# Threshold-Based Portfolio: The Role of the Threshold and Its Applications

Sang Il Lee · Seong Joon Yoo

Received: date / Accepted: date

Abstract This paper aims at developing a new method by which to build a data-driven portfolio featuring a target risk-return. We first present a comparative study of recurrent neural network models (RNNs), including a simple RNN, long short-term memory (LSTM), and gated recurrent unit (GRU) for selecting the best predictor to use in portfolio construction. The models are applied to the investment universe consisted of ten stocks in the S&P500. The experimental results shows that LSTM outperforms the others in terms of hit ratio of one-monthahead forecasts. We then build predictive threshold-based portfolios (TBPs) that are subsets of the universe satisfying given threshold criteria for the predicted returns. The TBPs are rebalanced monthly to restore equal weights to each security within the TBPs. We find that the risk and return profile of the realized TBP represents a monotonically increasing frontier on the risk-return plane, where the equally weighted portfolio (EWP) of all ten stocks plays a role in their lower bound. This shows the availability of TBPs in targeting specific risk-return levels, and an EWP based on all the assets plays a role in the reference portfolio of TBPs. In the process, thresholds play dominant roles in characterizing risk, return, and the prediction accuracy of the subset. The TBP is more data-driven in designing portfolio target risk and return than existing ones, in the sense that it requires no prior knowledge of finance such as financial assumptions, financial mathematics, or expert insights. In a practical application, we present the TBP management procedure for a time horizon extending over multiple time periods; we also discuss their application to mean-variance portfolios to reduce estimation risk.

Keywords Portfolio management  $\cdot$  Recurrent neural networks  $\cdot$  Heuristic efficient frontier  $\cdot$  Financial time series

Sang Il Lee

Seong Joon Yoo

Department of Computer Engineering, Sejong University, Seoul, 05006, Republic of Korea E-mail: silee@sejong.ac.kr

Department of Computer Engineering, Sejong University, Seoul, 05006, Republic of Korea E-mail: sjyoo@sejong.ac.kr

# **1** Introduction

Today, machine learning has come to play an integral role in many parts of the financial ecosystem, from portfolio management and algorithmic trading, to fraud detection and loan/insurance underwriting. Time series are one of the most common data types encountered in finance, and so time-series analysis is one of the most widely used traditional approaches in finance and economics. The development of machine learning algorithms has opened a new vista for modeling the complexity of financial time series as an alternative to the traditional econometric models, by effectively combining diverse data and capturing nonlinear behavior. For this reason, financial time-series modeling has been one of the most interesting topics that has arisen in the application of machine learning to finance. Researchers have successfully modeled financial time series by focusing primarily on prediction accuracy or automatic trading rules [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]. Nevertheless, financial modeling and applications remain daunting, given the difficulties arising from the essentially nonlinear, complex, and evolutionary characteristics of the financial market.

On the other hand, asset allocation has been traditionally considered an issue central to investment and risk management. Markowitz [17] was the first to introduce a rigorous mathematical framework for allocation, called modern portfolio theory (MPT). Based on a mean–variance optimization technique, MPT provides a method by which to assemble assets and maximize the expected return of the portfolio for a given level of risk. Following Markowitz's thinking, new portfolio models have been subsequently proposed for more practical use and to achieve a better understanding of portfolios. Examples include the thee-factor asset pricing model [18], the Black–Litterman model[19], the resampled efficient frontier model [20], the global minimum variance model [21], the maximum diversification portfolio [22], and the risk-parity portfolio [23, 24]. Additionally, dynamic/tactical asset allocations based on simple rules or market anomalies were developed to automatically adjust portfolios in response to market changes [25, 26, 27, 28]. These studies show that today there is general consensus about the importance of effective combinations of assets.

In this study, we propose a new method for constructing a data-driven portfolio using recurrent neural networks (RNNs)-based future return predictions. Throughout this study, we will refer to this portfolio as a threshold-based portfolio (TBP), since its properties are characterized by the threshold levels imposed on predicted returns. In particular, this study makes the following main contributions to the literature:

- It examines the ability of RNNs to forecast one-month-ahead stock returns.
- It develops a new TBP portfolio method and analyzes their properties.
  - The threshold can be used for a parameter to draw a TBP frontier that comprises the set of TBP points on a risk-return plane. This implies that one can build a portfolio with a specific risk-return level by selecting the appropriate threshold level.
  - The equally weighted portfolio (EWP) is the lower bound of the TBPs on the TBP frontier. This implies that the TBPs can be characterized with respect to a reference portfolio, EW.

• In practical application, it develops the management process for TBPs for pursuing specific risk-return levels over multiple-periods and for the method incorporating TBPs into MPT.

