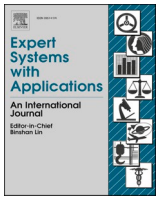
Expert Systems With Applications 237 (2024) 121404

Contents lists available at ScienceDirect

Expert Systems With Applications

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



Deep learning in stock portfolio selection and predictions

Chaher Alzaman

Department of Supply Chain and Business Technology Management, John Molson School of Business, MB 12-107, John Molson School of Business, Concordia University, 1450 Guy, Montreal, Quebec H3G-1M8, Canada

A R T I C L E I N F O

Keywords:
Machine learning
Deep learning
Financial markets
Portfolio management
Neural networks
Deep RankNet

A B S T R A C T

Deep learning (DL) has made its way into many disciplines ranging from health care to self-driving cars. In financial markets, we see a rich literature for DL applications. Particularly, investors require robust algorithms that can navigate and make sense of extremely noisy and volatile markets. In this work, we use deep learning to select a portfolio of stocks and use a genetic algorithm to optimize the hyperparameters of DL. The work analyzes the improvement in using genetic-based hyperparameter optimization over grid searches. The Genetic Algorithm brings 40% improvements in prediction when compared to a random-grid search. Novelty-wise, the work couples a genetic-based hyperparameter optimization with multiple Deep RankNet models to predict the behavior of financial assets. Our results show promising portfolio returns 20% better than the general market. In the highly volatile COVID 19 period, the models exceed market returns by more than double. Overall, this paper brings a comprehensive work that integrates hyperparameter optimization, Deep RankNet, LSTM, period size variations, input variable transformation, feature selection, training/evaluation ratio analysis, and multiple portfolio selection strategies.

1. Introduction

Shah (2007) disserts two focal stock prediction methodologies. The first is Fundamental Analysis. Here, practitioners perform analysis based on the company's economical credentials (past performance, assets, earnings, etc.) rather than stock value. The second is Technical Analysis. Here, experts put all their efforts into studying past stock price patterns. The efficient market hypothesis reminds us that it is impossible to predict the price of financial assets (Jensen, 1978). Further, the Random-walk hypothesis states that a stock price changes independently of its history. So, the price of a given stock tomorrow will only depend on tomorrow's information regardless of today's price (Malkiel, 1973). However, the work of Chen, Zhang, Mehlawat, and Jia (2020) and Basak, Kar, Saha, Khaidem, and Dey (2019), and many others beg to be different. Where they advocate that certain elements in stock behavior are somewhat predictable. It is also important to note the work of Kirkpatrick and Dahlquist (2011) in behavioral finance (behavioral analysis) where they argue against the assumption of investor actions' rationality. Hence, supporters of technical analysis advocate this argument by iterating that rationality is not constantly dominant in investor decisions and hence markets are not always consistent with the random walk and efficient market theories (Rousis & Papathanasiou, 2018).

In stock market prices, Liu and Yeh (2017) emphasize that the relations between the weights of stocks and portfolio performance are nonlinear. Deep learning algorithms are very much suited for nonlinear behaviors since they encompass activation functions that presume nonlinear behavior. Hence, deep learning, and precisely neural networks, tend to be an excellent tool to handle the challenge since they are very suited for nonlinear models. Deep Learning is a subset of Machine Learning (ML) geared toward more complex systems (e.g., big data). Here, the learning is carried out by artificial neural networks (ANN). LSTM (long-term memory) is an artificial recurrent neural network geared towards sequential data, such as closing prices of financial assets. Hyperparameter optimization is the optimization of important neural network parameters such as the number of neurons, number of hidden layers, batch size, learning rate, and others to maximize the performance of learning algorithms. These parameters are usually set by the user before training. However, for complex systems such as financial systems, the optimal setting of these parameters is of great importance as financial markets' data are known to be noisy and quite sensitive to small variations in the input.

In this work we first identify optimal hyperparameters using a genetic algorithm, then we use LSTM and Deep RankNet to pick individual stocks from the Toronto Stock Exchange, TSE, into portfolios for cyclical trading. The TSE is used for numerous motives. First, there is a lack of works that study the TSE. Second, unlike American and Chinese

E-mail address: chaher.alzaman@concordia.ca.

https://doi.org/10.1016/j.eswa.2023.121404
Received 17 May 2022; Received in revised form 7 March 2023; Accepted 29 August 2023
Available online 9 September 2023

financial markets, which are predominant in the state-of-the-art, the TSE is greatly influenced by oil/gas stocks and commodity-based stocks in general. This might bring results that might can supplement the research. Third, the TSE has witnessed recent market shocks due to the plummeting of oil/gas prices during the 2019–2021 period (COVID 19). It would be interesting to observe what impact that has on the performance of different deep learning algorithms. Largely, this work assesses the performance of distinct deep learning models. Explicitly, the paper demonstrates the improvements resulting from the integration of hyperparameter optimization.

This work aims to predict the movement of financial assets by the use of innovative deep matching (ranking) algorithms, the integration of feature engineering and hyperparameter optimization, and risk matrices analysis; in addition to a comparative study of a non-ranking deep learning algorithm (LSTM). To the best of our knowledge, there has not been a work that combines hyperparameter optimization and Deep RankNet. Overall, the paper integrates portfolio selection, hyperparameter optimization, period size variations, input variable transformation, technical indicator analysis, feature selection, training/evaluation ratio analysis, and multiple indices for gauging the validity and variation in the results.

The structure of the paper starts with the literature review. The next section highlights the methods used in this work where a background on LSTM, Grid search, and genetic algorithm for hyperparameter optimization are supplied. The subsequent section details the results. Finally, we end the work with a conclusion highlighting contributions and future research.

2. Literature review

Many works in the area of Financial Markets ML applications conduct a comparative analysis between different ML techniques. In this realm, numerous works evaluate different ML techniques to choose the best performing algorithms given an explicit financial asset. To name a few, the works of Basak et al. 2020, Hegazy et al. (2014), Ismail, Noorani, Ismail, Razak, and Alias (2020), Kumar, Dogra, Utreja, and Yadav (2018), Nelson, Pereira, and de Oliveira (2017), Patel, Shah, Thakkar, and Kotecha (2015), Shen, Jiang, and Zhang (2012), Singh and Srivastava (2017), Usmani, Adil, Raza, and Ali (2016), and Vijh, Chandola, Tikkiwal, and Kumar (2020) contrast different techniques with the respect to the criterion of market prediction accuracy. Next in importance, we see works that focus on one ML technique and improve its prediction performance (Akita, Yoshihara, Matsubara, & Uehara, 2016; Kim, Ku, Chang, & Song, 2020; Mehtab, Sen, & Dutta, 2020; Nabipour, Nayyeri, Jabani, Shahab, & Mosavi, 2020; Porshnev, Redkin, & Shevchenko, 2013; Reddy, 2018; Vazirani, Sharma, & Sharma, 2020).

For Portfolio Selection, we start with the work of Chen et al. (2020). They use an XGBoost algorithm with multi-features, including technical indicators. They perform feature selection and hyperparameter optimization. Their work focuses on the Stock Exchange (SE) of Shanghai. Second, the work of Liu and Yeh (2017) utilizes neural networks to predict American-based stocks. Their work doesn't perform hyperparameter optimization nor does it perform feature selection. The strength of their work is in the portfolio selection methodology. Paiva, Cardoso, Hanaoka, and Duarte (2019) introduce a Support Vector Machine (SVM) algorithm to predict financial assets' predictions for stocks traded in Sao Paulo. They perform hyperparameter optimization to better tune the SVM. Long, Lu, and Cui (2019) introduce a multi-filters neural network (MFNN) geared towards feature extraction and price movement prediction for financial time series data. They start by utilizing convolutional and recurrent neurons to extract features from data samples. For the recurrent neural network, they use LSTM. One important thing to note is that they employ a 1-minute stock price frequency data set. This could be a limitation since medium and small investors might not have access to such meticulous data, especially for smaller trading markets.

Guo, Rao, Zhu, and Xu (2020) present a novel deep matching algorithm that integrates stock profiles (including textual input). Their resulting portfolio strategies outperform the CSI300 index (Shanghai Stock Exchange). Li and Tan (2021) provide a Deep Rank network to rank stocks concerning their excess returns. The work utilizes a Deep RankNet, classification, and regression models. The work provides a detailed procedure for a classification, ranking, and neural network scoring model.

They deploy a rolling time window and specify the length of the input duration and holding period. They focus on 10 ($k = 10$) stock picks per portfolio. They find the Deep RankNet methodology to be far superior, however, they do not contest different k's or integrate hyperparameter optimization. This work shall do both. Aside from portfolio selection works, we see numerous works that utilize deep learning for stock predictions. To list a few, we have the works of Akita et al. (2016), Kim et al. (2020), Liu and Yeh (2017), Nabipour et al. (2020), and Nelson et al. (2017). Akita et al. (2016) utilize LSTM for numerical data (stock prices) combined with Paragraph Vector for textual (newspaper) data. Nelson et al. (2017) use LSTM and focus on the movement of stocks (predicting if the price of a stock is going up or down). In addition, their work prophesizes actual buying strategies rather than theoretical assessments. Table 1 depicts the inclusion of important factors that this work will integrate. The table organizes the literature according to the Machine Learning (ML) algorithm of choice. This is present in the second and seventh columns of the work. Here we focus on the algorithm the author(s) choose for predictive analytics or the main algorithm(s) which the work analyze. The third and eighth columns present the integration or non-integration of hyperparameter optimization. While the subsequent columns show the integration or null case of batch size and training ratio optimization.

