimizing trading policies to maximize the expected profit, even in the volatile and uncertain conditions that often exist in real markets [10, 15, 32]. However, current approaches operate on a restricted set of input information, i.e., they mainly rely on time-series information regarding the price of assets. This can enforce an upper limit on the performance of such approaches since this contrasts with the information that is usually available to human traders. Indeed, human traders, apart from observing price-related information, also take into account their prior knowledge, the sentiment that is expressed regarding various markets and assets, as well as general news and forecasts.

Contrary to this, trading using DL is typically tackled as a problem that can be solved solely by relying on a single modality, i.e., price time series, without taking into account any additional external information. Indeed, the additional complexity and development costs regarding collecting the appropriate data, preprocessing them, and then transforming them into a form that can be exploited by DL models often discourage DL researchers and companies from exploiting these valuable sources of information. Recent evidence suggests that external information, mainly provided in the form of sentiment regarding various financial assets [8, 9, 26, 36] and typically collected from social media, often has a positive effect on the accuracy of trading agents. However, little work has been done so far in this direction, especially for exploiting large-scale datasets that contain news articles regarding financial assets, while many important questions arise when designing such a system. For example, how does the accuracy of the model trained to extract sentiment-related information can affect the performance of the subsequent DL model? Is it possible to train DL-based sentiment extractors that are tailored especially to financial tasks without spending too much effort in annotating financial data sources? Can we combine multiple sentiment extraction models to have a more reliable way of estimating sentiment for financial trading?

In this work, we provide an extensive experimental study, along with a simple, yet effective way to combine sentiment extracted from multiple sources, to answer these research questions. More specifically, the contribution of this work is multifold. First, we examine whether the use of sentiment information, as extracted from various online sources, including news articles, is beneficial when training DL agents for trading. Then, we proceed by collecting a large pool of data from online sources related to cryptocurriencies and perform unsupervised training in a BERT architecture [11], to further adapt it for the task of financial-aware sentiment extraction, while we also evaluate the impact of using different models for extracting sentiment from financial documents. Finally, we propose a simple, yet effective way to fuse the information extracted from different sentiment extractors, demonstrating that such multisource strategy can indeed increase the profitability of the resulting models. To provide a reliable evaluation, we performed experiments using a wide variety of deep learning models, ranging from multilayer perceptrons (MLPs) to convolutional neural networks (CNNs) and recurrent neural networks (RNNs). To this end, we go beyond the existing literature that typically just evaluates a few handpicked models, with and without sentiment information, and we provide an extensive evaluation, often including more than 50 different configurations per architecture. The experimental results confirm our initial hypothesis regarding the impact of sentiment for financial trading, demonstrating that for cryptocurriences, such as Bitcoin, sentiment information can be a strong predictor of future price movements, while also demonstrating that using multiple sentiment sources can also lead to improved trading performance.

The rest of the paper is structured as follows. First, related work is introduced in Sect. 2. Then, we present the used notation and analytically describe the proposed method in Sect. 3. Next, we provide an extensive experimental evaluation in Sect. 4. Finally, Sect. 5 concludes the paper and discusses possible future research directions.

2 Related work

There is a vast amount of recent works that focus on developing intelligent agents for financial trading based solely on price-related information. Indeed, the development of Deep Learning (DL) enabled automated agents allows for exploiting the vast amount of data collected from financial markets, outperforming to a significant degree the methods used until then [5, 12, 28, 29, 37]. A wide range of different methods have been used to this end, ranging from supervised learning approaches using convolutional neural networks (CNNs) and long short-term memory (LSTM) recurrent neural networks (RNNs) [18, 19] to deep reinforcement learning (DRL) methodologies [10, 32]. Even though these approaches can lead to very promising results, they do not take into account the sentiment that is expressed in various media. The method proposed in this paper on the other hand focuses on exploiting sentiment information on top of features that can be extracted from historical price data.

Recent works have demonstrated that taking into account external information, mainly provided in the form of sentiment regarding various financial assets [8, 9, 26, 36], can significantly boost the performance of trading agents. At the same time, the development of powerful large-scale language models that can be

trained on large collections of text documents, such as BERT [11] and roBERTA [6, 7, 16], which can be fined-tuned on various tasks, such as sentiment analysis, provided additional powerful tools for automating sentiment extraction from online sources. Indeed, it has been demonstrated that such information can be very useful for developing trading agents [23, 26]. However, these approaches typically ignore that generic sentiment extractors, even when trained on large document collections, face significant challenges when used in domain-specific areas, such as finance [4]. In this work, we demonstrate that unsupervised pretraining can have a significant impact on sentiment analysis for the financial domain, for both sentiment prediction and when this information is subsequently used for training. Furthermore, we also demonstrate, for the first time to the best of our knowledge, which using multiple sentiment extractors can provide additional information to the subsequent DL models, further improving their trading performance.

Finally, note that this paper is an extended version of our preliminary work presented in [23]. In this version, we provide a more thorough and extensive evaluation, since we have collected an additional dataset, which contains over 800,000 documents, to evaluate the impact of using unsupervised pre-training, evaluated the impact of additional sentiment extraction models, as well as proposed a multisource fusion strategy for using different sentiment sources, further improving the trading performance.

3 Proposed method

In this section, we introduce the proposed data processing and fusion pipeline, as well as the employed financial forecasting setup. For the rest of this section, we assume that both the forecasting and the sampling time step is set to one day. This is without loss of generality since the proposed method can be trivially extended to work with longer or smaller time horizons, given that the appropriate data are collected.

The proposed data processing pipeline, along with the forecasting model, are shown in Fig. 1. First, the DL model receives the raw price candles from a financial data source, e.g., an exchange. These data are then preprocessed in order to obtain a single scalar value for each time step t that corresponds to the percentage change of the price of an asset:

$$p_t = \frac{c_t}{c_{t-1}} - 1, \tag{1}$$

where c_t denotes the close price at time t. It is worth noting that this is among the most well-established financial data preprocessing approaches for extracting stationary features [25, 31]. Then, these percentage changes are aggregated into a window of length L to form a vector that describes the price behavior of a specific asset during the last L steps:

$$\mathbf{x}_t^p = [p_{t-L-1}, \ldots, p_t] \in \mathbb{R}^L. \tag{2}$$

Note that we use the notation $\mathbf{x}_t^p$ to refer to the vector that contains the *history* of the previous L percentage changes at time t, while we use the notation p_t to refer to the scalar percentage change at time t.

