BIROn - Birkbeck Institutional Research Online

Matuozzo, Alberto and Yoo, Paul and Provetti, Alessandro (2023) A right kind of wrong: European equity market forecasting with custom feature engineering and loss functions. Expert Systems with Applications , ISSN 0957-4174.

Downloaded from: https://eprints.bbk.ac.uk/id/eprint/50828/

Usage Guidelines:
Please refer to usage guidelines at https://eprints.bbk.ac.uk/policies.html or alternatively contact lib-eprints@bbk.ac.uk.

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Highlights

A right kind of wrong: European equity market forecasting with custom feature engineering and loss functions

Expert Systems With Applications xxx (xxxx) xxx

Alberto Matuozzo, Paul D. Yoo*, Alessandro Provetti*

- Time-series data extracted from European stock markets is analysed, with a specific emphasis on predicting future returns.
- An enriched feature set is constructed with variables from financial time series.
- A new type of loss function addresses the intricacies of framing equity forecasting.
- Convolutional NNs process the generated 2D feature maps as if they were images.
- Results are validated against simulations with baseline (pseudo)-random strategies.

Graphical abstract and Research highlights will be displayed in online search result lists, the online contents list and the online article, but will not appear in the article PDF file or print unless it is mentioned in the journal specific style requirement. They are displayed in the proof pdf for review purpose only.

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

A right kind of wrong: European equity market forecasting with custom feature engineering and loss functions

Alberto Matuozzo[1], Paul D. Yoo [*,1], Alessandro Provetti [*,1]

Department of Computer Science and Information Systems, Birkbeck College, University of London, London WC1E 7HX, UK

ARTICLE INFO

Keywords:
AI in finance
Stock trading
Neural networks
Time-series prediction
Deep learning

ABSTRACT

This study makes time-series-based predictions on future returns of the STOXX Europe 600 and the German DAX by adopting (in addition to a lagged and transformed version of the target series) a diversified set of predictors. Feature engineering expands further — from the initial raw group of variables, to extract knowledge of market conditions and demand for hedging. A penalisation factor is introduced with loss functions to learn a model from neural networks, in order to adapt a traditional machine learning regression framework to solve the equity forecasting problems in question. Architectures based on convolutional neural network are proposed, treating the obtained feature map similarly to an image. Experiments over different time periods demonstrate that trading strategies derived from the forecasts are more profitable than models based on efficient market assumptions. The temporal, non-stationary structure of financial data has a significant impact on the out of sample success of any model. It thus can be seen that different architectures exhibit different resilience to changing market conditions.

1. Introduction

Learning a robust equity market forecasting model is of the utmost importance for the decision making process of market participants, corporations and regulators as they put in place: financing (e.g., companies), investing (e.g., savers and investors) and prudential policies (e.g., central banks, exchanges commissions). Equity market forecasting problems are now at the intersection of finance, statistics and computer science. Finance has contributed to the field leveraging domain knowledge to devise informative features. Computer science researchers have focused on algorithm development. As shown by a recent scientometric literature review (Kehinde et al., 2023), the term Artificial Neural Network (ANN) has become the most searched machine learning method in stock market forecasting. In fact, ANN was is the third most frequently used term in equity-markets forecasting papers, just below "Market" and "Index".

Deep learning (Goodfellow et al., 2016) techniques in particular are capable of extracting an effective knowledge representation despite the low signal-to-noise ratio observed in financial data. Unlike other machine learning methods, Recurrent and Convolutional Neural Networks (which we will introduce next), are able to process data that have a temporal dimension. Our study takes a data-centric approach in expanding the features set and designing solutions that take into account the peculiarities of applying to finance some machine learning algorithms that were designed for other domains. One such subtlety, which is explored in this article, is that a return-forecasting model might produce, on average, accurate point estimations yet become a loss-maker if the predicted returns carries a different sign from the actual values. The main contributions of this study are as follows.

1. Expand the feature set in a novel way, deriving variables engineered from well-referenced and publicly-available financial time series.
2. Propose a class of custom loss functions to address the subtleties of framing equity forecasting problems. We attempt returns prediction, penalising forecasts in the wrong direction.
3. Explore the adoption of architectures based on convolutions over 2 dimensions to leverage a heterogeneous set of features. As a result, the feature map is treated similarly to an image, transposing the deep learning success in computer vision, to time series.
4. Validate our new models over different periods for a geographic area (Europe) which we felt was under-researched so far. The experimental results show that our proposed concepts beat buy and hold, random and pseudo-random models in terms of gross profitability.

* Corresponding author.
E-mail addresses: amatuo01@mail.bbk.ac.uk (A. Matuozzo), p.yoo@bbk.ac.uk (P.D. Yoo), a.provetti@bbk.ac.uk (A. Provetti).

[1] Equal contribution.

https://doi.org/10.1016/j.eswa.2023.119854
Received 22 October 2022; Received in revised form 17 February 2023; Accepted 8 March 2023

2. Related works

A financial market is considered efficient, under the Efficient Market Hypothesis (EMH) if prices fully and instantaneously reflect all available public and private information (Fama, 1970). In an efficient market, the current price of a security would theoretically provide all the information required to raise or allocate capital of a firm.

Information can be considered valuable if market participants can act on it and generate profits after considering transaction costs. If markets were regarded efficient under EMH, then their mere existence would be an anomaly: only irrational agents would participate in a game with no information advantage. Grossman and Stiglitz (1982) overcome this paradox allowing prices to only partially reflect information, thus, that investors, who dedicate resources to research investment ideas, are rewarded.

By analysing different investment strategies and interviewing some of the best investors, Pedersen (2019) concludes that financial markets are efficiently inefficient: they are inefficient to the extent that a few participants will be able to uncover anomalies profitably, and in doing so cover their costs, and efficient enough not to encourage more players entering the arena.

The relationship between features adopted to solve financial forecasting problems is of such complexity that is difficult to be inferred by a domain researcher alone. Using simple models with predictors exclusively built by a finance expert, could lead to false discoveries or miss important phenomena, ultimately generating inaccurate forecasts.

Machine learning (ML) algorithms have been shown to outperform traditional statistical methods (e.g., ARIMA and OLS) in forecasting market trends (Qian & Gao, 2017) and predicting excess returns (Gu et al., 2020) in US equity markets. It is thus reasonable to assume that the better results of ML algorithms stem from their ability to extract patterns amongst a significant and heterogeneous number of variables, even in presence of non-linearities.

It is possible to consider Recurrent Neural Network (RNN) as a non-linear version of auto regressive models (Dixon et al., 2020). Even though this architecture can extract information from temporal structure of data, it is not effective when dealing with long sequences: RNNs suffer from the so-called exploding or vanishing gradient problem. The other model of reference, the Long Short-term Memory (LSTM) cell, Hochreiter and Schmidhuber (1997) significantly mitigates these issues incorporating, in addition to a short-term hidden state, a long-term cell state.

Nelson et al. (2017), in a rare simulation on emerging market equities (Brazil), found that LSTM networks give more accurate predictions than Random Forest (RF) and ANN. In a comparative study on stock selection amongst members of the S&P 500 (Fischer & Krauss, 2018), LSTM outperformed RF, ANN and Logistic regression. In another study on S&P 500 components return prediction, Guida (2018) LSTMs were compared to ANNs and Support Vector Regressors (SVRs). The main finding was that increasing the sample window length up to 10 time steps improved the LSTM performance, while hurting SVRs.