Starting with general hyperparameter optimization, we find the work of Basak et al. (2020), Chen et al. (2020), Hegazy et al. (2014), Mehtab et al. (2020), Nabipour et al. (2020), and others where hyperparameters are optimized using numerous tools such as random search, grid search, genetic-based search, Bayes-based search, and gradient-based search. However, the inclusion of certain factors is quite variable in the state of the art.

For recent works, we see many advantages to the use of deep learning in financial asset behavior predictions. The works of Kanwalet et al. (2022), Sharma and Shekhawat (2022), Shi et al. (2022), and many more listed in Table 1 attest to the power of deep learning in deciphering hidden patterns in the financial markets. We also observe works that push the envelope in terms of using novel deep learning approaches in the financial markets. To name a few, Lin, Chen, Sang, and Huang (2022) and Shi et al. (2022) utilize deep reinforced learning, Li et al. (2022) employ *sequence to sequence ranking* model, Sharma and Shekhawat (2022) exploit Restricted Bolzmann Machines, and Wu, Wang, and Wu (2022) apply autoencoder to denoise the data. Further, the-state-of-the-art is, more and more, exploiting risk matrices to quantify the financial return to risk. To name a few, the works of Zhao, Bai, Fang, and Wang (2022), Ma and Tan (2022), Leippold, Wang, and Zhou (2022), and Cuomo, Gatta, Giampaolo, Iorio, and Piccialli (2022) employ risk matrices and focus on the Sharpe ratio as a measure for performance. While in the comprehensive literature review of Dessain (2022), the importance of moving away from accuracy measures and more towards returns to risk matrices is stressed. Ren, Xu, and Duan (2022) concluded that utilizing the Fourier transform-based LSTM technique enhances the accuracy of predicting stock volatility dynamics, based on both statistical and economic perspectives. Overall, the work analyzes the effect of various oil shocks on the volatility of studied stocks. Acuna-García, Luz Canchola-Magdaleno, and Olmos Trejo (2022) propose a model for predicting short and medium-term prices of financial stocks based on a combination of Continuous Wavelet Transform and Neural Networks with Long Short-Term Memory. The model utilizes the most representative coefficients of the wavelet transform based on the time series to forecast future prices of stocks. Dezhkam and

Table 1
Factors inclusion in the state of the art (Deep Belief Net: DBN, Restricted Bolzmann Machines: RBM, Exponential Gradient: Exp. Grad, Deep Reinforced Learning: DRL, Unsupervised Learning: USL, Support Vector Machines: SVM, Artificial Neural Network: ANN, Graph Convolutional Network: GCN, Random Forest: RF).

Work	ML Algorithm of choice	Hyper-param	Batch Size	Ratio Analysis	Work	ML Algorithm of choice	Hyper-param	Batch Size	Ratio Analysis
Acuna-García et al. (2022)	Tranform-based	X	X	X	Ren et al. (2022)	Transform-based	X	X	X
Akita et al., 2016	LSTM	X	X	X	Ma & Tan, 2022	Multiple	X	X	X
Basak et al. 2020	XGBoost, RF	√	X	X	Ma et al., 2021	Multiple	√	X	X
Chen et al. (2020)	XGBoost	√	X	X	Mehtab et al. 2020	LSTM	√	X	X
Coqueret, 2022	Multiple	X	X	X	Nabipour et al., 2020	RNN & LSTM	√	X	X
Cuomo et al., 2022	USL	√	X	X	Nelson et al., 2017	LSTM	X	X	X
Dai, Liang, Dai, Huang, & Adnan, 2022	ANN	X	X	X	Paiva et al. (2019)	SVM	√	X	X
Dessain, 2022	Multiple	X	X	X	Patel et al. (2015)	RF	√	X	X
Du, 2022	Multiple	X	X	X	Pinelis & Ruppert, 2022	RF	X	X	X
Hegazy et al. (2014)	SVM	√	X	X	Porshnev et al. (2013)	SVM & ANN	X	X	X
Ismail et al. (2020)	Multiple	X	X	X	Reddy, 2018	SVM	X	X	X
Kanwal, Lau, Ng, Sim, & Chandrasekaran, 2022	ANN	√	√	X	Sharma & Shekhawat, 2022	DBN & RBM	X	X	X
Kim et al., 2020	ANN, LSTM	√	X	X	Shen et al. (2012)	SVM	X	X	X
Kumar et al. (2018)	RF	X	X	X	Singh et al. (2017)	DNN	√	X	X
Leippold et al., 2022	Multiple	X	X	X	Shi et al., 2022	GCN	X	X	X
Li, Zheng, Chen, Wang, & Xu, 2022a	Exp. Grad.	X	X	X	Usmani et al. (2016)	ANN	√	X	X
Li, Fu, Zhao, & Yang, 2022b	Multiple	X	X	X	Vazirani et al., 2020	Hybrid Regression	X	X	X
Li and Tan (2021)	Deep RankNet	X	X	X	Vijh et al. (2020)	RF and ANN	X	X	X
Lin et al., 2022	DRL	X	X	X	Yadav, Jha, & Sharan, 2020	LSTM	√	X	X
Liu & Yeh, 2017	NN	X	X	X	Zhao et al., 2022	Bayesian	X	X	X

Manzuri (2023) introduce a model labelled HHT-XGB, for predicting the changing trends in the next closing stock price. This model combines Hilbert-Huang Transform (HHT) and extreme gradient boost (XGBoost) techniques. HHT is used for feature engineering, while XGBoost classifies close price trends. The model produces a sequence of ups and downs, which is used to optimize the stocks' portfolio weights for best trading performance.

On the other hand, Table 1 illustrates a gap in the research. Most of the works do not explicitly employ hyperparameter optimization (Zhao et al., 2022, Coqueret, 2022, Du, 2022, and more). One of this work's main contributions is the explicit use of a genetic-based algorithm to tune the machine learning hyperparameters. Dessain (2022) classifies predicting returns performance metrics, within the context of machine learning, into error-based, accuracy-based, and investment-based metrics. The first metric is associated with a given algorithm prediction error. The second measures the returns based on the algorithm's prediction. The last is concerned with result-based metrics and risk-adjusted return-based metrics. This work employs all three categories and focuses on risk-adjusted return-based metrics.

3. Methods

The methods used in this work mainly encompass three algorithms: LSTM, Deep RankNet, and Genetic-based search. The aim is to predict the future returns of financial assets. The work aspires to tackle the following research questions in the presence of hyperparameters' genetic-based optimization:

I. Among batch size, training/evaluation ratio, rolling time window characteristics, and others, which of the hyperparameter(s) are significant in their influence?
II. Does LSTM perform better than Deep RankNet (TSE).
III. Does Varying k (number of financial assets in a portfolio) improve/affect the performance of Deep RankNet (i.e. reduce prediction error)?
IV. Does the profile of the trading market greatly affect the performance of Deep RankNet models?
V. How much does the volatility of the trading period affect the performance of the Deep RankNet models?

Before venturing into LSTM, we need to give a brief description of what artificial neural networks and recurrent neural networks are. Artificial Neural Networks (ANN), often referred to as just Neural Networks, are a simplified abstraction of the human brain's complex network of neurons. In ANN, neurons are processing units that perform some predefined mathematical operations (Sharda, Delen, & Turban, 2020). Since human learning and understanding are highly related to context and sequential information, Recurrent Neural Networks (RNN) are designed to process sequential inputs. In essence, RNNs are dynamic systems where a neuron at each time point depends on both the input of the system at that time and its state in previous periods (Sharda et al., 2020). One of the major flaws in RNN is memory loss. Particularly in long sequences, the algorithm will have a hard time carrying all the information from the previous step to later steps. In addition, RNNs suffer from what is called, 'the vanishing gradient problem.' Here the gradient gets smaller and smaller (vanishes) as it back propagates through time steps. This continues till the gradient become too small, and insignificant, and hence no substantial learning is presumed. This usually occurs in the earlier periods, earlier layers. Therefore, RNNs forget what it has seen in their long-term memory. Hochreiter and Schmidhuber (1997) introduce the Long Short-Term Memory (LSTM) network. It constitutes cell states and gates where the first act as a highway for relative information transfer and the latter filters (decides) the important information from non-important ones. Fig. 1 illustrates a schematic for LSTM, where multiple gates encompass activation functions. It closely follows the work of Fischer and Krauss (2018). LSTM can be thought of as a system of cells where information gets added or removed to the cell state by the use of gates depending on its importance (important information kept, irrelevant information discarded).