The remainder of this paper is organized as follows: Section 2 discusses some of the important work related to this area. Section 3 explains the simple recurrent neural network (S-RNN), long short-term memory (LSTM), and gated recurrent units (GRU). Section 4 provides experimental results regarding the prediction accuracy of the models and the performance of TBPs. In Section 5, we discuss the practical applications of TBPs. Finally, in Section 6, we conclude this paper and discuss possible future extensions of our work.

# 2 Related Work

We present LSTM-based predictions and prediction-based portfolios.

#### 2.1 Financial Time Series Prediction Using RNNs

Using conventional econometric models, financial economists have found there to be statistically significant relationship between stock returns and lagged variables. For example, Campbell et al. [29] investigated the relationship between aggregate stock market trading volume and the serial correlation of daily stock returns. They provide an evidence that a stock price decline on a high-volume day is more likely than a stock price decline on a low-volume day to be associated with an increase in the expected stock return. Choueifaty and Coignard [30] show that trading volume is a significant determinant of the lead-lag patterns observed in stock returns.

For this reason, we select RNN algorithms; these are superior for modeling time-lag effects in multi-dimensional financial time series, by virtue of feeding the network activations from a previous time step as inputs into the network, to influence predictions in the current step. In contrast, feed forward neural networks (FFNNs) are not appropriate for capturing these time-dependent dynamics: they operate on a fixed-size time windows, and so they can provide only limited temporal modeling.

RNNs are less commonly applied to financial time-series predictions, yet some recent studies has shown promising results for use in financial time-series prediction. Fischer and Krauss [13] deployed LSTM networks to predict one-day-ahead directional movements in a stock universe and constructed subset portfolios by selecting constituents outperforming the cross-sectional median return of all stocks in the next day. They found that LSTM networks outperform memory-free classification methods (i.e., a random forest (RAF), a deep neural net (DNN), and a logistic regression classifier (LOG)) on measures of the mean return per day, annualized standard deviation, annualized Sharpe ratio, and accuracy.

More recently, Bao et al. [31] developed a hybrid model called the WSAEs– LSTM, combined with wavelet transforms (WT), stacked autoencoders (SAEs), and LSTM to effectively combine diverse financial data, including historical trading data of open price, high price, low price, closing price, and volume and technical indicators of stock market indexes and macroeconomic variables. The experimental results show that it produces more accurate one-day-ahead stock price predictions than the similar models, including WLSTM (i.e., a combination of WT and LSTM), RNN, and LSTM.

## 2.2 Machine Learning Prediction-Based Investment Portfolios

Machine learning has been applied to portfolio construction while focusing on the portfolio optimization problem, with multiple objective functions being subject to a set of constraints. Machine learning-based prediction is a valuable tool that can be used to mitigate difficulties inherent in traditional methods, i.e., ranking stocks and assessing their future potential.

Freitas et al. [32] present a new model of prediction-based portfolio optimization for capturing short-term investment opportunities. For the universe of Brazilian stocks, they combined their neural network predictors featuring normal prediction errors with the mean-variance portfolio model, and show that the resulting portfolio outperforms the mean-variance model and beats the market index. More recently, Mishra et al. [33] developed a novel prediction based mean-variance (PBMV) model, as an alternative to the conventional Markowitz mean-variance model, to solve the constrained portfolio optimization problem. They present a lowcomplexity heuristic functional link artificial neural network (HFLANN) model to overcome the incorrect estimation taken as the mean of the past returns in the Markowitz mean-variance model, and carry out the portfolio optimization task by using multi-objective evolutionary algorithms (MOEAs). Ganeshapillai et al. [34] propose a machine learning-based method to build a connectedness matrix and address the shortcomings of correlation in capturing events such as large losses. They show that the matrix can be used to build portfolios that not only "beat the market," but also outperform optimal (i.e., minimum-variance) portfolios.

The results of these studies show that machine learning-based estimations can be effectively used to overcome certain limitations inherent in traditional method. With regard to prediction-based portfolios, our work is in line with this thinking, but is more fundamental in the sense that we focus heavily on predicted returns by imposing thresholds with respect to prediction and diversification effects, by aggregating stocks rather than adopting existing portfolio frameworks. This fact makes TVP more data-driven than existing models.

# 3 Models: S-RNN, LSTM, GRU

S-RNNs [35] are an extension of a conventional FFNN that adds a feedback connection to a feedback network consisting of three layers: an input layer, a hidden layer, and an output layer. However, [36] found it is difficult to train an S-RNN to capture long-term dependencies, because the gradients tend to either vanish or explode. Alternatively, LSTM [37] and GRU [38] have been proposed to overcome the problem by using a "gating" approach. The LSTM algorithm is local in space and time [37], which means that the computational complexity of learning LSTM models per weight and time step with the stochastic gradient descent (SGD) optimization technique is O(1), and the learning computational complexity per time step is O(W), where W is the number of weights. Hence, our model is capable of handling large-scale data, as the computational complexity of our model grows linearly with respect to the length of the input data.