The next relevant concept is the Convolutional layer. It can be briefly described as the sliding of a filter over the input. Thanks to the sliding filter, CNNs are able to learn local features and recognise them within a feature map.

Convolutional layers over 1 dimension (C1D) have been deployed in time-series forecasting, sliding filters across the time dimension. In a multivariate setting, each predictor can be treated as a channel of an image, with C1Ds deployed independently for each variable. For instance, Eapen et al. (2019) deployed C1Ds to extract features subsequently fed to LSTM layers to forecast the 1 week ahead level of the S&P 500. C1D-centred networks have been used to obtain a forecast directly, For instance, Borovykh et al. (2018) adapted the Wavenet architecture (van den Oord et al., 2016) to forecast the 1-day forward return for the S&P 500 conditioned on the volatility index VIX and the CBOE 10-year rates.

Convolutions over 2 dimensions (C2D), albeit less frequent in the time series forecasting literature, offer a clear advantage in a multivariate setting, i.e., to represent knowledge residing in the relationship between features and recognise patterns as they manifests in different points (in time) of the feature map (Goodfellow et al., 2016).

A recent promising approach, leveraging the success of deep architectures in image classification and video prediction, has been to lay out securities prices at a given point in time in a 2D map. Cohen et al. (2020) point out that professional traders derive technical signals, not from analysing time series of securities prices in numeric form, but from their visual representation (e.g. Japanese candlesticks, as in Fig. 1). The authors successfully converted a time-series classification problem while extracting technical signals, into an image classification problem.

Zeng et al. (2021) adopted the above approach and took it one step further: the price of multiple securities at a given point in time is laid out in a 2D grid as if it were an image. Each asset is a tile (pixel) in the feature map. As a result, multivariate time series are now represented as a succession of images. C2Ds are leveraged as they are able to extract knowledge from the time-scoped relationship between securities. Time series forecasting thus becomes a *video prediction* task, to which the proven neural network architectures for video prediction can be applied to.

Another framework, which Gudelek et al. (2017) and Sezer and Ozbayoglu (2018) adopted in trading simulations based on Exchange Traded Funds (ETFs) forecasts, is to present the data of a windowed dataset (a 2D grid: time steps, features) to a C2D network. Thus, each sample is treated as if it was an image over 1 channel. We see again the ability of C2D to extract knowledge on the time-varying relationships between features.

3. Datasets

Traders use an ensemble of economic, sentiment and technical indicators when assessing the opportunity set, making a forecast, and deciding on a course of action. To match this standard setup we built a comprehensive data pool, encompassing, besides the target index time series and widely-adopted technical indicators, *covariates* which will act as proxies for demand for hedging, interest rates and inflation. As we focus on European equity indices, we notice how those appear to be under-researched with respect to the corresponding US indices. The dataset comprises daily closing prices for the 2010–2020 period for STOXX Europe 600 (SXXP), the DAX and a set of predictors:

1. the future on Bunds (RX1), as a proxy for European interest rates.
2. the Wisdomtree Physical Gold Exchange Traded Commodity Fund (PHAU), as a proxy for demand to store wealth.
3. two implied-volatility indices: Euro STOXX 50 Volatility Index (VSTOXX, ticker V2X) and its German counterpart, VDAX (ticker V1X).

In addition, in order to capture information regarding trend and levels of exhaustion, the daily readings for the following *technical indicators* (Murphy, 1986) have been obtained:

4. Bollinger bands %B: the distance from the upper band (0 indicates the price is on the lower band);
5. Commodity Channel Index (CMCI): an oscillator indicating oversold conditions at −100 and overbought at 100;
6. Stochastic %K: oscillator oversold below 30 and overbought above 70, and
7. William's %R: oscillator flashing oversold for values below −80 and overbought for values above −20.
8. Relative Strength Index (RSI): oscillator indicating an oversold set up below 30 and overbought above 70;

Fig. 1. Example of visual cues from displaying time series: a SXXP 10-days hourly Japanese candlestick chart. *Source:* Bloomberg.

The dataset can be built by using a professional grade software (e.g Bloomberg, Refinitiv Eikon). Alternatively, data is publicly available on the internet at Yahoo finance.[2] or at investing.com[3]

Furthermore, technical indicators can be easily computed with the open source Python library TA-Lib.[4]

3.1. Feature engineering

Missing data, albeit rare, have been dealt with the forward fill method; they were mostly due to holidays affecting different securities' pricing. Time series of log returns are computed for the target series (SXXP and DAX), RX1 and PHAU. In fact, it is known that equity markets exhibit momentum both at single-stock level (Jegadeesh & Titman, 1993) and at index level (Moskowitz et al., 2012). The sum of the target index returns over 3 month (66 trading days) and 1 year (252 trading days) rolling are computed to measure short-term and long-term momentum, respectively.

A novel set of diverse features are engineered from the available data with the aim to extract (i) information content and hidden patterns related to the persistence of trends and (ii) changes in sentiment, thus capturing the *market regime* in which returns are generated.

From the target series, the direction of returns is extracted, in order to compute Shannon's entropy on a 3-months (66 trading days) rolling basis. Entropy has been used as an alternative measure of volatility in risk management (Sheraz et al., 2015) with the aim of capturing trend persistence; according to the information we have, this is the first study utilising such feature engineering approach in this domain.

As financial series are very noisy the entropy is almost always close to 1. Nevertheless, it can be observed in Fig. 2 how, during major market shifts, direction becomes more persistent, and hence the entropy indicator goes down.

Next, the 6-months (126 trading days) rolling skewness of returns is computed to capture change in the symmetry of the distribution. Financial markets undergo regime shifts (Bernstein, 1999); using stock returns time series as a predictor does not take into account the specific market phase in which samples are observed. To mitigate this, the target return series is here divided by its rolling 6-months (126 trading days) standard deviation, obtaining 'vol-scaled' returns. As a result,

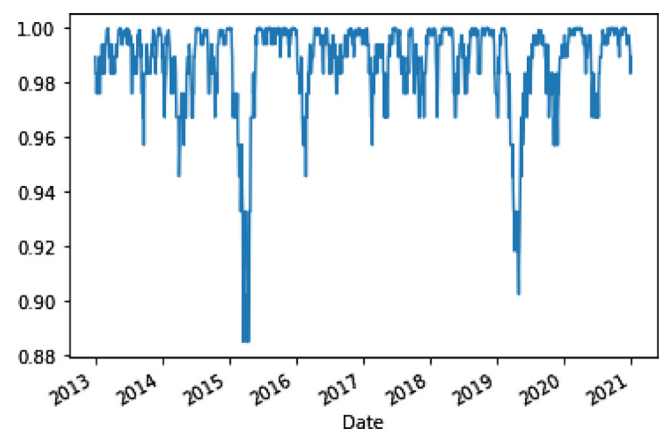

Fig. 2. SXXP Entropy. Trend persistence can be observed, e.g., in 2015, ahead of the start of the Quantitative Easing programme by the ECB.

the time series embeds not only the magnitude of returns but also the market environment in which those returns are generated. This scaling technique is dynamic, hence, reflecting the constantly changing nature of financial markets.