We utilize 42-time steps (t), where at every t the input is posed as x_t and an output h_{t-1} of the memory cell at the previous t (t-1). As per the

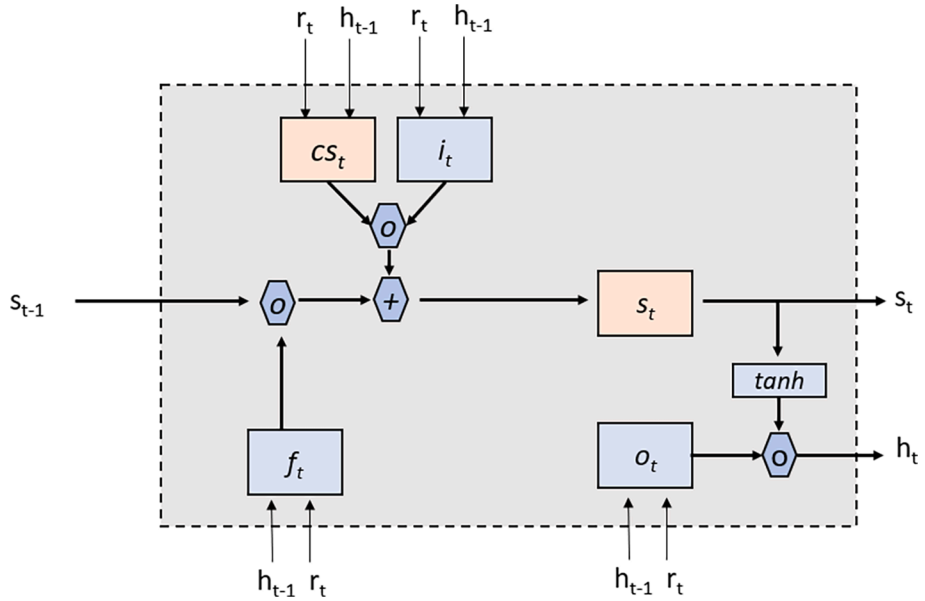


Fig. 1. LSTM Architecture of memory cell (f, forget; i, input cell; cs, candidate state value; s, cell state; o, output cell).

work of Fischer and Krauss (2018), we utilize the stock return as an input parameter based on the given stock closing P (P_t) for a specific period t.

$$R_t = \frac{P_t}{P_{t-1}} - 1 \tag{1}$$

In essence, the gates act as filters (see Fig. 1). The forget gate dictates which information to remove from the memory cell state. While the input gate ushers which information to add to the memory cell state. Finally, the output gate directs which information from the memory cell state to utilize as output. The equations below characterize the LSTM procedure in a vector form. Candidate state shall be represented as *cs*, while state cell is indicated by *s* alone. Input and output values are designated as *i* and *o*.

- r_t is an input vector at time step (t).
- The weight matrices are $W_{f,r}$, $W_{f,h}$, $W_{cs,r}$, $W_{cs,h}$, $W_{in,h}$, $W_{o,r}$, and $W_{o,h}$.
- The bias vectors are b_f, b_{cs}, b_{in}, and b_o.
- Activation function vectors are f_t, i_t, and o_t for the three gates respectively.
- The output vector for the LSTM layer is h_t.

The cell states and output are updated using the following procedure. First, LSTM needs to decide which information to discard. This is done using the activation function (Eq. (2)).

$$f_t = sigmoid(W_{f,r}r_t + W_{f,h}h_{t-1} + b_f) \tag{2}$$

Second, the LSTM layer decides which information to be added to the cell states in two parts. First, the candidate state value is computed (Eq. (3)). Second, the activation value of the input gate is computed (Eq. (4))

$$cs_t = tanh(W_{cs,r}r_t + W_{cs,h}h_{t-1} + b_{cs}) \tag{3}$$

$$i_t = sigmoid(W_{i,r}r_t + W_{i,h}h_{t-1} + b_i) \tag{4}$$

In the third step, the new cell states s t are calculated based on the results of the previous two steps ∘denoting the Hadamard (elementwise) product:

Then, the Hadamard product is used to arrive at the new cell state value (Eq. (5)).

$$s_t = f_t \bullet s_{t-1} + i_t \bullet cs_t \tag{5}$$

Finally, the output of the memory cell is computed using Eqs. (6) and (7).

$$o_t = sigmoid(W_{o,r}r_t + W_{o,h}h_{t-1} + b_o) \tag{6}$$

$$h_t = o_t \bullet \tanh(s_t) \tag{7}$$

The LSTM in this paper is a point-wise prediction algorithm (Ma, Han, & Wang, 2021) compared to the Deep RankNet, which is a List-wise stock ranking (Ma et al., 2021). In this way, the gains and/or losses of a ranking algorithm can be clearly distinguished.

Deep RankNet has been designed to learn the relative performance of documents/queries in a pairwise manner. It was pioneered by Burges et al. (2005). In their work, they use a probabilistic cost function that utilizes a pair of sample objects to instigate and learn how to rank documents/objects. The cost function effectively seeks to minimize the number of pairing instances needed to correctly order a set of items. Given the challenges of financial asset predictions, Li and Tan (2021) bring forward a Deep RankNet to rank the performance of financial assets and ultimately bundle high-performing stocks into trading portfolios. The model instituted in this work will implement the framework introduced by Li and Tan (2021). However, we integrate novel elements such as multi-size portfolios, where k (the number of stocks within a trading portfolio) is varied. Varying the value k can be useful for small traders, who might favor lower k due to its low start-up capital advantage, fixed transaction cost avoidance, and operational trade simplicity.

To start, an excess return is computed for a given stock concerning the composite index (market index). Here we contest 100 stocks all traded in the TSE. The equations for the return rate of a given stock, $R_{t,s}$, return of the index $R_{t,I}$, and excess return, ER (Li & Tan, 2021) are presented below.

$$R_t^s = \frac{P_t^s}{P_{t-1}^s} - 1 \tag{8}$$

$$R_t^I = \frac{P_t^I}{P_{t-1}^I} - 1 \tag{9}$$

$$ER_{t,m}^S = R_{t,m}^s - R_{t,m}^I \tag{10}$$

R = Return made by a given stock ($s \in S$); P = Closing Price a stock (or index i); t = trading day; ER = Excess return of stock s relative to index i, for holding period m; Fig. 2 illustrates the Deep RankNet

→Compute excess return for all N_{stocks}
→Rank Stocks' returns from highest to lowest
→Assign top 50% stocks a ranking label of *1_Class 1*
→Assign lowest 50% stocks a ranking label of *0_Class 2*
→Feed the *ER's* into a neural deep network as input
→Input Matrix is N_{stocks} × N_{days} (number of stock × number of days)
→→Each stock entry includes 20 days (N_{days}), past *ER's*
→Initiate Neural Network outputs
→→*Score 1* for *Class 1* data
→→*Score 2* for *Class 2* data
→→→Subtract *score 2* from *score 1* to produce *Dif*
→→→Input *Dif* into a Sigmoid function
→Apply loss cross entropy function
→Apply gradient descent
→Update Neural Network parameters
→Output neural network scoring model

Fig. 2. Deep RankNet procedure.

procedure used in this work. The procedure follows closely the work of Li and Tan (2021).

Genetic Algorithm (GA) is an optimization and search technique based on the principles of genetics and natural selection. Pertaining to this work, the neural network's hyperparameters are numerous and quite difficult to tune (we should demonstrate in the later section that a random grid search doesn't yield good results, and hence a more sophisticated algorithm is needed). We start by populating an *N* random instances matrix for *M*−vector elements. We shall label each row in Fig. 3 (schematic on the left) as a chromosome, where each chromosome consists of a set of values for the hyperparameters in the question. We run the neural network for each chromosome and produce a unique prediction error. Then, we sort the chromosomes from lowest error value to highest. Next, the top half of our population table/matrix (Fig. 3) is chosen for mating (i.e., natural selection). Mating produces offspring that will populate the top of the table while the parents (top half of the previous iterate) will populate the bottom half. This way, if the parents turn out to be more effective than their offspring, they are not lost in the process (Goldberg, 1989; Holland, 1975).

Natural Selection plays a major role in improving the fitness of the population of solutions and in achieving the best solution. The selection rate is the fraction of the population which survives for the next generation (in our case, it is 0.5). Two chromosomes are selected from the mating pool of chromosomes to produce two new offspring. Mating (see right side schematic in Fig. 3) is the creation of one or more offspring(s) from the parents selected in the pairing process. A crossover point is randomly selected between the first and last bits of the parents' chromosomes. First, parent 1 passes its numeric values to the right of the crossover point. In a like manner, parent 2 passes its numeric values to the left of the same crossover point. Consequently, the offspring has a portion of the numeric codes of both parents.

Mutations in GA ensure that the algorithm is not trapped in a local minimum. Random mutations alter a certain percentage of the bits in the list of chromosomes. It can introduce traits, not in the original population and keeps the GA from converging too fast before sampling the entire cost surface. So, in each generation, we introduce 4% new random chromosomes to assure that a broader exploration of the solution space is warranted.