In this work, we propose to also employ sentiment information about a financial asset, as expressed in various online sources, to extract additional information that can be useful for predicting the future behavior of the said asset. Let $\mathcal{X}_t$ denote a collection of textual documents that refer to the asset at hand and collected at time t, i.e., after time step $t-1$ and until time step t. Also, let $f_s(\mathbf{x}_d)$ denote a sentiment extractor that returns the sentiment of a document $\mathbf{x}_d$, where $\mathbf{x}_d$ is an appropriate representation of a textual document for the task of sentiment analysis, e.g., a sequence of the words that appear in the corresponding document [4]. Sentiment can be either represented as a scalar value that ranges from -1 (negative sentiment) to $+1$ (positive) sentiment, or as a three-valued vector that expresses the confidence that a text has positive, neutral, and negative sentiment, respectively. Therefore, for each time step we can extract the *polarity vector* regarding the asset at hand as follows:

$$\mathbf{s}_t = \frac{1}{|\mathcal{X}_t|} \sum_{\mathbf{x}_d \in \mathcal{X}_t} f_s(\mathbf{x}_d) \in \mathbb{R}^3, \tag{3}$$

where $|\mathcal{X}_t|$ denotes the number of text documents collected at time step t. Note that the three values described above (positive, negative, neural) are extracted unless otherwise noted. Then, we can similarly define the time series that describes the sentiment over a horizon of L time steps as:

$$\mathbf{x}_t^s = [\mathbf{s}_{t-L-1}, \ldots, \mathbf{s}_t] \in \mathbb{R}^{L \times 3}. \tag{4}$$

The most straightforward way to combine the price information ($\mathbf{x}_t^p$), with the sentiment information ($\mathbf{x}_t^s$), is to simply concatenate the corresponding vectors into a tensor:

$$\mathbf{x}_t = [\mathbf{x}_t^p; \mathbf{x}_t^s] \in \mathbb{R}^{L \times 4}. \tag{5}$$

Then, this tensor is fed to the corresponding DL model, as shown in Fig. 1. Furthermore, in this work, we propose using multiple sentiment extraction models to acquire a more robust estimation of the sentiment. Using just one model to extract the sentiment can often lead to a noisy estimation of the sentiment time series, especially when we lack enough data to reliably estimate the sentiment. Indeed, this is demonstrated in Fig. 2 where two different sentiment extraction models lead to a significantly different

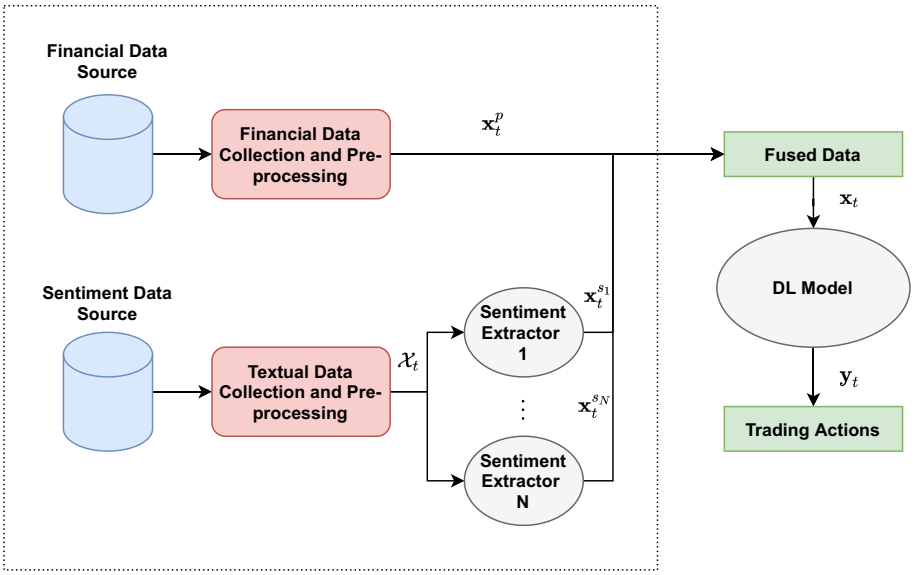

Fig. 1 Proposed trading pipeline: The trained DL models rely on two different information sources: **a** financial data sources, which provide price information, as well as **b** sentiment data sources, which provide sentiment information. Note that multiple sentiment extractors can be used to provide a more robust sentiment estimation. The collection of textual documents collected at time t is denoted by $\mathcal{X}_t$, the history of the L most recent price percentage changes is denoted by $\mathbf{x}_t^p$, while $\mathbf{x}_t^{s_i}$ refers to the sentiment time series extracted by the i-th sentiment extractor and $\mathbf{x}_t$ refers to the vector fed to the DL model that contains all the extracted information. The fused data $\mathbf{x}_t$ are then fed to a DL model that provides the trading signals $\mathbf{y}_t$

distribution of sentiment over the same time horizon. Motivated by this observation, along with recent findings in financial trading where it has been demonstrated that using diversification strategies when developing trading agents can improve their profitability [33], we propose employing multiple models for extracting the sentiment, while also ensuring that each of these models will provide a different "view" of the data. Intuitively, this can be thought of as the process of asking several "experts" regarding their opinion about the sentiment in financial markets and then using this information in another model to predict the price trend. Therefore, using a diversification strategy aims to maximize the amount of information that will be available to the subsequent DL model that is used for trading. There are several ways to generate the models that can be used for estimating the sentiment. For example, we can train the same model using different initializations, following well-known strategies used in DL model ensembling [17, 24, 38]. In this work, we opt for a diversification strategy, motivated by the findings reported in [33] for developing trading agents that rely solely on price information. More specifically, we train the same model using different information sources to provide complementary information to the subsequent DL model. More details for the used information are provided in Sect. 4. Therefore, the multidimensional sentiment time series can be extended to include the estimation from all the sentiment extractors simply by concatenating the information from the additional sentiment sources. For example, when using two sentiment extractors, the final tensor provided in (5) will be constructed as:

$$\mathbf{x}_t = [\mathbf{x}_t^p; \mathbf{x}_t^{s_1}; \mathbf{x}_t^{s_2}] \in \mathbb{R}^{L\times 7}, \tag{6}$$

where $\mathbf{x}_t^{s_1}$ and $\mathbf{x}_t^{s_2}$ denote the first and second sentiment time series.