Implied Volatility Indices, viz. the VIX, are utilised directly in forecasting models (Borovykh et al., 2018) to capture the demand for hedging, highlighting, from a sentiment point of view, the state of fear in the market. Our novel approach focuses on the idea that the absolute level of the index misses the important element of change, that in fact embodies a shift in market regime: a VIX level of 30 per se is less informative than knowing that it spiked from 15 or that it fell from 45. In this study, from the raw time series of VSTOXX and VDAX, we compute the logarithm of ratio of implied volatility indices at a given time, over their value observed the previous day (i.e. log return) to capture the change in demand for hedging or, from a sentiment perspective, the change in fear. Furthermore, the related standard deviation (here, 'vol of vol') is computed on a 3-month (66 trading days) rolling basis to capture the persistence of this emotional state.

At the end of the feature engineering phase we obtained the following predictors:

[2] Please see https://uk.yahoo.finance.com
[3] Please see https://uk.investing.com
[4] Please see https://mrjbq7.github.io/ta-lib/

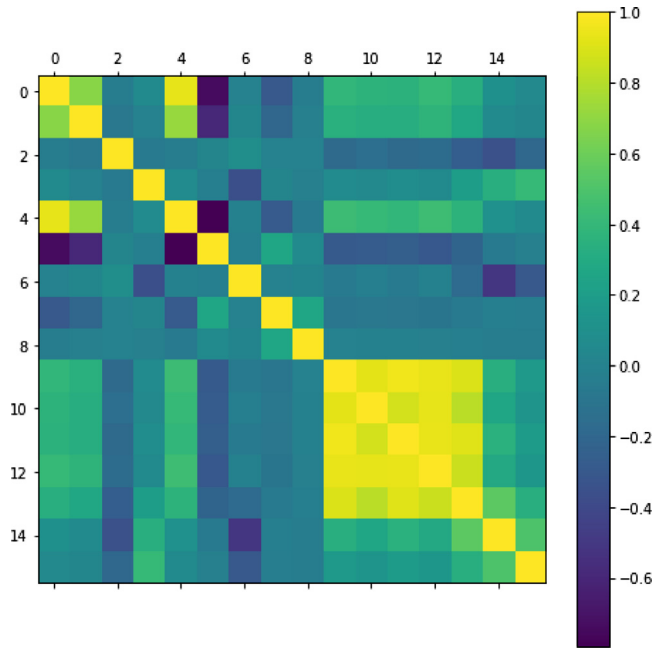

Fig. 3. SXXP Features correlation heatmap. Technical indicators appear to be strongly correlated with each other.

1. Entropy;
2. Skewness;
3. *Vol-scaled* returns;
4. Implied volatility index change;
5. *Vol-of-vol;*
6. RX1 returns;
7. PHAU returns;
8. the five technical indicators (in the same order as in Section 3);
9. 3-months momentum and
10. 1-year momentum.

These 14 features are now re-arranged in the four categories of statistical, macro, sentiment, technical and momentum. Thus, we will have: Entropy, Skewness, Vol-scaled returns, Implied-volatility index change (e.g. V2X change), Vol-of-vol, RX1 returns, PHAU returns, Bollinger %B, CMCI, Stochastic %K, RSI, 3mth- and 1yr-momentum. In comparison, Sezer and Ozbayoglu (2018) adopted 15 features. Yet our selection is more eclectic and goes beyond technical indicators to include macro and sentiment (volatility) gauges.

3.2. Exploratory descriptive analysis

The heatmap in Fig. 3 shows the respective degrees of correlation found in the final dataset; the target SXXP returns and direction are in column 0 and 1, followed by the 14 features described above. The heatmap shows how technical indicators appear to be significantly correlated to each other. Similar considerations could be drawn for the DAX.

The datasets are described in Tables 1, 2, 3, and 4 below. For clarity content has been split in 2 subsections. Notice how Dax experienced a positive $+4\sigma$ day on April 24th 2017 in the aftermath of the first round of the French presidential elections. Both SXXP and DAX experienced a -7σ on March 12th 2020, and a $+4\sigma$ day shortly afterwards on March 24th 2020.

4. Framing the prediction problem

The stock market predictive modelling problems can be framed either as a classification or regression task. In the former the aim is to forecast market direction, in the latter the aim is to derive a point estimate of a future return.

Predictive accuracy in this domain does not necessarily translate into trading profitability. Adopting the classification framework has the risk that the efforts of identifying the direction of the market could be thwarted by instances where the market moves significantly and we mistake its direction. On the other hand, deriving point estimation might lead to the wrong trading decision (i.e., losses) when the predicted return is in fact close to the actual value, albeit in the opposite direction. An inaccurate forecaster might lead to profitable trading if the predictions, although distant from the observed value, happen to fall in the right direction. An attempt to solve this conundrum, offered in both industry and academic research (Borovykh et al., 2018; Guida, 2018) is to use the regression framework to develop a model and then extrapolate the direction, evaluating the solutions according to the hit ratio (i.e., accuracy).

In this study, we adopt the regression framework with the aim of forecasting the one day forward return of the SXXP and the DAX. The forecast is used to simulate trading strategies. So, again under the sole assumption that market are *efficiently inefficient* (Grossman & Stiglitz, 1982; Pedersen, 2019) we aim to assess whether there is "skill" in a forecasting model, where skill is understood in comparison to, picking the direction to trade at random or even according to the frequency of positive/negative days in the training set, called pseudo-random. Once the model predicts a point estimate, direction and conviction are extracted. The direction is used to establish a long or short position on a daily basis. The conviction indicator c (between +1 and −1) is obtained by passing the predicted return as argument of the hyperbolic tangent function. The conviction value is a further input to simulate a trading strategy with variable position sizing where, instead of deploying +1/−1 risk unit of capital according to the extrapolated direction, it would put at risk 1+ c / −1−c units.

4.1. New loss functions

In order to mitigate the aforementioned issue of making a point forecast that, despite being close to the actual value, falls in a wrong direction and thus prompts the wrong course of action, we propose two ideas to insert a small penalisation factor $p > 1$ to the standard regression loss functions. One option prescribes including the penalisation as a multiplicative term (to be tuned by the researcher) to increase the error when the sign of the point forecast is different from the actual return. In another variant, we focus on penalising exclusively the errors occurring in a wrong direction and deemed large enough to represent a source of significant losses. The threshold value to apply the penalisation in this case is set to one: the data is standardised before training, and hence pertain to errors larger than one standard deviation. The formulas below apply the concept to absolute error, thus, deriving the loss functions defined respectively as Directional Absolute Error (DAE) and Directional Big Error (DBE). The former applies the penalisation to every forecast occurring in a wrong direction, while the latter applies only to the larger examples.

$$dae(y, \hat{y}, p) = \begin{cases} |y - \hat{y}| \cdot p & \text{if } y \cdot \hat{y} < 0 \\ |y - \hat{y}| & \text{otherwise.} \end{cases} \tag{1}$$

$$dbe(y, \hat{y}, p) = \begin{cases} |y - \hat{y}| \cdot p & \text{if } y \cdot \hat{y} < 0 \text{ and } |y - \hat{y}| > 1 \\ |y - \hat{y}| & \text{otherwise.} \end{cases} \tag{2}$$

5. Architectures

Our experiments were conducted by training 5 different neural networks architectures, all based on convolution layers. In literature convolutions over 1-dimension (C1D) architectures are conceived to

Table 1
Summary of the values in our SXXP dataset, features 1 to 7.