Hence, the work includes the following elements shown in the flow/block diagram (see Fig. 4). We start with a genetic and grid-search hyperparameter optimizations. Then, the more robust algorithm is chosen for hyperparameter tuning (the diamond objects represent comparative analysis phases). Then, important features are chosen as input based on the state-of-the-arts and preliminary runs. The input and output are prepared for both the LSTM and Deep RankNet. It is important to note that the input features are unique to the algorithm. Given that the Deep RankNet utilizes ranking, the input features are quite different to the non-ranking LSTM. Then, a comparative study is initiated between the Deep RankNet and LSTM. Afterward, the work contests different Deep RankNet scenarios based on different *k* stocks. Finally, a COVID 19 period stress test is applicated to see the performance of the RankNet algorithms in times of market turmoil.

4. Results

To begin, we exclusively study stocks traded on the Toronto Stock Exchange (TSE). We employ 100 TSE stocks (closing prices based on yahoo finance Python's API) for the study's period. The work necessitates complete data across all periods in the study. Hence, numerous preliminary steps were performed to eliminate financial assets with missing data, irresponsive updating characteristics (given the used API), and inconsistencies. Table 2 represents a vast topography of financial

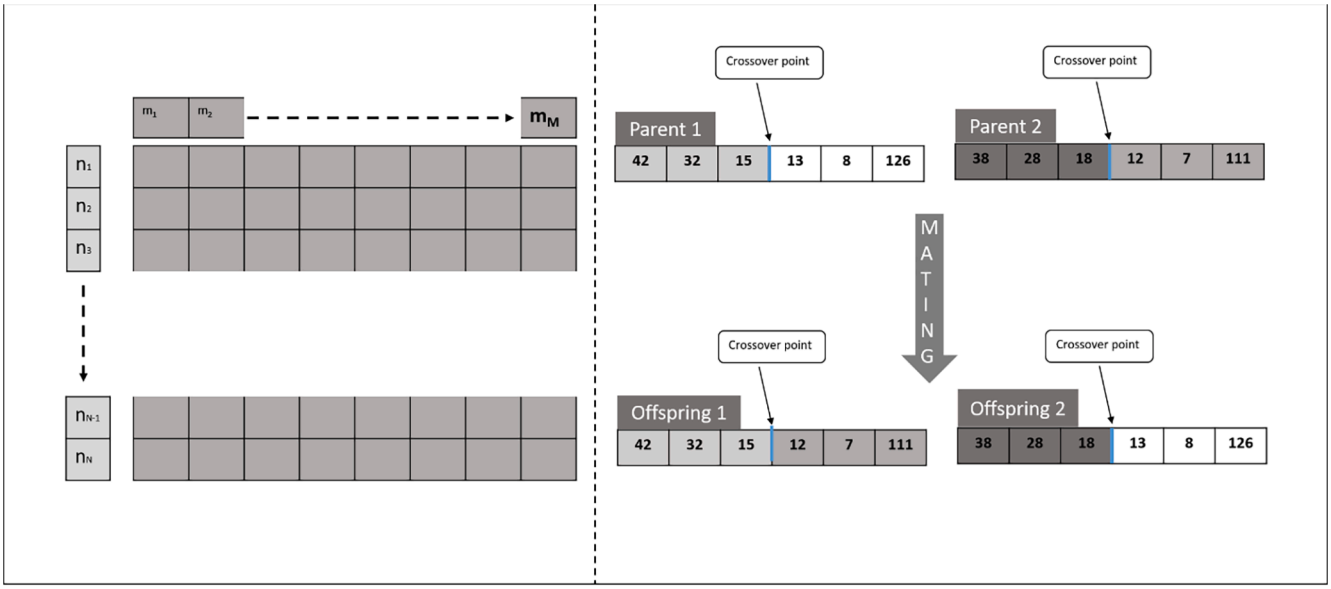


Fig. 3. Depiction of Chromosomes (Left) and Mating (Right) in GA.

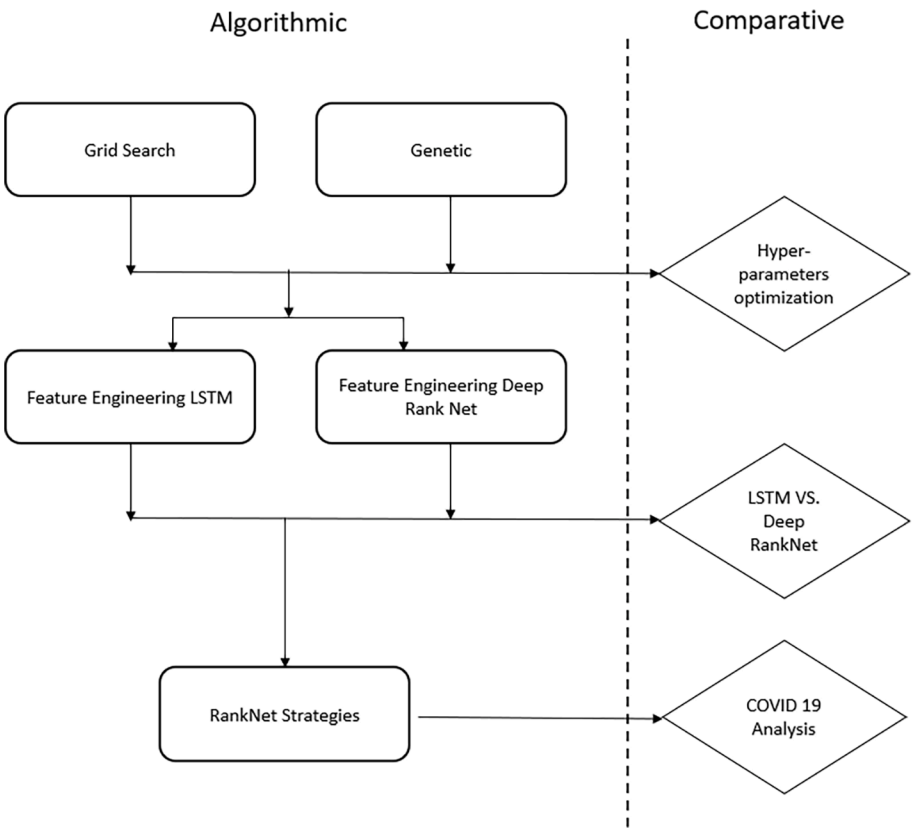


Fig. 4. Block Diagram of the paper's methodological framework.

Table 2
Stocks used in the study (all traded in TSE).

Industrial Field	Industrial Field
Technology, software application	Industrial commodities, Metals, and Materials
Financial services, mortgage Finance	Consumer, Restaurants
Energy, Oil/Gas	Healthcare, Drug Manufacturer
Healthcare, medical care facilities	Financial services, Asset Management
Financial services	Technology, Communication Equipment
Consumer, Apparel Retail	

assets used in this work.

In preliminary runs, the importance of setting LSTM's hyper-parameters such as batch size, the number of neurons, and the number of hidden layers was noted. We start with a random-grid search for hyperparameter optimization, followed by a genetic-based optimization with the integration of technical indicators. In the literature, a high variety of technical indicators was used. To list a few, the works of Akita et al. (2016), Basak et al. (2020), Chen et al. (2020), and Hegazy et al. (2014) integrate technical indicators as features in their respective ML models. In this work, we identify, and later use, four technical indicators that are prominent in the literature: Moving Average (MA), Exponential Moving Average (EMA), Moving Average Convergence/Divergence (MACD), and Relative Strength Index (RSI) (Ayala, García-Torres, Noguera, Gómez-Vela, & Divina, 2021). These factors have brought positive results in the literature and are in common among numerous works (Akita et al., 2016, Basak et al. 2020, Chen et al., 2020, Ayalta et al. 2021). Thereafter, results are presented for the LSTM and Deep RankNet models. Subsequently, the efficacy of different k portfolio size models is presented with a special case on the turbulent COVID 19 period.

4.1. Random grid

To get a feel for the sensitivity of the model to hyperparameters, we start with a random-grid search. The hyperparameters we entertain are the training to test ratio, the batch size, the number of neurons, the stopping criterion tolerance, the number of lags, the number of hidden layers, and the number of days in the period. Before we jump into the results, let's introduce all the said hyperparameters. To start, the batch size is the number of samples that are passed through the network at one time. Larger batch sizes usually mean faster training since the machine is processing more data at once. However, larger batch sizes can sacrifice the quality of the training. It might lead the model to generalize on data it has not seen before. Hence, the batch size needs to be tuned to balance the trade-off between the training speed and the quality of training. The number of neurons is also another parameter that can be tuned. A higher number of neurons adds complexity to the neural network but sometimes at the cost of overfitting the data. Next is the stopping criterion. Usually, the stopping criterion for a neural network algorithm is not specified and the duration of the training is governed by the number of epochs. Practitioners set the epoch to a high number to assure effective training. However, for certain models, the number of epochs might make the model terminate prematurely. Having a higher number of epochs might mitigate this risk but at the expense of missing the optimal number of training epochs. As a result, the number of hidden layers brings about complexity to the neural networks but sometimes at the cost of overfitting. We will contest a series of different hidden layer configurations and see the impact they have on the predictions. Next, the number of lags is quite specific to time-series predictions. In time-series predictions, the lags are used as independent variables. Since stock trading prices are a form of univariate time series, autocorrelation can be present between one day and the previous. Hence, LSTM allows for lagged data (previous days' prices) in its architecture. However, the number of lags is quite challenging to set. We need to contest small and

large ranges of lags. Finally, we look at the number of periods included in the LSTM training set. The literature seems to favor a longer period window for training. However, this might not be practical, since newer stocks might only be one to three years old. Also, we need to investigate the cases where recent trends are more important than older ones; Sometimes this might be important, especially given business cycle patterns (recession and inflation).