Several different approaches have been proposed in the literature for training DL models for financial trading, ranging from classification-based methods [29] to complex reinforcement learning setups which aim to simulate the trading environment [10]. In this work, we opt for the following classification-based setup, where a DL model is trained to predict the price movements that are more likely to lead to profit. More specifically, the ground labels for training the DL model are generated as:

$$l_t = \begin{cases} 1 & \text{if } \frac{c_{t+1}}{c_t} - 1 > c_{thres} \\ -1 & \text{se } \frac{c_{t+1}}{c_t} - 1 < -c_{thres} \\ 0 & \text{otherwise,} \end{cases} \tag{7}$$

where c_{thres} denotes the threshold for considering that a price movement is a potential candidate for performing a profitable trade. Therefore, the label "1" corresponds to a long position, i.e., the price is expected to increase and the agent should buy the asset to make a profit when the prices increases, while the label "-1" corresponds to a short position, i.e., the price is expected to decrease and the agent should borrow the asset and sell it to make a profit

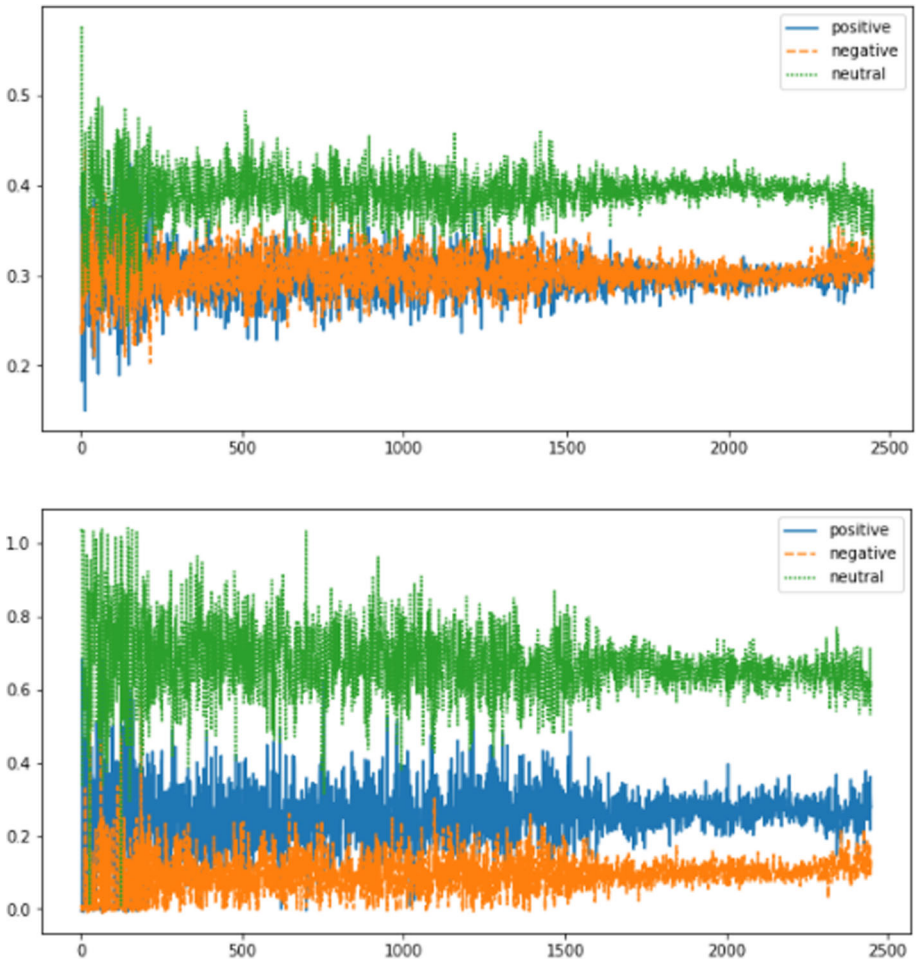

Fig. 2 Sentiment time series for the financial sentiment data source used for training and evaluating the models. The upper three time series were extracted using a BERT model, while the lower three time series were extracted using a CryptoBERT model. Please refer to Sect. 4 for more details regarding the exact experimental setup used

when the price decreases. The label "0" indicates market conditions that probably do not allow the specific agent to perform profitable trades, i.e., the agent should exit the market. Typically, c_{thres} is set to a value high enough to overcome any commission fees, as well as to account for price slippage that might occur. Please note that during backtesting, the consecutive "long" or "short" positions do not lead to multiple commissions (since the agent simply keeps the already existing position open), while the exit position ("0") closes the currently open position and materializes any gain/loss acquired.

After generating the labels, the DL model can be directly trained using the cross entropy loss, i.e.,

$$\mathcal{L} = -\frac{1}{N}\sum_{t=1}^{N}\sum_{j=1}^{3}[\mathbf{l}_t]_j \log([g_{\mathbf{W}}(\mathbf{x}_t)]_j), \tag{8}$$

where $g_{\mathbf{W}}(\cdot)$ denotes the DL model employed for 3-way classification, $\mathbf{l}_t$ is the one-hot encoding of l_t, the notation $[\mathbf{x}_t]_j$ is used to refer to the j-th element of a vector $\mathbf{x}_t$, and N is the total number of time steps for the training time series, assuming that the time series is continuous. Then, the model can be readily trained using gradient descent, i.e.,

$$\mathbf{W}' = \mathbf{W} - \eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}}, \tag{9}$$

where $\mathbf{W}$ denotes the parameters of the model $g_{\mathbf{W}}(\cdot)$. In this work, mini-batch gradient descent is used, while the Adam algorithm is employed for the optimization [13]. Please also note that among the main aims of this work is to evaluate whether using sentiment information can have a positive impact on the trading performance of a DL agent. To this end, we used three different models, i.e., a) a multilayer perceptron (MLP) (after appropriately flattening the input tensor into a vector), b) a 1-D convolutional neural network, and c) a long short-term memory (LSTM)-based network. All of these network architectures are widely used for training agents that can provide trading signals [29, 30, 37]. As we explain in detail in Sect. 4, we performed several experiments to evaluate the impact of using sentiment information on trading for a wide range of different setups and architectures, including models pre-trained on large-scale financial datasets, as well as fusing information for multiple sentiment extractors.