	Return	Entropy	Skewness	VScaledRets	V2X_chg	Vol_of_vol	RX1_rets	PHAU_rets
mean	0.01%	0.99	−0.30	0.01	0.00	0.07	0.01%	0.01%
std.	1.08%	0.02	0.54	1.05	0.07	0.02	0.37%	0.92%
min	−12.19%	0.88	−4.39	−7.54	−0.43	0.04	−2.71%	−8.34%
max	8.07%	1.00	0.63	4.16	0.47	0.13	1.96%	6.47%

Table 2
Summary of the values in our SXXP dataset, features 8 to 14.

	Bollinger %B	CMCI	Stochastic %K	Willaims %R	RSI	3M_mom	1Yr_mom
mean	55.13%	16.20	60.34	−40.58	52.88	0.88%	3.46%
std.	32.91%	106.76	31.14	31.68	11.51	7.74%	11.07%
min	−38.81%	−304.84	0.00	−100.00	10.82	−39.18%	−29.55%
max	130.37%	291.42	100.00	0.00	81.57	26.00%	27.09%

Table 3
Summary of the values in our DAX dataset, features 1 to 7.

	Return	Entropy	Skewness	VScaledRets	V1X_chg	Vol_of_vol	RX1_rets	PHAU_rets
mean	0.03%	0.986	−0.236	0.015	0.000	0.062	0.01%	0.01%
std.	1.31%	0.017	0.496	1.043	0.063	0.014	0.37%	0.93%
min	−13.05%	0.902	−4.324	−7.543	−0.370	0.036	−2.71%	−8.34%
max	10.41%	1.000	1.133	4.866	0.411	0.115	1.96%	6.47%

Table 4
Summary of the values in our DAX dataset, features 8 to 14.

	Bollinger %B	CMCI	Stochastic %K	Willaims %R	RSI	3M_mom %	1Yr_mom %
mean	56%	15.77	59.77	−41.31	53.32	0.02	0.07
std.	33%	107.31	30.86	31.28	12.10	0.10	0.14
min	−35%	−282.05	0.00	−100.00	10.23	−0.44	−0.32
max	135%	321.57	100.00	0.00	84.64	0.36	0.36

take advantage of the capability of extracting features detecting patterns across time, independently over each predictor, as if they were different channels of an image.

Convolutions over 2-dimensions (C2D), albeit less frequent in the literature, offer the advantage, in a time series multivariate setting, to represent knowledge residing in the relationship between features over time (Gudelek et al., 2017; Sezer & Ozbayoglu, 2018). This is the approach followed here: each sample is treated as if it was an image over one channel. C2D are deployed to derive a knowledge representation of both features and their interaction.

In order to mitigate the issue of correlation between feature maps (caused by adjacent 'pixels' being highly correlated), it is particularly likely to occur when sliding kernels across technical indicators, the spatial dropout (Tompson et al., 2015) is adopted. Compared to traditional dropout (Srivastava et al., 2014), spatial dropout cancels at random entire filters, instead of individual activations, in doing so spurring independence between feature maps. The summary of the resulting models are as follows:

1. C1D_E: 2 blocks composed by a C1D layer (32 and 64 filters respectively) with kernel size equal to 3, ELU activation and He normal initialisation; Max Pooling and Spatial Dropout(0.4). The output is connected to a dense layer of 128 units with dropout (0.5), fully connected to one neuron with linear activation to generate the forecast.
2. C1D_LSTM: inspired by Eapen et al. (2019), it consists of 2 Convolution blocks as above (albeit with ReLU activation), followed by a Bidirectional LSTM layer with 200 units and dropout (0.5) connected to one neuron to output the prediction.
3. C2D1: Convolutional 2D layer composed by 64 (3 × 3) filters, ReLU activation, followed by Spatial Dropout, Max Pooling and Batch Normalisation. The output is connected to a dense layer of 50 units and dropout (0.2). The final layer, with linear activation, outputs the forecast.
4. C2D2: this architecture increases the expressive power of convolutional neural networks while leveraging the computational efficiencies obtained by stacking layers, inspired by VGGNet (Simonyan & Zisserman, 2015). As a result, it is composed by 2 blocks, each composed by: 2 C2D layers (32 filters on the first block, 64 on the second) with ReLU activation, followed by Max Pooling, Spatial Dropout (0.4) and Batch Normalisation. The fully-connected part of the network encompasses a dense layer of 128 units, followed by dropout (0.5), and the final unit for the prediction.
5. C2D_FUN: this dual head network configuration aims to mimic the fact that a human trader will inspect the same time series (often via visual representation) and look for patterns occurring over a varying number of observations (Fig. 4). As a result, 2 branches host a C2D-centered bloc (similar to C2D1) of different kernel size (3 and 5 respectively) in order to process the same inputs independently and produce a vector output via a 100 units dense layer. These are then concatenated, and, after dropout (0.5), connected to one unit to make a forecast.

Our architecture also reflects the idea that trading simulations layers should be deployed with incremental complexity; hence, we begin with a straightforward C1D model and finish with the dual-head C2D.

6. Experiments

Our experimental setup consists of the now-standard pairing of Tensorflow (Abadi et al., 2016) and Keras.[5] Experiments are run performing a time-series walk-forward validation procedure which encompasses 1250 samples for training and 250 for testing over 3 different

[5] Please see https://keras.io/

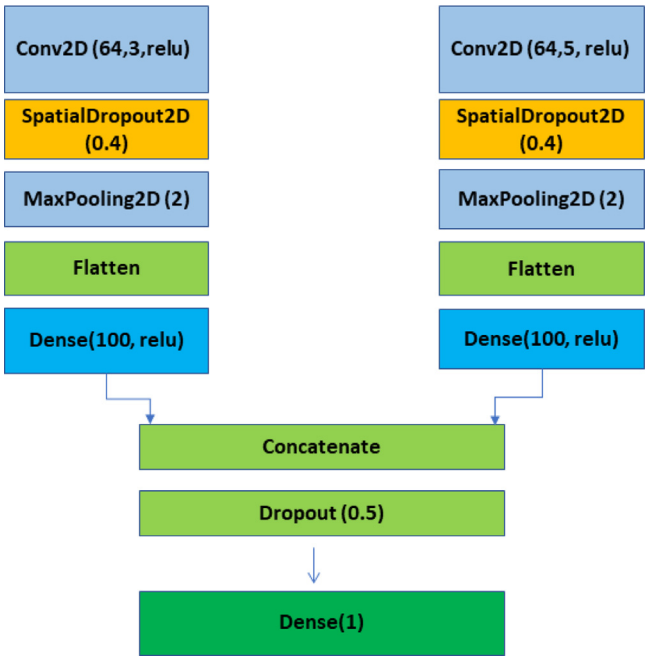

Fig. 4. C2D_FUN architecture: explore the same time series independently with filters of different size.