Table 3 exhibits the hyperparameters' statistical significance of five leading stocks in the TSE. The hyperparameters in the table are going to be labeled: Batch size, No. Neurons (number of neurons), Stopping Criterion (the maximum number of non-improvements in the solution till convergence), No. lags (number of previous periods to the current included as feature(s) in LSTM), No. of hidden Layers (number of hidden layers), Size of the period (number of trading days included in the training/evaluation data), and Ratio (the ratio of training to evaluation data). The main idea behind the random-grid search is to populate the said parameters with random values. We populate a minimum of 200 random values for each parameter. To demonstrate the high variability in the results, Table 3 shows the statistically significant effect each parameter has on the prediction error. A checkmark in the table translates to a significant statistical relationship of a p-value less than 0.05. While an 'X' signifies a p-value bigger than 0.05. The only parameter that brings a significant effect in all four stocks is the size of the period.

4.2. Genetic hyperparameter optimization and portfolio trading

Genetic-based algorithms are adaptive to a variety of applications. Notably, they are suited to black-box models such as deep learning. We utilize the genetic algorithm (GA) introduced in section 3 to optimize the hyperparameters. The number of generations is set to 4 (higher values did not bring significantly improved outcomes in preliminary runs). All other parameters are listed in Table 4, where distinct ranges are tried preliminarily. While the lower and upper limits (UL and LL) are fixed for the random assignment of our factors (hyperparameters).

Table 5 features the results of five different stocks when the genetic algorithm is used. We see impressive improvements. In comparison to the grid-random-search, the genetic algorithm improves the results by almost 40% (see Table 6). On average, the genetic algorithm brings a percent prediction error of 1.6% lower than the Grid-search. It is vital to restate that all stocks studied in this work are traded in the TSE.

The training ratio we use is 80% training and 20% evaluation where LSTM is fed the training sets and then training is gaged (i.e., tests the training) using the evaluation data. Critical to note, test data in this work will refer to hidden data: Data the LSTM never sees. Hence, overall the genetic algorithm does improve prediction accuracy when coupled with LSTM while eliminating any data leakage. The later sections will shine the light on Deep RankNet and its performance capability concerning LSTM and concerning the value of k (number of stocks in a portfolio).

4.3. Deep RankNet

The main contribution of the work is that we implement a genetic search for hyperparameter optimization and Deep RankNet. To the best of our knowledge, there is not a similar work in the literature. Also, the work contests the number of stocks chosen within a given portfolio,

Table 3
Statistical Significance of factors (less than 0.05p-value).

Stock	Batch Size	No. Neurons	Stopping Criterion	No. lags	No. of hidden Layers	Size of the Period	Ratio
TRP	X	X	X	X	X	√	√
EXE	X	X	X	X	√	√	X
QSR	X	X	√	X	√	√	X
SMC	X	X	√	X	X	√	X

Table 4
Genetic algorithm parameter generations.

Genetic algorithm parameters
Generations = 4 (Number of generations in genetic algorithm)
Population size = 48 (Number of solutions)
mutation rate = 0.15
Batch size UL = 50 Batch size LL = 12
Neurons UL = 70 Neurons LL = 22
Stopping Criterion UL = 24 Stopping Criterion LL = 11
Number of lags UL = 16Number of lags LL = 1
Number of hidden UL = 16Number of hidden LL = 1
Number of periods UL = 259Number of periods LL = 55
UL: Upper Limit, LL = Lower Limit

while for example the work of Li and Tan (2021) only contests the k_{10} (k = 10) model. This is quite important for small traders who would want to utilize a simpler and less capital-intensive investment strategy. Largely, we start with a comparison between LSTM and Deep RankNet model, then we look at the performance of Deep RankNet models in high volatility periods, and finally, we evaluate the overall performance of different Deep RankNet models.

4.3.1. Deep RankNet and LSTM

To evaluate the performance of the Deep RankNet model, a rolling time window expanding from 2015/11/6 to 2021/11/27 is employed. The time horizon is subdivided into equally spaced time windows having 38 out of sample training/evaluations data and a 2-day holding period. The width of the time window is set higher than the work of (Li & Tan, 2021) since we are using the optimal hyperparameter output directives from the genetic algorithm. For each time window, the best ten *LSTM* performing stocks (highest predicted returns) are lumped into a portfolio for trading while the top ten ranked stocks output from the Deep RankNet model are lumped in another. Fig. 5 illustrates the superiority in performance for the Deep RankNet model. The figure presents the mean readings for 30 non-overlapping periods of 30 days each stretching from 2015 to 11-6 to 2020-01-09.

Table 7 illustrates the advantage of Deep RankNet 10, where both the average and median of returns are higher for the RankNet model. While the t-statistic exhibit a higher signal-to-noise ratio for the Deep RankNet signaling better accuracy at lesser variability. In retrospect to the Risk Metrics approach (Mina & Xiao, 2001), we evaluate the historic tail risk of the portfolios. Based on the historic 1% and 5% VAR (value at risk), the extent of possible financial losses is significantly less for the RankNet model. Sequentially, the Sharpe ratio for the Deep RankNet is significantly higher for the RankNet model. Hence, the Deep RankNet brings superior excess return per unit of risk. While the Sortino ratio indicates smaller downside losses in the RankNet model. The skewness of the returns designates a positive tendency which is an encouraging attribute for investors (Krauss, Do, & Huck, 2017). Alternatively, using the quartiles and the two extremities of the returns, it can be concluded that the overall returns exhibit a leptokurtic behavior, which classically features large outliers. This is quite typical and can be seen in other works (Krauss et al., 2017).

Moving forward, the one substantial advantage of Deep RankNet is its simplicity, efficacity, and speed. The Deep RankNet is computationally efficient since we rank all stocks all at once (so learning is more global), while LSTM learns the behavior of each stock independently. For instance, in this study, 100 stocks are studied. LSTM requires a lot of computational effort. For small investors and even medium-size investing firms, this can be quite impeding.

4.3.2. Performance deep RankNet models for COVID 19 pandemic period

The period starting from November 2019 and ending in November 2021 is picked in the study to gauge the performance of the Deep RankNet models in volatile circumstances (COVID Pandemic, Table 8). Here, we aim to show the agility and robustness of the models in

Table 5
Percent Average Prediction Errors for five stocks.

Stock	Average Prediction Error (%)	Batch Size	No. Neurons	Stopping Criterion	No. Lags	No. Hidden Layers	Size of the period
SMC	2.60	44	40	20	2	6	131
EXE	0.70	19	53	14	2	1	255
ZZZ	2.34	38	57	12	13	3	259
TRP	2.67	12	34	18	7	7	176
KLS	3.51	28	68	14	7	9	152

Table 6
Comparison between Genetic and Grid-search (*batch size, followed by neurons, stopping criterion, number of lags, and number of hidden layers last).

Stock	Hyper Parameters*	Genetic (Absolute Percent Prediction Error)	Grid-Random (Absolute Percent Prediction Error)
SMC	48. 66. 23. 13. 4.	2.471723	3.346693
EXE	34. 57. 14. 5. 8.	2.229095	1.720312
ZZZ	32. 47. 19. 9. 5.	1.964031	5.203928
TRP	34. 52. 17. 6. 9.	0.514455	1.494121
KLS	20. 67. 17. 13. 5.	3.086602	2.793172
KEL	44. 65. 24. 10. 10.	3.693114	11.03511
EQB	33. 49. 21. 8. 1.	2.010776	5.568764
ERO	15. 62. 15. 5. 2.	2.622884	2.831305
MUX	30. 62. 13. 3. 12.	0.780949	3.83661
NCU	50. 43. 12. 4. 11.	3.252497	3.798446
TCS	42. 25. 16. 12. 4.	3.426823	2.03306

exceptionally volatile epochs. We use a rolling time window of 20 days training/learning period and two days of holding, as per the work of Li and Tan (2021). However, this work furthers the research to evaluate the performance of different k models. In Table 8, on average we see the Deep RankNet3 ($k = 3$) model outperforming the market and other models. The Kurtosis is quite high for the market as well as the models. This is due to the strong effect of strong outliers. It is important to note that the market behavior of the TSE (last column in the table: Market) is quite hindered by the COVID 19 pandemic due to the commodity-centric nature of the Canadian economy. The COVID 19 pandemic has hit the oil market severely and thus negatively affected the performance of many financial assets in the TSE. Conversely, all Deep RankNet models outperform the market. Digging deeper into the results, the RankNet10 exhibits less variation highlighted by a low standard deviation and standard error.