4 Experimental evaluation

In this section, we provide the experimental evaluation of the proposed sentiment-aware trading pipeline. First, we introduce the employed setup and hyper-parameters used for the conducted experiments, as well as the datasets used as a source of price and sentiment information for the conducted experiments. Then, we present and discuss the experimental results, evaluating the validity of all hypotheses presented in Sect. 1.

4.1 Data and experimental setup

Regarding the financial data source, we use the daily close prices for Bitcoin-USD (United States Dollar) currency pair. This dataset is plotted in Fig. 3. For extracting sentiment information for the same period of time, we used a dataset published by BDC Consulting [1], which contains over 200,000 titles of financial articles collected from various sites that publish articles on cryptocurrencies, such as Cointelegraph and CoinDesk. This dataset provides data for 5 years, from 2015 to 2020. The sentiment extracted using the FinBERT model [4] is shown in Fig. 4 for this dataset. Therefore, we used the first four years for training the DL models (2015-2019), while the last year (2019-2020) was used for performing the evaluation/backtesting of the trading agents. For both the training and testing datasets, we carefully aligned the textual data and price data, using the corresponding timestamps to ensure that no information from the future can leak into each training window.

Furthermore, in this work, we employed two additional datasets. The first one was an annotated textual dataset from financial sources (without aligned price information) used for training a BERT model [2, 3]. This dataset used for the sentiment analysis task consists of documents related to financial and cryptocurrency topics along with its labels characterizing the sentiment it expresses (positive, negative, and neutral) [2, 3]. More specifically, different types of documents, such as news and tweets, as well as financial documents were used. In addition, we

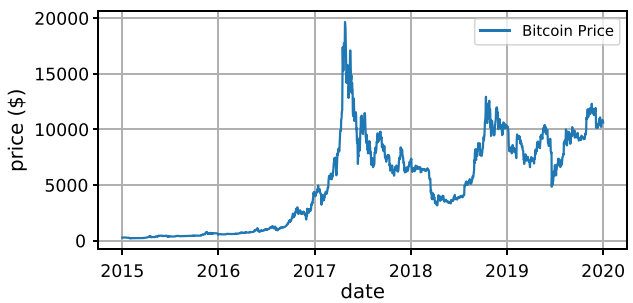

Fig. 3 BTC-USD price during the period 2015-2020

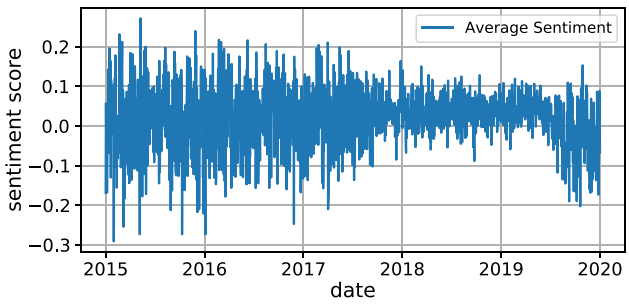

Fig. 4 Average sentiment score per day, as expressed by the documents contained in the BDC Consulting dataset. The finBERT model was used for extracting the sentiment of the titles of news articles published each day. Note that -1 corresponds to the most negative sentiment, while 1 corresponds to the most positive sentiment

preprocessed the documents to clean them from tags, links, or symbols. Thus, the dataset contains 119,286 annotated samples of which 41,738 express positive sentiment, 36,528 express negative sentiment, and 40,999 are neutral. Moreover, the dataset was further divided into training, testing, and validation sets. In more detail, we divided the dataset into the training set and testing set at 80% and 20%, respectively. Then, we divided the training set into the final training set and validation set at 90% and 10%, respectively. Therefore, we have 85,885 samples in the final training set, 23,858 samples in the testing set, and 9,543 samples in the validation set. This dataset was used for supervised training of the BERT-based models, as well as for evaluating their accuracy in sentiment analysis. Two models were trained using this dataset. The first one is denoted by "BERT" and was simply trained on the sentiment analysis downstream task. The second one is denoted by "CryptoBERT." For this model, we first followed an unsupervised pre-training procedure, using the dataset described below.

We also collected a dataset from various online data sources using crypto-related keywords, such as Bitcoin and Ethereum. The following online sources were used:

`twitter.com, telegram.com, fool.com, bloomberg.com, livemint.com, cryptoslate, reuters.com, coinspeaker.com, cryptobriefing.com, forexcrunch.com, news.bitcoin.com, mckinsey.com, coinbase.com, financialpost.com, ledgerinsights.com, cnbc.com, criptonoticias.com, themarket.co.uk, investing.com, crypto-newsflash.com, coindesk.com, axios.com, dailyhodl.com, societegenerale.com, nbcchicago.com, newsbtc.com, morningporridge.com,`

`cointelegraph.com`, `reddit.com`, and `insights.deribit.com`.

The collection process spanned over a 6-month period, with collection systems running mostly non-stop. This has resulted in the collection of 154,481 web articles, 570,865 tweets, and 90,268 telegram posts. All collected items are dated between 2015 and 2021. The used sources are split into two categories, those which are being handled by a generic web scraper and those that require a specialized scrapper to work with the site's API. Most of the websites mentioned earlier fall into the first category with the exceptions being Twitter and Telegram. In those two cases, we implemented a specialized scrapper that can fully utilize their APIs. Collecting older articles has been far more challenging than collecting current ones. Traversing websites to find older articles, if no archive is provided, is done as a depth-first search, with fixed depth so as to not impose too heavy a load on the content provider. Most websites do utilize a form of article suggestion mechanism that is biased towards more recent articles, therefore making it harder to discover older ones.