Table 5
Predicting SXXP 2018 returns: DAE and DBE loss respectively.

Model	MAE	Accur.	P&L	Rank	L_P&L
C1D_E	0.62%	54.00%	4.03%	3	5.14%
C1D_LSTM	0.63%	53.63%	2.12%	4	4.82%
C2D1	0.63%	55.09%	13.60%	2	19.49%
C2D2	0.63%	52.56%	−1.94%	5	−0.49%
C2D_FUN	0.63%	55.09%	17.19%	1	23.81%
Model	MAE	Accur.	P&L	Rank	L_P&L
C1D_E	0.62%	53.87%	3.94%	4	5.01%
C1D_LSTM	0.63%	54.15%	5.28%	3	7.77%
C2D1	0.63%	54.88%	12.00%	2	16.23%
C2D2	0.63%	53.07%	−0.66%	5	0.81%
C2D_FUN	0.63%	54.55%	16.01%	1	22.13%

Table 6
Predicting SXXP 2019 returns: DAE and DBE loss respectively.

Model	MAE	Accur.	P&L	Rank	L_P&L
C1D_E	0.50%	55.57%	10.91%	3	12.16%
C1D_LSTM	0.50%	54.24%	7.82%	4	9.16%
C2D1	0.51%	55.07%	12.02%	1	13.00%
C2D2	0.51%	54.56%	11.51%	2	11.95%
C2D_FUN	0.51%	51.49%	5.60%	5	7.14%
Model	MAE	Accur.	P&L	Rank	L_P&L
C1D_E	0.50%	55.33%	10.59%	3	12.13%
C1D_LSTM	0.50%	55.96%	14.77%	1	16.85%
C2D1	0.51%	53.29%	10.52%	4	12.13%
C2D2	0.51%	55.83%	14.05%	2	15.92%
C2D_FUN	0.51%	52.27%	9.21%	5	11.59%

periods (each fold is shifted forward by 250 trading days) which correspond approximately to 2018, 2019 and 2020.[6]

Next, response and predictors are standardised. The data is windowed with a look back periods of 10 days (hence 10 x 14 images), leaving a gap as wide as the number of time steps between each training and test set (de Prado, 2018). The window length was configured to strike a balance taking into account the number of features, the number of samples and the noisy nature of stock market data. The choice of 10 days was informed by preliminary experiments showing that longer time windows of 20 and 50 days lead to deterioration of performance. It is thus reasonable to infer that lengthening the time window could make it harder for models to detect signal from noise.

Training is conducted using the well-known Adam optimiser. Learning rate are tuned for C2D based architectures choosing between 10^{-3} (its default value) and 10^{-4} to mitigate bias.

Simulations are run using both DAE and DBE loss functions. The penalisation parameter are tuned on the training set (curtailing 20% of training samples as validation set), in most cases we obtain p=1.1. Models are fit with batch size of 10 and early stopping. The performance of the proposed models has been valued on both predictive and trading skills. Therefore, the results are presented adopting the following metrics:

1. Mean Absolute Error (MAE) of the predicted return.
2. Accuracy (i.e hit ratio) of the direction derived from the sign of the predicted return.
3. Gross profitability of the simple and conviction weighted trading strategies as discussed in Section 4 (P&L and L_P&L respectively).
4. Rank according to profitability in each period (lower is better): this will highlight how different architectures perform differently over different time periods. Different models exhibit different resilience to changing market environment.

[6] For SXXP the first training period encompasses samples from March 1st 2013 to January 15th 2018; the related test set (P1) spans from January 30th 2018 to January 21st 2019. The subsequent fold, following the same logic, starts on February 21st 2014.

Given the stochastic nature of neural networks, each experiment is repeated 30 times. We report the average performance; this also mimics the results obtained adopting a network ensemble.

Of course, any given architecture can perform differently over different periods. In general equity markets are non-stationary so the choice of deploying one single forecasting model would be very sensitive: if we look at performance on the most recent historical data, a single model might well generate costly losses as market conditions change. In this spirit we analyse the experimental results in the next section.

6.1. Discussion

The experimental results on the SXXP are shown in Tables 5, 6 and 7 for 2018, 2019 and 2020 respectively. We found that, irrespective of the loss function, differences in forecast accuracy do not translate directly into differences in profitability.

Different algorithms are able to extract knowledge on the direction and the size of the return differently: for example, in 2020, with DBE training (as in Table 7), the C1D_LSTM architecture, despite producing on average more accurate predictions than C1D_E, generates less profits. Irrespective of the period, models are able to combine successfully the point estimation with the direction, particularly when training with DBE: conviction enhances the profitability (L_P&L is bigger than P&L).

Looking at different testing periods, it can be seen that 2019 (as in Table 6) was the easiest to predict (lowest MAE), while 2020 was the hardest. Having said that, the higher volatility environment of 2020 is captured profitably: every model outperformed the SXXP (down 4.63%) and every network trained with DBE was profitable.

Comparing DAE and DBE training, it can be seen that even though MAE and accuracy are similar, adopting the latter loss function is on average more effective in generating profitable signals, particularly in the third period (as shown in Table 7), the year of COVID-19 outbreak; above all comparing conviction levered strategies. This is particularly noticeable considering the C1D_E architecture: similar MAE and accuracy performance across loss functions, translate into significantly

Fig. 5. SXXP realised volatility.
Source: Bloomberg.

Table 7
Predicting SXXP 2020 returns: DAE and DBE loss respectively.

Model	MAE	Accur.	P&L	Rank	L_P&L
C1D_E	1.16%	51.88%	2.53%	4	2.93%
C1D_LSTM	1.16%	52.17%	−0.01%	5	0.12%
C2D1	1.16%	52.37%	13.71%	2	15.16%
C2D2	1.16%	53.11%	8.38%	3	9.69%
C2D_FUN	1.15%	54.71%	19.46%	1	30.91%
Model	MAE	Accur.	P&L	Rank	L_P&L
C1D_E	1.16%	52.07%	12.38%	3	14.48%
C1D_LSTM	1.16%	53.52%	4.42%	5	4.54%
C2D1	1.16%	53.33%	13.10%	2	21.59%
C2D2	1.16%	52.76%	7.54%	4	8.71%
C2D_FUN	1.14%	54.67%	17.79%	1	28.81%

Table 8
SXXP, difference in average returns between training regimes by period: DBE–DAE.

Period	P&L_Δ	L_P&L_Δ
P1	0.31%	−0.17%
P2	2.25%	3.04%
P3	2.23%	3.86%

larger profits adopting DBE. Looking at the results obtained deploying the C2D1 model in the same period, a divergence in profitability can be observed once position sizing takes into account conviction.

Let us delve further into the divergence in average profitability between different loss functions per period (shown in Table 8). We observe that in P1 the difference between training regimes is small, and actually to the advantage of DAE if considering the trading strategy with variable position sizing. The divergence is in favour of DBE in the subsequent periods, with the maximum realised by conviction-based trading in P3.

As desired, DBE training penalises models for large errors in the wrong direction; thus, our hypothesis in terms of financial logic, is that DBE improves on DAE when the market undergoes major gyrations, and several large peak to trough moves occurs within a given period. Further, DBE can be observed to reflect the realised volatility (shown in Fig. 5), particularly when computed over 10 days.