The Skewness indicator shows similar values to those of the research of Li and Tan (2021). All ranking models display positive skewness which is a positive attribute for investors and traders. Historical 1% VAR (Value at Risk) fluctuates a little bit higher than that of the market. Here we see the RankNet10 model bringing the lowest VAR among all the models. For the 5% VAR, again the RankNet10 brings the lowest risk, however, the risk now deviates from the market value. We see a similar tendency in the work of Li and Tan (2021). The Sharpe ratio, which is expressed as the excess return per unit of risk, quantified in standard deviations, records the highest value for the Deep RankNet3 model where investors can expect the highest reward for the incremental risk they undertake. In all models, we see a Sharpe ratio much higher than the market, which means the average return brought by the Deep RankNet models is significant per unit of volatility or total risk. The Sortino

Table 7
Return characteristics of LSTM and Deep RankNet concerning Market.

	LSTM	Deep Rank 10	Market
Mean	−0.02765	0.001241	0.001031
Standard Deviation	0.029888	0.027594	0.011646
t-statisitic	−0.9252	0.044975	0.08856
Quartile 1	0.04327	−0.01324	−0.00485
Median	−0.0221	0.001183	0.000881
Quartile 3	−0.01363	0.015397	0.007514
Standard Error	0.003004	0.003004	0.00117
1% VAR	−0.10474	−0.05423	−0.02969
5% VAR	−0.0797	−0.03215	−0.01785
Kurtosis	2.67977	8.02478	1.90059
Skewness	0.08246	1.27012	−0.03022
Sharpe ratio	−10.3441	0.502837	0.99013
Sortino ratio	12.7633	0.838235	1.42211

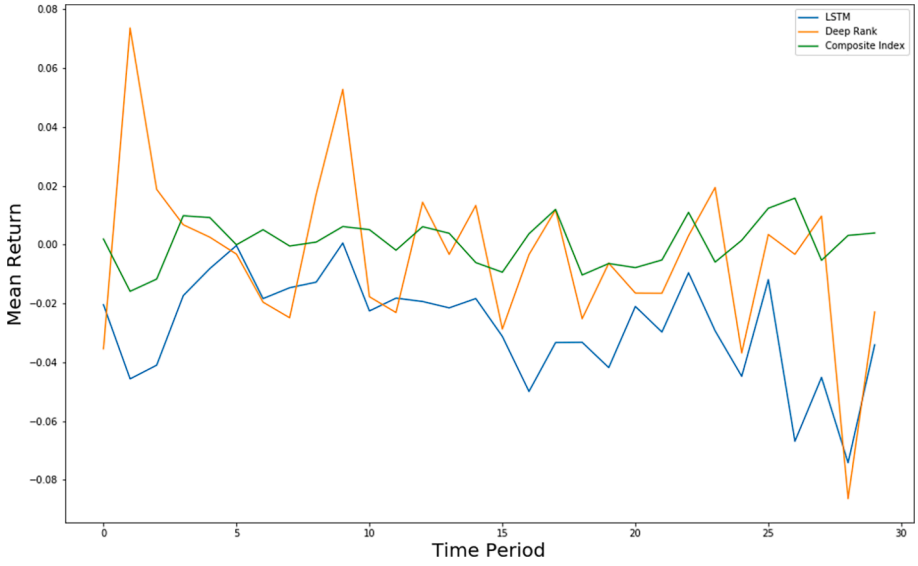


Fig. 5. Deep RankNet 10 model performance compared to LSTM and Market.

Table 8
Return characteristics during a high volatility period (COVID pandemic).

	Deep Rank 3	Deep Rank 5	Deep Rank 10	Market
Mean	0.0112042	0.0098324	0.0054097	−0.0002317
Standard Deviation	0.0910619	0.0806379	0.0535071	0.0281870
Quartile 1	−0.0273160	−0.0271612	−0.0016667	−0.0039364
Median	−0.0045247	−0.0017371	0.0037266	0.0014921
Quartile 3	0.0394800	0.0316374	0.0222270	0.0076225
Standard Error	0.0081776	0.0072415	0.0072415	0.0025313
99% VAR	−0.1807110	−0.1819110	−0.1270970	−0.1116900
95% VAR	−0.0919412	−0.0721400	−0.0641543	−0.0284244
Kurtosis	10.5848000	7.4565300	5.4772200	20.3888000
Skewness	2.2164200	1.8040500	1.2463600	−0.6213600
Sharpe ratio	1.3756300	1.3632400	1.3632400	−0.0919156
Sortino ratio	2.8100900	2.6491900	1.8817900	−0.0910433

Table 9
Revenue characteristics for more stable periods.

	Deep Rank 3	Deep Rank 5	Deep Rank 10	Market
Mean	0.0032632	0.0026494	0.0020388	0.0012043
Standard Deviation	0.0249442	0.0194135	0.0143223	0.0088033
t-statistic	0.1308200	0.1364700	0.1423540	0.1368000
Quartile 1	−0.0097100	−0.0078558	−0.0050192	−0.0034425
Median	0.0031545	0.0028589	0.0026322	0.0015514
Quartile 3	0.0175028	0.0157235	0.0116242	0.0066549
Standard Error	0.0014426	0.0011227	0.0011227	0.0005091
1% VAR	−0.0637852	−0.0490752	−0.0350077	−0.0256849
5% VAR	−0.0377661	−0.0326335	−0.0243798	−0.0140409
Kurtosis	1.8133600	1.2322700	1.7075500	1.4110000
Skewness	−0.3293560	−0.3330060	−0.7580100	−0.5348600
Sharpe ratio	1.4626100	1.5257800	1.5915700	1.5294700
Sortino ratio	2.0188700	2.1710400	2.0806800	2.0429600

ratio, which scales the returns by their downside deviation (Krauss et al., 2017) shows a higher value for the Deep RankNet3 model. So the risk of downside losses is somewhat mitigated. Overall, the models seem to behave well under volatile conditions and ultimately outperform the market. The annual returns averaged for all the models, are 113% better than the market, before transaction costs. The RankNet3 model brings an annual excess return of 143% to that of the market. The absolute annual sum of returns after discounting the transaction cost (0.16% per trade) is 120%. While the more risk-averse RankNet10 brings annual excess returns of 50% after discounting the transaction costs and 70% excess annual return to the market.

4.3.3. Overall performance of Deep RankNet models

Now, we study longer and more stable periods. Running 35 sub-periods between 2015 and 2021, the RankNet3 brings the highest mean return at the cost of higher variation. Fig. 6 illustrates the better performance of the RankNet3 model, but to have a more holistic picture, we need to dig deeper into the analytics.

Table 9 highlights important analytics. The t-statistic tends to be the highest for the RankNet10. The median value for returns is higher for RankNet3. However, the 1% and 5% historical VARs are higher for the RankNet3 model. This is quite understandable since picking a smaller number of stocks (less diversification) tends to bring higher risk. The RankNet10 model brings the lowest VAR values of all the deep rank models. The Sharpe ratio enforces this further as it is higher for the RankNet10 as it bears substantial gains for the risk ventured. Since we are no longer looking to extraordinary periods, the Market does perform better but is still inferior to the RankNet models. The RankNet3 model brings 26% excess annual returns to that of the market. Discounting the transaction costs, the RankNet3 brings around 21% annual returns. It is important to note, that the drop in marginal returns of RankNet models is due to the better performance of the overall market in the longer period of the study.

4.3.4. Analysis

In analysis, we can see some important insights in terms of hyper-parameter optimization, feature engineering, the performance of the Deep RankNet in comparison to a more traditional non-rank deep learning algorithm, the impact of COVID on the performance of the algorithms at discussion, and the performance of different *k-stock* strategies for Deep RankNet. To start, we perform hyperparameter optimization using a grid search and then using a genetic algorithm. The genetic algorithm improves the results by almost 40%. Hence we tune all the discussed hyperparameters using a genetic-based algorithm. Concerning features, we see the length (size) of the training period to be the most important feature with respect to performance. When comparing the Deep RankNet to LSTM, the Deep RankNet brings better and higher daily returns (Deep RankNet brings a daily returns average of 0.4% higher than LSTM). When comparing different RankNet strategies ($k = 3$, $k = 5$, and $k = 10$), we start the analysis for the highly volatile COVID19 period. Here all RankNet strategies excel and bring high returns. At the cusp of the COVID19 pandemic, we see all techniques performing better than the market, which is in line with the works of Bogomolov (2013), Do and Faff (2010), and Huck (2010) where they