The main motivation of this work is to evaluate the impact of using sentiment information across a wide range of DL models and configurations. To this end, we did not limit the evaluation to a smaller number of handpicked DL models. Instead, we evaluated a wide range of models for different hyper-parameters, including a different number of layers, neurons per layer, learning rates, and dropout rates. More specifically, for the MLP model we evaluated models with 1, 2, and 3 layers and 8, 16, 32, 64, and 128 neurons per layer. For the CNN models, we evaluated models with 1, 2, and 3 convolutional layers (all followed by a final classification layer) and 4, 8, and 16 filters per layer and kernel sizes equal to 3, 4, and 5. Finally, for the LSTM models, we experimented with 1, 2, and 3 layers and 8, 16, 32, 64, and 128 neurons per LSTM layer. For all the configurations we used the Adam optimizer [13]. Therefore, we trained and evaluated the models with three different learning rates, i.e., 10^{-2}, 10^{-3}, and 10^{-4}. For the experiments using the FinBERT model, we also used different dropout rates for the layers [27], i.e., 0.1, 0.2, and 0.4, but we concluded that the effect is minimal, so we did not include dropout in the subsequent experiments. Also, for FinBERT, we used a single-dimensional sentiment time series, while three-dimensional time series (corresponding to "positive," "neutral," and "negative" sentiment) were extracted from the rest of the models. All possible model configurations that were produced by the different combinations of the aforementioned parameters were trained and evaluated.

4.2 Experimental evaluation

First, we performed an initial set of experiments using the FinBERT model, in order to evaluate whether the use of sentiment information can be beneficial in financial trading. To this end, we examined the average performance of different configurations for three different kinds of inputs: a) price alone, b) sentiment alone, and c) combined price and sentiment. The evaluation results for the test set are provided in Table 1, where we compare the average profit and loss (PnL) metric [31], which allows us to estimate the expected profit and/or loss of a trading agent over a specific period of time. PnL is calculated as

$$PnL = \sum_{t=1}^{N} \delta_t p_t - |\delta_t - \delta_{t-1}|c, \tag{10}$$

where N denotes the total duration of the backtesting period (number of time steps), p_t is the return at time step t as provided in (1), c is the commission paid for realizing profits/losses and δ_t is an index variable used to indicate the current position, which is defined as:

$$\delta_t = \begin{cases} -1, \text{if agent holds a short position at time step } t \\ 1, \text{if agent holds a long position at time step } t \\ 0, \text{if the agent is not in the market at time step } t \end{cases}. \tag{11}$$

Note that we define $\delta_0 = 0$ and higher PnL values indicate higher profit (better performance). We report the average over the top-50 performing configurations, in order to ensure a fair comparison between the different models. Using sentiment information alone provides better PnL compared to just using the price while combining the price and sentiment together allows for slightly improving the obtained results.

These results are also confirmed in the evaluation performed for the training set for individual agents, as provided in the left column of Fig. 5, where we also examine the convergence speed of the models by evaluating three different snapshots of the agents, i.e., at epoch 100, 200,

Table 1 Average percentage (%) profit and loss (PnL) for the 50 top-performing configurations for each model (backtesting performed on the test set, i.e,. 2019-2020). The prediction horizon was set to 1 day. The lot size used is constant for the whole duration of the backtest regardless of accumulated profits or losses

Input Modality	MLP (%)	CNN (%)	LSTM (%)
Price	201	219	214
Sentiment	221	**228**	222
Price and sentiment	**224**	**228**	**224**

Bold values indicate best performance for each group

and 300. Using price alone leads to a PnL of about 7. On the other hand, the obtained results clearly demonstrate that the DL models learn significantly faster when sentiment information is available since there are very small differences between the three model snapshots (i.e., epochs 100, 200, and 300) and the final training PnL reaches values over 30. This result demonstrates that sentiment information for cryptocurrencies, such as Bitcoin, might actually be a stronger predictor of its future behavior compared to the information provided by the price time series. Combining price and sentiment information together shows a bit mixed result, possibly limiting overfitting issues that might occur when sentiment is used, since the maximum train PnL, in this case, is around 20, while the models converge slower compared to only using sentiment input.

Indeed, similar results are obtained for the test evaluation, where the trained DL models are evaluated on unseen test data, as shown in the right column of Fig. 5. The models that were trained using sentiment information consistently perform better compared to the corresponding models that were trained only using price information as input. Combining price and sentiment information seems to lead to slightly better behavior. Therefore, the obtained results confirmed our initial hypothesis that taking into account sentiment information can lead to agents that perform consistently better trades since in all evaluated cases using sentiment information as input increased the obtained PnL.

After this set of experiments, we proceeded to evaluate whether the unsupervised training of a BERT architecture can increase the accuracy of sentiment analysis. The results are shown in Table 2, where the term "BERT" is used to refer to the supervised training of a BERT model without pre-training, while the term "CryptoBERT" is used for the proposed pretrained architecture using the collected dataset. We also report results for two different cases: a) finetuning of the classification layer only and b) training of the whole architecture.The benefit of the unsupervised pre-training is especially evident in the case where only the classification layer is trained since in this case, the accuracy increases by 8%. On the other hand, the improvements are smaller (1%) when the whole model is trained in an end-to-end fashion. Note that the dataset used for supervised training and evaluation contains only documents related to finance, which explains the observed positive impact of unsupervised pre-training. We also compared the proposed approach to other state-of-the-art large-scale language models, i.e., a) the roBERTA-base (TweetEval) [6], which was trained on 58M tweets and then finetuned for sentiment analysis on TweetEval dataset, as well as b) the XLM-roBERTa-base (multilingual) [7], which was trained on 198M tweets and finetuned for sentiment analysis in a multilingual dataset. Again, the benefits of the proposed unsupervised pre-training and finetuning in the financial domain are evident, since these methods, despite being trained on significantly larger datasets, achieve lower accuracy compared to the proposed one (60% vs. 90%).