Moreover, we can see in Fig. 6 how the ranking in terms of P&L changes abruptly from one period to the next.

Different periods exhibit different market conditions; there is not a winning algorithmic choice a priori. Hence, our rationale for running simulations in 3 periods with very different regimes: for example, the SXXP was down 10.86% in P1, up 17.4% in P2, and in P3 we have the Covid induced shock. Empirical research is warranted in order to identify the most appropriate architecture. While we are seeing that adopting C2D is a valid method, we notice however that this is not always a successful choice: the C2D2 architecture is loss making in P1 and highly profitable in P2.

This exposes the danger of choosing a model validated on a given time period, to deploy it immediately afterwards when market conditions change. Choosing the model to send in production looking at the performance achieved in the first testing period for deployment in the second, will generate sub optimal results: the C2D_FUN architecture, best performer in 2018 (rank 1), generates the lowest profits the following period (rank 5).

Fig. 7 shows the cumulated profits over the test sets, assuming that at the end of every period, the capital allocation is reset: this is to mimic the dynamics of trading desks that see at the beginning of every year their P&L and available capital reset. Irrespective of the loss function, over the 3 years period considered, trading strategies built on the forecasting models devised here, are more profitable than buy and hold, random, and pseudo-random counterparts. The best performing model in terms of cumulated profitability is C2D_FUN trained with DBE. Moreover, training models with DBE, translate into larger cumulated profits for 4 out of 5 architectures considered. This turns out to be particularly advantageous for networks centred on C1D layers.

For the DAX we have similar findings, shown in Tables 9–11. Thanks to extensive experiments we can conclude that our models acquire a knowledge representation of the problem incorporating both direction and amount of the move.

Conviction-levered strategies outperform direction based strategies with static position size; this is irrespective of the algorithm or the testing period. Adopting a dynamic position size depending on conviction not only enhances profits, but, mitigates losses, as can be seen in Table 9.

A further emerging observation is that year 2019 was the easiest period to predict (as in Table 10), while 2020 was the most difficult of the three (Table 11). Even though predicting the size of movements is very hard in 2020 given the major moves between February and June as the global economic arena was coming to terms with the COVID-19 pandemic, models successfully manage to navigate the turbulent environment.

In conclusion we are satisfied that every architecture records a positive performance and outperforms the baselines.

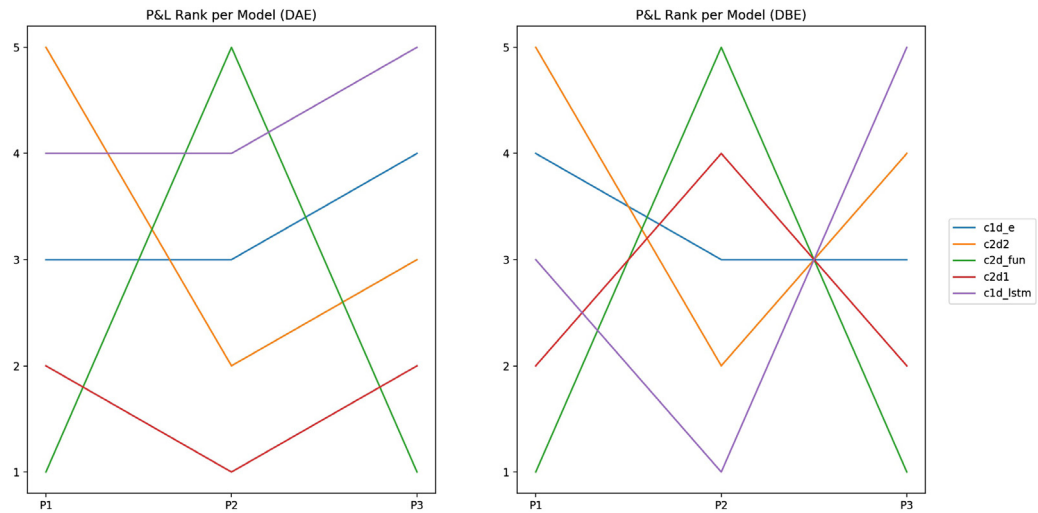

Fig. 6. SXXP, DAE and DBE training: model profitability rank by period (lower is better).

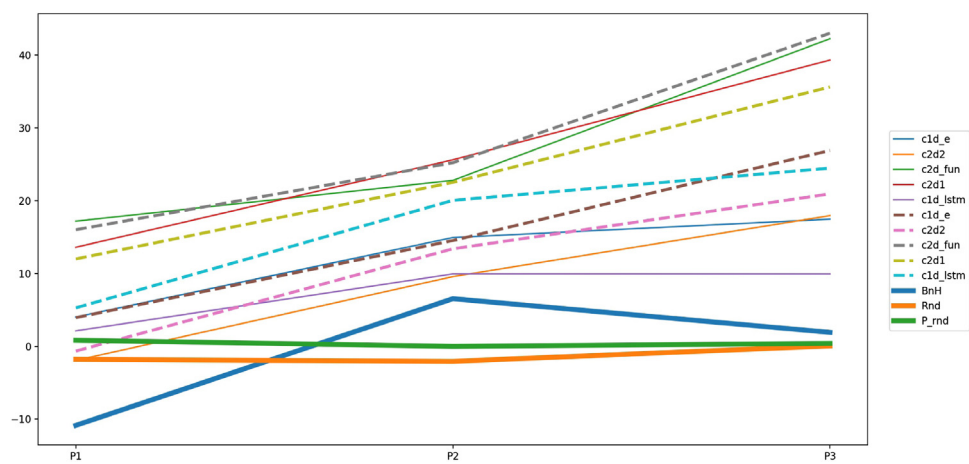

Fig. 7. SXXP cumulated gross profitability in % terms over the testing periods. Models beat random strategies (thick lines). DBE-trained arch. (intermittent lines) outperforms DAEs.

Table 9
Predicting DAX 2018 returns: DAE and DBE loss respectively.

Model	MAE	Accuracy	P&L	Rank	L_P&L
C1D_E	0.80%	49.57%	−10.17%	4	−10.05%
C1D_LSTM	0.82%	49.65%	−8.64%	3	−6.83%
C2D1	0.83%	51.68%	6.18%	2	10.97%
C2D2	0.83%	48.45%	−14.81%	5	−14.50%
C2D_FUN	0.80%	52.80%	11.06%	1	17.44%
Model	MAE	Accuracy	P&L	Rank	L_P&L
C1D_E	0.81%	49.65%	−10.02%	5	−9.72%
C1D_LSTM	0.81%	49.99%	−7.33%	3	−5.33%
C2D1	0.82%	52.13%	7.51%	2	12.78%
C2D2	0.82%	49.65%	−9.65%	4	−9.07%
C2D_FUN	0.82%	53.15%	14.37%	1	22.62%

Table 10
Predicting DAX 2019 returns: DAE and DBE loss respectively.