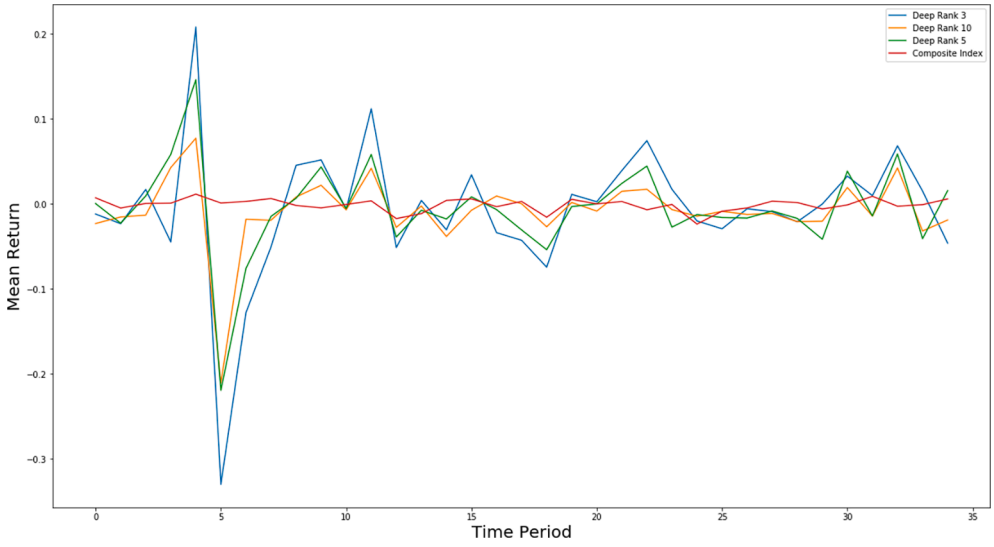


Fig. 6. Performance of different RankNet models concerning the market.

have shown the effectiveness of trading strategies in periods of high turmoil. During longer periods, and hence more stable conditions, we see the highest returns to risk coming from the RankNet10 model. The model brings the highest Sharpe Ratio of 1.59, which is a very positive indicator for investors. Among the three ranking models, we then observe the impact of selecting fewer stocks for a portfolio. We find that lowering the number of stocks, brings higher returns but at the expense of risk. The RankNet3 (k = 3) brings the highest daily return of 0.33% (0.17% after discounting transaction cost of 0.16%). However, it also brings the lowest Sharpe Ratio which translates into smaller gains for the risk undertaken compared to RankNet10. Turning back to prediction imporvement, we see improvements that can be seen in transform-based models as in the work Dezhkam and Manzuri (2023) which had shown 99.8% better results when using a transform-model than forming the portfolio using raw financial. This especially true during periods of high volatililty.

In our study, we utilized several technical indicators to analyze the stock market trends and predict future prices. However, upon analyzing our results, we did not find strong evidence that including these technical indicators significantly improved the accuracy of our predictions. While some technical indicators may provide valuable insights into market trends and patterns, their effectiveness in predicting stock prices can vary depending on a range of factors, including market conditions, data quality, and the choice of indicators used. The number of stocks in a portfolio has been studied previously (Krauss et al., 2017), but our contribution is the use of the Deep Rank Matching algorithm coupled with hyperparameter optimization. Importantly, we emphasize that the use of a lower number of stocks per trade per portfolio may have significant implications for small traders, where fixed costs may apply for each stock trade. Furthermore, we have looked at a variety of K values (3, 5, and 10).

4.3.5. *Practical implications*

There are many challenges to investors and decision-makers when it comes to investing in the stock market. The questions are many. Which portfolio to choose? What risk profile to undertake? Which market condition is favorable? What revenue is acceptable for a given risk exposure? This work helps answer these questions in an objective manner. We present an automated methodology for picking stocks with the option of three, five, and ten stocks per portfolio. We present a risk matrix for all the different options highlighting not only returns but risk exposure. Investors who are not willing to venture into high-risk investments can use this work to pick and choose. The work also supplies a baseline for the market given each investing strategy. Fig. 6 illustrates the movement of each strategy with respect to the composite index (market). While the work also illustrates the accuracy of the predictions during the COVID 19 period as an extreme case for the impact of market conditions on predictions. Moreover, the work highlights the return of each strategy with respect to the unit risk undertaken; this can assist investors on what level of risk to undertake.

Outside the world of financial trading, this work furthers the overall applicability of machine learning algorithms. Deep RankNet algorithms are still young and can influence many other fields. There are many examples of its potential usage but we would like to entertain two. First, in supply chain management, Deep RankNet can be used to rank components or parts. Multiple parts can be ranked with respect to a given criterion such as importance, price, value, reliability, etc. In manufacturing, plants source hundred of parts for a given function. These parts can be ranked, based on historic data given the criterion in question. Second, in the field of electric motors where fault diagnosis is an important task. Vitally, electric motors use about 68% of total generated electricity (Głowacz et al., 2021). A Deep RankNet algorithm can be used to classify different acoustic signals in response to fault diagnosis.

5. Conclusion

The objective of the work is to construct financial portfolios via the use of deep learning. The practical implication of this stretches from academics searching for more robust machine learning algorithms, to investors and decision-makers needing assistance in portfolio selection and risk analysis. The work also can benefit decision-makers in operations, and management in general, who can use the framework to better forecast the future demand of critical parts and products.

The main contributions of this work are the synthesis of multiple stocks and the effects of multiple periods using a novel approach. Importantly, the work includes portfolio selection to resemble practical managerial implications. The work also dives into multiple factors that affect the performance of deep learning algorithms. We've taken the time to explicitly discuss many of the factors in this work: Training ratio, Batchsize, Stopping criterion, period size, and others. The results indicate insignificant gains in optimizing batch size as a hyperparameter while the training/evaluation ratio appeared also important. More importantly, the length and/or stretch of the trading period brings the highest statistical importance of all the parameters studied. The overall integration of hyperparameter optimization tends to improve Deep Learning performance by 40% compared to classical random grid approach. The next contribution of work is the comparative study between the LSTM and Deep RankNet given feature engineering improvements. The Deep RankNet models prove to be superior to that of LSTM. While, when comparing the different RankNet strategies, the results show the benefit of smaller k portfolios in terms of returns but the advantage of larger k portfolios in the overall return to risk dynamic. Remarkably, all RankNet strategies exhibit superior performance during the cusp of the COVID 19 pandemic and brought higher than 100% excess returns to that of the market. To the best of our knowledge, this is the first work utilizing Deep RankNet models to study a Canadian financial market (market with high exposure to commodities). It is also the first to coalesce Deep RankNet, LSTM, and a genetic-based algorithm.

As per the case of any research, there are limitations. To start, we focus on the TSE which renders our findings unique to the market conditions. However, other studies on the S&P500 and Chinese stock markets have rendered similar results. This is not to say that our results are universal. We suggest for practitioners to use our methodology on other financial markets and evaluate the outcomes. In fact, this could be the subject of future research where researchers can compare the results coming from different financial markets. Second, there are a handful of factors/criteria that can be can be manipulated, or further investigated. For instance, ranking stocks based on their relationship to the median value of all assets can be altered. Future research can look at an ample variety of ranking criteria to improve performance. Criteria such as variability, trading volume, and/or textual information might render improvements. The holding of a given trading option shadows the work of Li and Tan (2021). It would be interesting to investigate the effect of changing the size of the holding period. This can be important since different holding strategies might render diverse profiles of returns, volatility, and risk. Finally, we would like to emphasize the importance of examining the variation of the number of stocks in each portfolio, which has been studied before, but our contribution lies in the use of the Deep Rank Matching algorithm coupled with hyperparameter optimization.

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 will be made available on request.

References

Acuna-García, J. A., Luz Canchola-Magdaleno, S., & Olmos Trejo, C. A. (2022). Stock Market Forecasting Using Continuous Wavelet Transform And Long Short-Term Memory Neural Networks. *International Journal of Advanced Research in Computer Science, 13*(6).

Akita, R., Yoshihara, A., Matsubara, T., & Uehara, K. (2016). Deep learning for stock prediction using numerical and textual information. In *In 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)* (pp. 1–6). IEEE.

Ayala, J., García-Torres, M., Noguera, J. L. V., Gómez-Vela, F., & Divina, F. (2021). Technical analysis strategy optimization using a machine learning approach in stock market indices. *Knowledge-Based Systems, 225*, Article 107119.

Basak, S., Kar, S., Saha, S., Khaidem, L., & Dey, S. R. (2019). Predicting the direction of stock market prices using tree-based classifiers. *The North American Journal of Economics and Finance, 47*, 552–567.

Bogomolov, T. (2013). Pairs trading based on statistical variability of the spread process. *Quantitative Finance, 13*(9), 1411–1430. https://doi.org/10.1080/14697688.2012.74 8934

Burges, C., Tal, S., Erin, R., Ari, L., Matt, D., Nicole, H., & Greg, H. (2005). Learning to Rank Using Gradient Descent. In *In Proceedings of the 22nd International Conference on Machine Learning* (pp. 89–96). New York: Association of Computing Machinery.

Chen, W., Zhang, H., Mehlawat, M. K., & Jia, L. (2020). Mean-variance portfolio optimization using machine learning-based stock price prediction. *Applied Soft Computing, 106943*.

Coqueret, G. (2022). Persistence in factor-based supervised learning models. *The Journal of Finance and Data Science, 8*, 12–34.

Cuomo, S., Gatta, F., Giampaolo, F., Iorio, C., & Piccialli, F. (2022). An unsupervised learning framework for marketneutral portfolio. *Expert Systems with Applications, 192*, Article 116308.