We also conducted a qualitative evaluation where we compared the output of the best performing large-scale model (roBERTa-base (TweetEval)) to the proposed CryptoBERT model. The results of this evaluation are provided in Table 3. It is evident that in all of the presented cases the proposed method indeed captures the correct sentiment with higher confidence compared to the roBERTa-base model. The generic roBERTa-base model tends to classify most documents to the neutral class. This behavior can be attributed to using generic datasets for training such models that probably lack the necessary domain-specific knowledge required for this task. Quite interestingly, the roBERTa-base model can also misclassify neutral sentences as negative with quite high confidence and without an apparent reason for this decision. Overall, the proposed method tends to be less confident, yet classifies a larger number of documents correctly.

Next, we evaluated these two models, i.e., BERT and CryptoBERT, as sentiment sources for financial trading. The experimental results are reported in Table 4. Note that we report both the price direction accuracy (%) and the acquired profit and loss. First, note that in all cases the accuracy of the models increases when the two modalities are combined. In most the cases, this also translates into an increase in the observed PnL. The slight discrepancy between these two quantities is expected, since the model is not directly optimized to maximize the PnL, e.g., using reinforcement learning [32]. Furthermore, note that CryptoBERT does not consistently lead to improved accuracy or PnL. However, there are some cases, where despite having lower accuracy, it can achieve higher PnL. This can be potentially attributed to its better ability to correlate significant price movements to the corresponding sentiment.

Combining sentiment and price improves the expected performance for both BERT and CryptoBERT. Therefore, it is not clear which model should be preferred. The sentiment extracted using both of these models is shown in Fig. 2, where the differences between the sentiment extracted by these models are depicted. For example, CryptoBERT leads to a much more clear distinction between positive and negative sentiment, while the positive sentiment is the prevalent one. At the same time, we can observe that the sentiment movements are correlated at various points of the sentiment time series extracted by these two models. Based on these observations, we repeated the experimental evaluation by using the sentiment time series extracted by both models. The experimental results reported in Table 5 demonstrate that in most of the

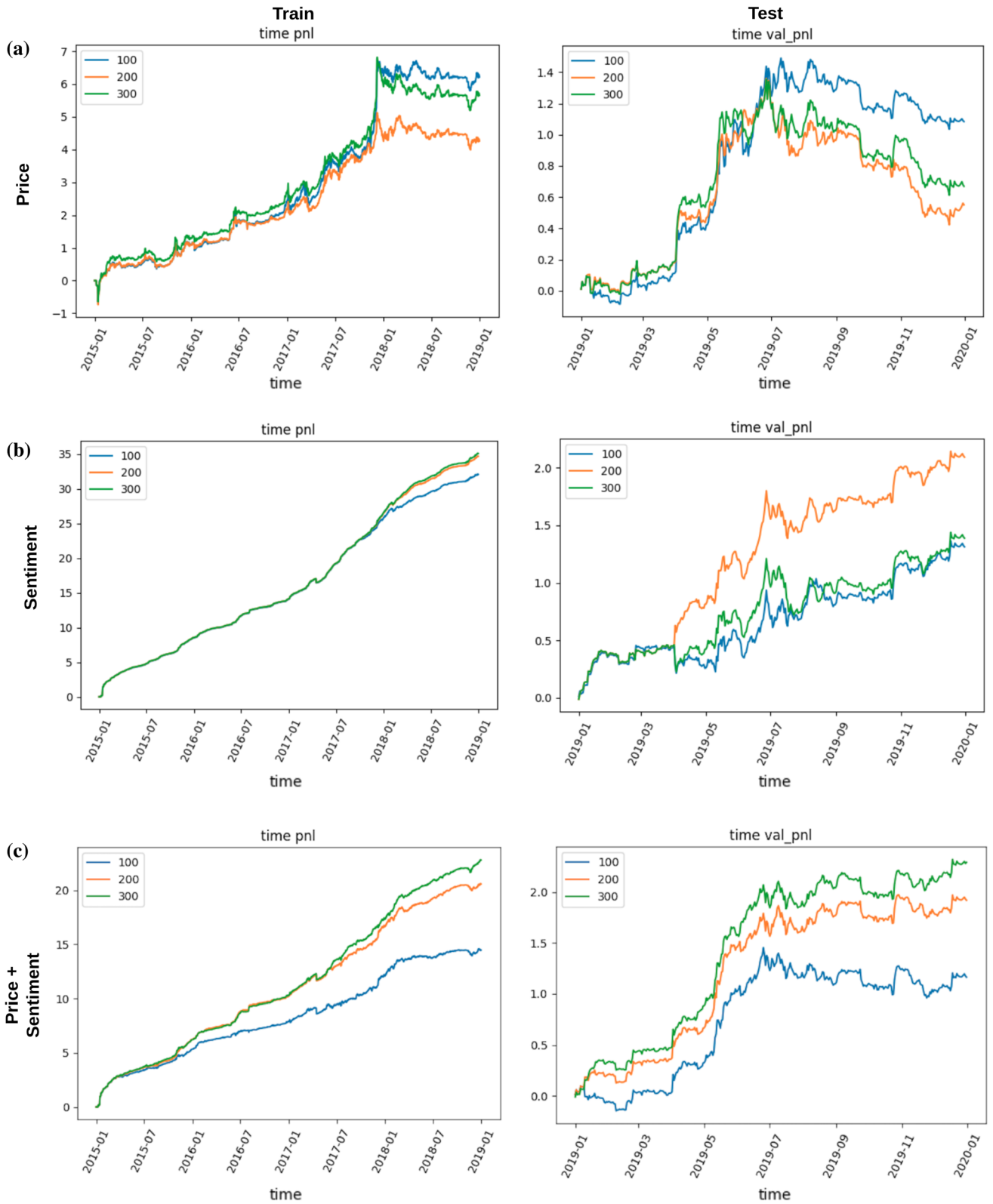

Fig. 5 Train (left column) and test (right column) PnL for MLP architectures trained on three different input sources: a) price alone, b) sentiment alone, and c) combined price and sentiment. Please note that the different lines correspond to different training epochs, i.e., blue refers to a model trained for 100 epochs, orange to a model trained for 200 epochs and green to a model trained for 300 epochs. Figure best viewed in color