Model	MAE	Accuracy	P&L	Rank	L_P&L
C1D_E	0.65%	54.61%	17.45%	3	21.65%
C1D_LSTM	0.64%	54.49%	17.54%	2	22.77%
C2D1	0.65%	54.83%	16.05%	4	19.12%
C2D2	0.65%	55.15%	17.72%	1	20.93%
C2D_FUN	0.65%	52.04%	15.14%	5	21.49%
Model	MAE	Accuracy	P&L	Rank	L_P&L
C1D_E	0.65%	53.97%	16.48%	4	21.43%
C1D_LSTM	0.64%	54.69%	20.03%	1	26.40%
C2D1	0.65%	52.25%	17.63%	3	22.40%
C2D2	0.65%	56.12%	19.85%	2	20.59%
C2D_FUN	0.65%	52.57%	16.14%	5	22.65%

Let us now explore the cumulated profitability over time, which is summarised in Fig. 8. Every model, except C2D2 trained with DAE, managed to outperform the baselines adopted in this study.

Comparing DAE and DBE training over time, the latter exhibit higher profitability, thanks to significant improvements recorded for C2D based models. The top performing architecture in terms of cumulated profitability is the C2D_FUN with DBE training.

The improvement in profitability with DBE appears more stable for the DAX than the SXXP across periods as seen in Table 12; the DAX is more volatile than the SXXP. The divergence in favour of DBE is lowest in P2, and at its peak when adopting conviction-based trading in P3. Next we see from the historical volatility chart in Fig. 9 that 2020 was the most turbulent year among those considered.

As we see in the rank chart of Fig. 10, non-stationarity is once again revealed: the C2D_fun architecture, pictured in green, was the most profitable in P1, the worst in the following period and the second best in P3!

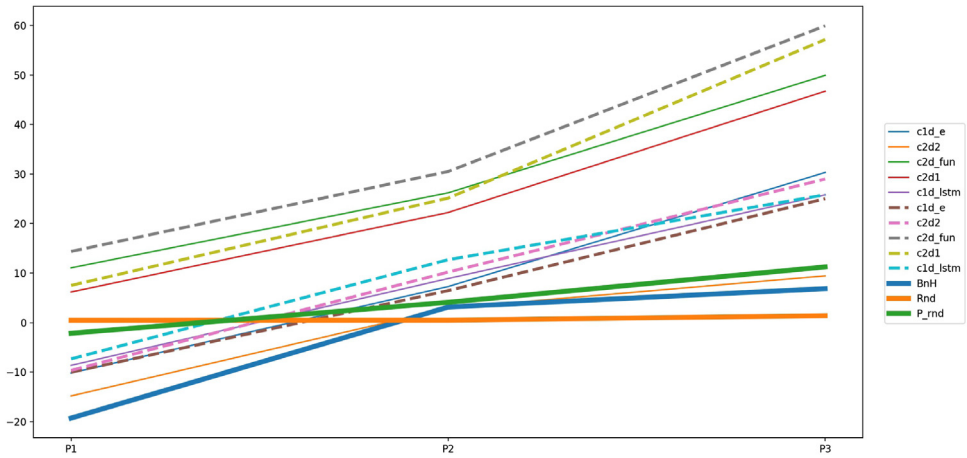

Fig. 8. DAX Cumulated gross profitability. Models beat random strategies (thick lines). DBE-trained architectures (intermittent lines) outperform DAE counterparts.

Fig. 9. DAX realised volatility.
Source: Bloomberg.

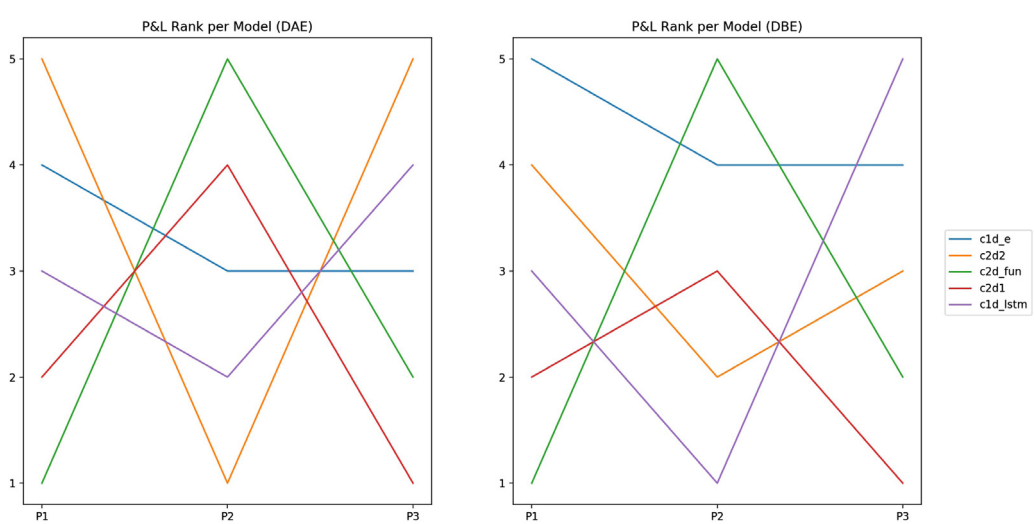

Fig. 10. DAE and DBE training for DAX. Models profitability rank (lower is better) by period.

Table 11
Predicting DAX 2020 returns: DAE and DBE loss respectively.

Model	MAE	Accuracy	P&L	Rank	L_P&L
C1D_E	1.35%	53.68%	23.03%	3	27.29%
C1D_LSTM	1.35%	52.84%	16.90%	4	18.43%
C2D1	1.35%	53.17%	24.49%	1	34.84%
C2D2	1.36%	50.13%	6.53%	5	7.62%
C2D_FUN	1.37%	52.76%	23.72%	2	24.63%
Model	MAE	Accuracy	P&L	Rank	L_P&L
C1D_E	1.35%	53.21%	18.57%	4	22.04%
C1D_LSTM	1.35%	52.95%	13.11%	5	14.36%
C2D1	1.35%	54.05%	32.02%	1	42.01%
C2D2	1.35%	51.27%	18.79%	3	21.17%
C2D_FUN	1.35%	54.21%	29.42%	2	38.90%

Table 12
DAX, difference in average returns between training regimes by period: DBE–DAE.

Period	P&L_Δ	L_P&L_Δ
P1	2.252%	2.848%
P2	1.244%	1.499%
P3	3.446%	5.133%

The issue of model selection, given the peculiarities of these financial markets, is therefore the prominent one. Notwithstanding the development of an auxiliary model to predict market regimes, or the input of a domain expert in identifying similarities between future economic environment and the past, we believe that a plausible course of action is to choose a forecaster based on its resilience (P&L rank) over different periods. Alternatively, ensemble solutions appear to be a promising avenue to solve such algorithmic selection problems.

7. Conclusions

Machine learning techniques cannot simply be applied *pari passu* given the peculiarity of the domain. This study has proposed the adoption of a penalisation mechanism to the mean absolute error loss function for equity market return forecasting problems. In particular, penalising significant errors in a wrong direction was the most effective technique. It would be interesting to apply the penalisation to a different function, e.g., the so-called Huber loss.

The approach of treating a multivariate time series dataset similarly to an image has proven fruitful. The features have been ordered using domain knowledge; it would be interesting to test the impact on C2D models if predictors were to be placed in the feature map at random.