Dai, H. L., Liang, C. X., Dai, H. M., Huang, C. Y., & Adnan, R. M. (2022). An online portfolio strategy based on trend promote price tracing ensemble learning algorithm. *Knowledge-Based Systems, 239*, Article 107957.

Dessain, J. (2022). Machine learning models predicting returns: Why most popular performance metrics are misleading and proposal for an efficient metric. *Expert Systems with Applications, 199*, Article 116970.

Dezhkam, A., & Manzuri, M. T. (2023). Forecasting stock market for an efficient portfolio by combining XGBoost and Hilbert-Huang transform. *Engineering Applications of Artificial Intelligence, 118*, Article 105626.

Do, B., & Faff, R. (2010). Does simple pairs trading still work? *Financial Analysts Journal, 66*(4), 83–95.

Du, J. (2022). Mean–variance portfolio optimization with deep learning based-forecasts for cointegrated stocks. *Expert Systems with Applications, 201*, Article 117005.

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.

Głowacz, A., Tadeusiewicz, R., Legutko, S., Caesarendra, W., Irfan, M., Liu, H., … Xiang, J. (2021). Fault diagnosis of angle grinders and electric impact drills using acoustic signals. *Applied Acoustics, 179*, Article 108070.

Goldberg, D. (1989). *Genetic algorithm in search, optimization and machine learning.* Reading, MA: Addison-Wesley.

Guo, G., Rao, Y., Zhu, F., & Xu, F. (2020). Innovative deep matching algorithm for stock portfolio selection using deep stock profiles. *PLoS One1, 15*(11), e0241573.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation, 9*(8), 1735–1780.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. Ann Arbor: University of Michigan Press.

Huck, N. (2010). Pairs trading and outranking: The multi-step-ahead forecasting case. *European Journal of Operational Research, 207*(3), 1702–1716.

Ismail, M. S., Noorani, M. S. M., Ismail, M., Razak, F. A., & Alias, M. A. (2020). Predicting next day direction of stock price movement using machine learning methods with persistent homology: Evidence from Kuala Lumpur Stock Exchange. *Applied Soft Computing, 106422*.

Jensen, M. (1978). Some anomalous evidence regarding market efficiency. *Journal of Financial Economics, 6*(2–3), 95–101.

Kanwal, A., Lau, M. F., Ng, S. P., Sim, K. Y., & Chandrasekaran, S. (2022). BiCuDNNLSTM-1dCNN—A hybrid deep learning-based predictive model for stock price prediction. *Expert Systems with Applications, 202*, Article 117123.

Kim, S., Ku, S., Chang, W., & Song, J. W. (2020). Predicting the Direction of US Stock Prices Using Effective Transfer Entropy and Machine Learning Techniques. *IEEE Access, 8*, 111660–111682.

Kirkpatrick C., Dahlquist J. (2011). Technical Analysis, The Complete Resource For Financial Market Technicians, Pearson Education Inc.

Krauss, C., Do, X. A., & Huck, N. (2017). Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. *European Journal of Operational Research, 259*(2), 689–702.

Kumar, I., Dogra, K., Utreja, C., & Yadav, P. (2018). A comparative study of supervised machine learning algorithms for stock market trend prediction. In *In 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT)* (pp. 1003–1007). IEEE.

Leippold, M., Wang, Q., & Zhou, W. (2022). Machine learning in the Chinese stock market. *Journal of Financial Economics, 145*(2), 64–82.

Li, Y., Fu, K., Zhao, Y., & Yang, C. (2022b). How to make machine select stocks like fund managers? Use scoring and screening model. *Expert Systems with Applications, 196*, Article 116629.

Li, Y., & Tan, Z. (2021). Stock Portfolio Selection with Deep RankNet. *The Journal of Financial Data Science, 3*(3), 108–120.

Li, Y., Zheng, X., Chen, C., Wang, J., & Xu, S. (2022a). Exponential Gradient with Momentum for Online Portfolio Selection. *Expert Systems with Applications, 187*, Article 115889.

Lin, Y. C., Chen, C. T., Sang, C. Y., & Huang, S. H. (2022). Multiagent-based deep reinforcement learning for risk-shifting portfolio management. *Applied Soft Computing, 123*, Article 108894.

Liu, Y. C., & Yeh, I. C. (2017). Using mixture design and neural networks to build stock selection decision support systems. *Neural Computing and Applications, 28*(3), 521–535.

Long, W., Lu, Z., & Cui, L. (2019). Deep learning-based feature engineering for stock price movement prediction. *Knowledge-Based Systems, 164*, 163–173.

Ma, Y., Han, R., & Wang, W. (2021). Portfolio optimization with return prediction using deep learning and machine learning. *Expert Systems with Applications, 165*, Article 113973.

Ma, T., & Tan, Y. (2022). Stock Ranking with Multi-Task Learning. *Expert Systems with Applications, 199*, Article 116886.

Malkiel, B. G. (1973). *A Random Walk Down Wall Street*. New York: Norton.

Mehtab S., Sen J., Dutta A. (2020). Stock price prediction using machine learning and LSTM-based deep learning models. arXiv preprint arXiv:2009.10819.

Mina, J., & Xiao, J. Y. (2001). Return to RiskMetrics: The evolution of a standard. *RiskMetrics Group, 1*, 1–11.

Nabipour, M., Nayyeri, P., Jabani, H., Shahab, S., & Mosavi, A. (2020). Predicting Stock Market Trends Using Machine Learning and Deep Learning Algorithms Via Continuous and Binary Data; a Comparative Analysis. *IEEE Access, 8*, 150199–150212.

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

Paiva, F. D., Cardoso, R. T. N., Hanaoka, G. P., & Duarte, W. M. (2019). Decision-making for financial trading: A fusion approach of machine learning and portfolio selection. *Expert Systems with Applications, 115*, 635–655.

Patel, J., Shah, S., Thakkar, P., & Kotecha, K. (2015). Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert systems with applications, 42*(1), 259–268.

Pinelis, M., & Ruppert, D. (2022). Machine learning portfolio allocation. *The Journal of Finance and Data Science, 8*, 35–54.

Porshnev, A., Redkin, I., & Shevchenko, A. (2013). Machine learning in the prediction of stock market indicators based on historical data and data from twitter sentiment analysis. In *In 2013 IEEE 13th International Conference on Data Mining Workshops* (pp. 440–444). IEEE.

Reddy, V. K. S. (2018). Stock market prediction using machine learning. *International Research Journal of Engineering and Technology, 5*(10).

Ren, X., Xu, W., & Duan, K. (2022). Fourier transform based LSTM stock prediction model under oil shocks. *Quantitative Finance and Economics, 6*(2), 342–358.

Rousis P., Papathanasiou S. (2018). Is Technical Analysis Profitable on Athens Stock Exchange? Mega Journal of Business Research. 2018.

Shah, V. H. (2007). Machine learning techniques for stock prediction. *Foundations of Machine Learning, Spring, 1*(1), 6–12.

Sharda, R., Delen, D., & Turban, E. (2020). Analytics. *Data Science, & Artificial Intelligence. Pearson.*

Sharma, M., & Shekhawat, H. S. (2022). Portfolio optimization and return prediction by integrating modified deep belief network and recurrent neural network. *Knowledge-Based Systems, 109024*.

Shen, S., Jiang, H., & Zhang, T. (2012). *Stock market forecasting using machine learning algorithms* (pp. 1–5). Stanford, CA: Department of Electrical Engineering, Stanford University.

Shi, S., Li, J., Li, G., Pan, P., Chen, Q., & Sun, Q. (2022). GPM: A graph convolutional network based reinforcement learning framework for portfolio management. *Neurocomputing, 498*, 14–27.

Singh, R., & Srivastava, S. (2017). Stock prediction using deep learning. *Multimedia Tools and Applications, 76*(18), 18569–18584.

Usmani, M., Adil, S. H., Raza, K., & Ali, S. S. A. (2016). Stock market prediction using machine learning techniques. In *In 2016 3rd international conference on computer and information sciences (ICCOINS)* (pp. 322–327). IEEE.

Vazirani, S., Sharma, A., & Sharma, P. (2020). Analysis of various machine learning algorithm and hybrid model for stock market prediction using python. In *In 2020 International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE)* (pp. 203–207). IEEE.

Vijh, M., Chandola, D., Tikkiwal, V. A., & Kumar, A. (2020). Stock Closing Price Prediction using Machine Learning Techniques. *Procedia Computer Science, 167*, 599–606.

Wu, D., Wang, X., & Wu, S. (2022). A Hybrid Framework Based on Extreme Learning Machine, Discrete Wavelet Transform, and Autoencoder with Feature Penalty for Stock Prediction. *Expert Systems with Applications, 118006*.

Yadav, A., Jha, C. K., & Sharan, A. (2020). Optimizing LSTM for time series prediction in Indian stock market. *Procedia Computer Science, 167*, 2091–2100.

Zhao, D., Bai, L., Fang, Y., & Wang, S. (2022). Multi-period portfolio selection with investor views based on scenario tree. *Applied Mathematics and Computation, 418*, Article 126813.