Table 2 Evaluating the impact of unsupervised pre-training using financial documents (CryptoBERT) compared to regular training (BERT) and other large-scale models. Two setups are evaluated for BERT and CRYPTOBERT: (a) classification layer only training and (b) full training

Model	Accuracy	F1-score
roBERTa-base (TweetEval) [6]	0.55	0.55
XLM-roBERTa-base (multilingual) [7]	0.53	0.52
BERT (classification layer only)	0.60	0.60
CryptoBERT (classification layer only)	**0.68**	**0.69**
BERT (full training)	0.91	0.91
CryptoBERT (full training)	**0.92**	**0.92**

Bold values indicate best performance for each group

cases the multisource model achieves higher PnL than both the individual BERT and CryptoBERT models reported in Table 4. Again, we observe that the combination of two modalities leads to higher accuracy, yet lower PnL in a few cases. Based on these results, we expect that using more advanced approaches that can directly optimize the PnL to avoid this behavior, as discussed previously.

5 Conclusion

In this work, we conducted an extensive experimental study to evaluate the impact of using sentiment-enriched information for providing Bitcoin price movement indications. The conducted evaluation revealed that sentiment can indeed be a useful predictor, which can be also combined with other modalities, such as price information, to further improve the performance of the model. The experimental evaluation also highlighted that the model used to extract the sentiment can have a significant impact on the subsequent trading performance. Based on this observation, we proposed a simple, yet effective fusion strategy, which allows for fusing the sentiment information arising from different DL-based sentiment extractors. Indeed, this approach led to significant improvements. At the same time, we found out that forecasting accuracy is not necessarily fully correlated with trading performance, i.e., models can achieve lower forecasting accuracy, yet have higher PnL gains.

These observations raise a series of interesting future research questions. First, the proposed method can be also combined with reinforcement learning approaches, similar to [32], to directly optimize the models for the task at hand, instead of using a proxy task. Furthermore, we continue collecting data for various sources, aiming to compile a high-frequency dataset, allowing us to evaluate the impact of sentiment when fine-grained information is available, i.e., on a minute level. This approach also brings out several interesting questions on how this information should be presented to the models, e.g., using multiresolution bag-of-features representations could be employed to feed this information to the models in a more efficient manner [22]. Finally, more advanced ways of combining sentiment and price information, e.g., using transformer-based architectures for data fusion [11], could further improve the

Table 3 Qualitative evaluation between the roBERTa-base model and the proposed model. The winning class, along with confidence for each prediction are provided

Document	roBERTa-base	CryptoBERT
bitcoin (btc/usd) forecast and analysis on march 23, 2018	Negative: 0.93	Neutral: 0.56
'ripple and ethereum are horrible projects', says tone vays	Neutral: 0.87	Negative: 0.61
cruise stocks plummet as coronavirus hits global shores	Neutral: 0.87	Negative: 0.41
here's how you can profit from the online retrail megatrend	Neutral: 0.61	Positive: 0.43

Table 4 Average accuracy (%) and percentage (%) profit and loss (PnL) for the 50 top-performing configurations for each model (backtesting performed on the test set, i.e., 2019-2020). '(s)' denotes models trained only on sentiment sources, while '(s+p)' denotes models trained both on the price and sentiment modality. The prediction horizon was set to 1 day. The lot size used is constant for the whole duration of the backtest regardless of accumulated profits or losses

	BERT (Acc.)	BERT (PnL)	CryptoBERT (Acc.)	CryptoBERT (PnL)
MLP (s)	45.1% ± 2.7%	106.8% ± 22.8%	43.6% ± 4.4%	**186.6% ± 24.5%**
MLP (s + p)	**45.9% ± 3.4%**	**164.0% ± 20.6%**	**45.3% ± 3.3%**	161.2% ± 20.6%
CNN (s)	44.1% ± 4.6%	**200.7% ± 17.7%**	43.8% ± 4.6%	**197.2% ± 18.7%**
CNN (s + p)	**47.3% ± 2.9%**	177.0% ± 18.7%	**47.3% ± 2.8%**	173.7% ± 16.2%
LSTM (s)	44.5% ± 3.1%	127.5% ± 43.7%	43.4% ± 3.8%	169.1% ± 27.6%
LSTM (s + p)	**51.7% ± 2.6%**	**231.3% ± 22.1%**	**51.5% ± 2.8%**	**222.1% ± 18.9%**

Table 5 Average accuracy (%) and percentage (%) profit and loss (PnL) for the 50 top-performing configurations using both BERT and CryptoBERT sentiment sources for each model (backtesting performed on the test set, i.e., 2019-2020). '(s)' denotes models trained only on sentiment sources, while '(s+p)' denotes models trained both on the price and sentiment modality. The prediction horizon was set to 1 day. The lot size used is constant for the whole duration of the backtest regardless of accumulated profits or losses

	Test	PnL
MLP (s)	$43.9\% \pm 4.9\%$	$\mathbf{182.6\% \pm 24.5\%}$
MLP (s + p)	$\mathbf{46.6\% \pm 3.0\%}$	$176.2\% \pm 20.4\%$
CNN (s)	$44.2\% \pm 4.4\%$	$\mathbf{203.2\% \pm 23.6\%}$
CNN (s + p)	$\mathbf{47.1\% \pm 2.7\%}$	$181\% \pm 21.2\%$
LSTM (s)	$43.7\% \pm 3.9\%$	$173.4\% \pm 21\%$
LSTM (s + p)	$\mathbf{52.2\% \pm 3\%}$	$\mathbf{225.1\% \pm 22.4\%}$

obtained results, since, in some cases, combining these two sources of information only led to marginal improvements over just using sentiment information.

Acknowledgements This work has been co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH - CREATE - INNOVATE (project code: T2EDK-02094).