This study successfully demonstrated that trading European equities according to neural networks generated forecasts outperforms random and pseudo-random direction selection. It is reasonable to infer that European equities, perhaps because of the lower amount of capital deployed by systematic strategies compared to the US, have a lower degree of efficiency, and thus exhibit patterns hidden in daily data that can be converted into profitable signals. It would be interesting to corroborate these results with weekly or monthly data in order to go beyond short term trading. Model performance has not been stable over different periods so further research could investigate and potentially leverage the fact that different models have shown different resilience to changing market conditions.

Machine learning models are subject to performance decay if not updated; this is particularly relevant for stock market prediction, given the non-stationarity of data. Further work could attempt to quantify the loss in performance in P&L terms and investigate ways to mitigate the issue by leveraging models that have a low rank at different points in time.

We have run simulations on 5 increasingly more complex, convolution-centred architectures. Further research might seek to further raise complexity, e.g., adopting transformers architectures.

This study has tackled SXXP and DAX returns forecasting by developing specific models for each case. It would be interesting to compare findings with a 'global' model, trained on both indices at once. It is reasonable to suppose that more training samples would allow for a larger feature map and/or more complex models.

A specific hurdle we faced in this research, and which will also affect further efforts, is the lack of a common set of institutional investor-grade multivariate timeseries datasets. This lack of unified data results in a heterogeneous body of research were studies consider different data, spanning over different periods and/or different features (Matuozzo et al., 2022). It is therefore hard to comparatively assess research findings. Common datasets should be created and made publicly available to foster cooperation and advancement.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data is available upon request.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I. J., Harp, A., Irving, G., Isard, M., Jia, Y., Józefowicz, R., Kaiser, L., Kudlur, M., ... Zheng, X. (2016). TensorFlow: Large-scale machine learning on heterogeneous distributed systems. CoRR arXiv:1603.04467, URL http://arxiv.org/abs/1603.04467.

Bernstein, P. L. (1999). Why the efficient market offers hope to active management. *Journal of Applied Corporate Finance*, *12*, 129–136.

Borovykh, A., Bohté, S. M., & Oosterlee, C. W. (2018). Conditional time series forecasting with convolutional neural networks.

Cohen, N., Balch, T., & Veloso, M. (2020). Trading via image classification. In T. Balch (Ed.), *ICAIF '20: The first ACM international conference on AI in Finance, New York, NY, USA, October 15-16, 2020* (pp. 53:1–53:6). ACM, http://dx.doi.org/10.1145/3383455.3422544.

de Prado, M. L. (2018). *Advances in financial machine learning*. O'Reilly.

Dixon, M. F., Halperin, I., & Bilokon, P. (2020). *Vol. 1406*, *Machine learning in finance*. Springer.

Eapen, J., Bein, D., & Verma, A. (2019). Novel deep learning model with CNN and bi-directional LSTM for improved stock market index prediction. In *IEEE 9th annual computing and communication workshop and conference, CCWC 2019, Las Vegas, NV, USA, January 7-9, 2019* (pp. 264–270). IEEE, http://dx.doi.org/10.1109/CCWC.2019.8666592.

Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The Journal of Finance*, *25*(2), 383–417, URL http://www.jstor.org/stable/2325486.

Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, *270*(2), 654–669. http://dx.doi.org/10.1016/j.ejor.2017.11.054.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. The MIT Press.

Grossman, S. J., & Stiglitz, J. E. (1982). On the impossibility of informationally efficient markets. *American Economic Review*, *72*, 393–408. http://dx.doi.org/10.7916/D8765R99.

Gu, S., Kelly, B., & Xiu, D. (2020). Empirical asset pricing via machine learning. *The Review of Financial Studies*, *33*(5), 2223–2273.

Gudelek, M. U., Boluk, S. A., & Ozbayoglu, A. M. (2017). A deep learning based stock trading model with 2-D CNN trend detection. In *2017 IEEE symposium series on computational intelligence, SSCI 2017, Honolulu, HI, USA, November 27 - Dec. 1, 2017* (pp. 1–8). IEEE, http://dx.doi.org/10.1109/SSCI.2017.8285188.

Guida, T. (2018). *Big data and machine learning in quantitative investment*. Wiley.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780. http://dx.doi.org/10.1162/neco.1997.9.8.1735.

Jegadeesh, N., & Titman, S. (1993). Returns to buying winners and selling losers: Implications for stock market efficiency. *The Journal of Finance*, *48*(1), 65–91. http://dx.doi.org/10.1111/j.1540-6261.1993.tb04702.x, URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-6261.1993.tb04702.x.

Kehinde, T., Chan, F. T., & Chung, S. (2023). Scientometric review and analysis of recent approaches to stock market forecasting: Two decades survey. *Expert Systems with Applications*, *213*, Article 119299.

Matuozzo, A., Yoo, P. D., Provetti, A., & Kim, M. H. (2022). Machine learning methods for equity time series forecasting: A compendium. In *CIKM'22: Workshop on applied machine learning methods for time series forecasting (AMLTS), October 21, 2022, Atlanta, GA, USA*.

Moskowitz, T. J., Ooi, Y. H., & Pedersen, L. H. (2012). Time series momentum. *Journal of Financial Economics*, *104*(2), 228–250.

Murphy, J. J. (1986). *Technical analysis of the futures markets: A comprehensive guide to trading methods and applications*. Prentice Hall.

Nelson, D. M., Pereira, A. C., & De Oliveira, R. A. (2017). Stock market's price movement prediction with LSTM neural networks. In *2017 international joint conference on neural networks* (pp. 1419–1426). Ieee.

Pedersen, L. H. (2019). *Efficiently inefficient: how smart money invests and market prices are determined*. Princeton University Press.

Qian, X.-Y., & Gao, S. (2017). Financial series prediction: Comparison between precision of time series models and machine learning methods.

Sezer, O. B., & Ozbayoglu, A. M. (2018). Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach. *Applied Soft Computing*, *70*, 525–538.

Sheraz, M., Dedu, S., & Preda, V. (2015). Entropy measures for assessing volatile markets. *Procedia Economics and Finance*, *22*, 655–662.

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In Y. Bengio, & Y. LeCun (Eds.), *3rd international conference on learning representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, conference track proceedings* (pp. 1–14). URL http://arxiv.org/abs/1409.1556.

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, *15*, 1929–1958.

Tompson, J., Goroshin, R., Jain, A., LeCun, Y., & Bregler, C. (2015). Efficient object localization using convolutional networks. In *IEEE conference on computer vision and pattern recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015* (pp. 648–656). IEEE Computer Society, http://dx.doi.org/10.1109/CVPR.2015.7298664.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A. W., & Kavukcuoglu, K. (2016). WaveNet: A generative model for raw audio. In *The 9th ISCA speech synthesis workshop, Sunnyvale, CA, USA, 13-15 September 2016* (p. 125). ISCA, URL http://www.isca-speech.org/archive/SSW_2016/abstracts/ssw9_DS-4_van_den_Oord.html.

Zeng, Z., Balch, T., & Veloso, M. (2021). Deep video prediction for time series forecasting. In A. Calinescu, & L. Szpruch (Eds.), *ICAIF'21: 2nd ACM international conference on AI in finance, virtual event, November 3 - 5, 2021* (pp. 39:1–39:7). ACM, http://dx.doi.org/10.1145/3490354.3494404.