Declarations

Conflict of interest The authors declare no competing interests.

References

1. (2020) Analyzing crypto headlines—BDC consulting. https://bdcenter.digital/insights/cryptocurrency/analyzing-crypto-headlines
2. Almalis I (2021a) Financial news analysis with machine learning. Master's thesis, School of Informatics, Aristotle University of Thessaloniki
3. Almalis I (2021b) Ml_in_finance. https://github.com/ialmalis/ML_in_Finance
4. Araci D (2019) Finbert: Financial sentiment analysis with pre-trained language models. arXiv preprint arXiv:1908.10063
5. Bao W, Yue J, Rao Y (2017) A deep learning framework for financial time series using stacked autoencoders and long-short term memory. PLoS ONE 12(7):e0180-944
6. Barbieri F, Camacho-Collados J, Neves L, et al (2020) Tweeteval: Unified benchmark and comparative evaluation for tweet classification. arXiv preprint arXiv:2010.12421
7. Barbieri F, Anke LE, Camacho-Collados J (2021) Xlm-t: A multilingual language model toolkit for twitter. arXiv preprint arXiv:2104.12250
8. Chantona K, Purba R, Halim A (2020) News sentiment analysis in forex trading using r-cnn on deep recurrent q-network. In: Proceedings of the fifth international conference on informatics and computing, pp. 1–7
9. Day MY, Lee CC (2016) Deep learning for financial sentiment analysis on finance news providers. In: Proceedings of the IEEE/ACM international conference on advances in social networks analysis and mining, pp. 1127–1134
10. Deng Y, Bao F, Kong Y et al (2016) Deep direct reinforcement learning for financial signal representation and trading. IEEE Trans Neural Netw Learn Syst 28(3):653–664
11. Devlin J, Chang MW, Lee K, et al (2018) Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805
12. Dixon M, Klabjan D, Bang JH (2017) Classification-based financial markets prediction using deep neural networks. Algor Fin 6(3–4):67–77
13. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980
14. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444
15. Lei K, Zhang B, Li Y et al (2020) Time-driven feature-aware jointly deep reinforcement learning for financial signal representation and algorithmic trading. Expert Syst Appl 140(112):872
16. Liu Y, Ott M, Goyal N, et al (2019) Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692
17. Livieris IE, Iliadis L, Pintelas P (2021) On ensemble techniques of weight-constrained neural networks. Evol Syst 12(1):155–167
18. Mehtab S, Sen J (2020) Stock price prediction using convolutional neural networks on a multivariate timeseries. arXiv preprint arXiv:2001.09769
19. Mehtab S, Sen J, Dasgupta S (2020) Robust analysis of stock price time series using cnn and lstm-based deep learning models. In: Proceedings of the international conference on electronics, communication and aerospace technology, pp. 1481–1486
20. Oh SL, Hagiwara Y, Raghavendra U et al (2020) A deep learning approach for parkinson's disease diagnosis from EEG signals. Neural Comput Appl 32(15):927–933
21. Oyedotun OK, Khashman A (2017) Deep learning in vision-based static hand gesture recognition. Neural Comput Appl 28(12):3941–3951
22. Passalis N, Tefas A, Kanniainen J et al (2020) Temporal logistic neural bag-of-features for financial time series forecasting leveraging limit order book data. Pattern Recogn Lett 136:183–189
23. Passalis N, Seficha S, Tsantekidis A, et al (2021) Learning sentiment-aware trading strategies for bitcoin leveraging deep learning-based financial news analysis. In: Proceedings of the IFIP international conference on artificial intelligence applications and innovations, pp. 757–766
24. Pławiak P, Acharya UR (2020) Novel deep genetic ensemble of classifiers for arrhythmia detection using ECG signals. Neural Comput Appl 32(15):137–161
25. Schäfer R, Guhr T (2010) Local normalization: uncovering correlations in non-stationary financial time series. Phys A 389(18):3856–3865
26. Shi Y, Zheng Y, Guo K et al (2021) Stock movement prediction with sentiment analysis based on deep learning networks. Concurr Comput Pract Exp 33(6):e6076
27. Srivastava N, Hinton G, Krizhevsky A et al (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958
28. Tran DT, Iosifidis A, Kanniainen J et al (2018) Temporal attention-augmented bilinear network for financial time-series data analysis. IEEE Trans Neural Netw Learn Syst 30(5):1407–1418
29. Tsantekidis A, Passalis N, Tefas A, et al (2017a) Forecasting stock prices from the limit order book using convolutional neural networks. In: Proceedings of the IEEE conference on business informatics (CBI), pp. 7–12
30. Tsantekidis A, Passalis N, Tefas A, et al (2017b) Using deep learning to detect price change indications in financial markets.

In: Proceedings of the European signal processing conference, pp. 2511–2515
31. Tsantekidis A, Passalis N, Tefas A et al (2020) Using deep learning for price prediction by exploiting stationary limit order book features. Appl Soft Comput 93(106):401
32. Tsantekidis A, Passalis N, Toufa AS, et al (2020b) Price trailing for financial trading using deep reinforcement learning. In: IEEE Transactions on neural networks and learning systems
33. Tsantekidis A, Passalis N, Tefas A (2021) Diversity-driven knowledge distillation for financial trading using deep reinforcement learning. Neural Netw 140:193–202
34. Wei X, Chen W, Li X (2021) Exploring the financial indicators to improve the pattern recognition of economic data based on machine learning. Neural Comput Appl 33(2):723–737
35. Yu P, Yan X (2020) Stock price prediction based on deep neural networks. Neural Comput Appl 32(6):1609–1628
36. Zhang W, Skiena S (2010) Trading strategies to exploit blog and news sentiment. In: Proceedings of the international AAAI conference on web and social media
37. Zhang Z, Zohren S, Roberts S (2019) Deeplob: deep convolutional neural networks for limit order books. IEEE Trans Signal Process 67(11):3001–3012
38. Zimmerman S, Kruschwitz U, Fox C (2018) Improving hate speech detection with deep learning ensembles. In: Proceedings of the eleventh international conference on language resources and evaluation