# 3.1 LSTM Architecture

LSTMs can effectively learn important pieces of information that may be found at different positions in the financial time series, by controlling what is added and removed from memory in the hidden layers. This is conducted by using a combination of three gates: (1) a forget gate, (2) an input gate, and (3) an output gate.

**Forget Gate:** An LSTM cell receives the current input  $x_t \in \mathbb{R}^d$ , the hidden state vectors  $\mathbf{h}_{t-1} \in \mathbb{R}^n$ , and a cell state  $C_{t-1} \in \mathbb{R}^n$  at time t-1. The forget gate then is then calculated as

$$f_t = \sigma(W_f \mathbf{x}_t + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{b}_f), \qquad (1)$$

where:

- $\mathbf{W}_f \in \mathbb{R}^{n \times d}$  is the weight matrix from the input  $\mathbf{x}_t$  to the forget gate  $\mathbf{f}_t$ ,
- $-\mathbf{U}_{f} \in \mathbb{R}^{n \times n}$  is the weight matrix from the previous hidden vector  $\mathbf{h}_{t-1}$  to the forget gate  $\mathbf{f}_t$ ,
- $-\mathbf{b}_f \in \mathbb{R}^n$  is the forget gate bias,
- $-\mathbf{f}_t \in \mathbb{R}^n$  is the output of the gate, which determines the amount to be erased from the previous cell state, and
- $-\sigma(\cdot)$  is a sigma function.

**Input Gate:** The input gate  $\mathbf{i}_t$ , which is used to scale the candidate update vector  $\mathbf{C}_t \in \mathbb{R}^n$ , determines what parts of  $\mathbf{C}_t$  are added to the corresponding memory cell element at time t, based on the recurrent connection from the hidden vector  $\mathbf{h}_{t-1}$  and the input at time  $t, \mathbf{x}_t$ :

$$\mathbf{i}_t = \sigma(\mathbf{W}_i \mathbf{x}_t + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{b}_i), \tag{2}$$

$$\widetilde{\mathbf{C}}_t = tanh(\mathbf{W}_c \mathbf{x}_t + \mathbf{U}_c \mathbf{h}_{t-1} + \mathbf{b}_c), \tag{3}$$

where:

- $\mathbf{W}_i \in \mathbb{R}^{n \times d}, \mathbf{U}_i \in \mathbb{R}^{n \times n}, \text{ and } \mathbf{b}_i \in \mathbb{R}^n \text{ are the input gate parameters,}$  $\mathbf{W}_c \in \mathbb{R}^{n \times d}, \mathbf{U}_c \in \mathbb{R}^{n \times n},$
- $-\mathbf{b}_c \in \mathbb{R}^n$  are the parameters for selecting a candidate state,  $\widetilde{\mathbf{C}}_t$ , and
- $tanh(\cdot)$  is the tanh function.

Then, the current state of the cell  $\mathbf{C}_t \in \mathbb{R}^n$  is given by

$$\mathbf{C}_t = \mathbf{i}_t \odot \mathbf{C}_t + \mathbf{f}_t \odot \mathbf{C}_{t-1},\tag{4}$$

where  $\odot$  represents the element-wise Hadamard product.

**Output Gate:** The output gate  $\mathbf{o}_t \in \mathbb{R}^n$ , which is used to calculate the output  $\mathbf{h}_t \in \mathbb{R}^n$ , determines the output from the current cell state:

$$\mathbf{o}_t = \sigma(\mathbf{W}_o \mathbf{x}_t + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{b}_o), \tag{5}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot tanh(\mathbf{C}_t),\tag{6}$$

where  $\mathbf{W}_o \in \mathbb{R}^{n \times d}$ ,  $\mathbf{U}_o \in \mathbb{R}^{n \times n}$ , and  $\mathbf{b}_o \in \mathbb{R}^n$  are the output gate parameters. The hidden vector  $\mathbf{h}_t$  of the memory cell can be used as the final output of the network.

# 3.2 GRU Architecture

The structure of a GRU can be expressed as follows:

$$\mathbf{z}_t = \sigma(\mathbf{W}_z \mathbf{x}_t + \mathbf{U}_z \mathbf{h}_{t-1} + \mathbf{b}_z), \tag{7}$$

$$\mathbf{r}_{t} = \sigma(\mathbf{W}_{r}\mathbf{x}_{t} + \mathbf{U}_{r}\mathbf{h}_{t-1} + \mathbf{b}_{r}), \tag{8}$$

$$\mathbf{h}_{t} = \mathbf{z}_{t} \odot \mathbf{h}_{t-1} + (1 - \mathbf{z}_{t}) \odot \tanh[\mathbf{W}_{h}\mathbf{x}_{t} + \mathbf{U}_{h}(\mathbf{r}_{t} \odot \mathbf{h}_{t-1}) + \mathbf{b}_{h}], \qquad (9)$$

where:

- $-\mathbf{x}_t, \mathbf{h}_t, \mathbf{z}_t$ , and  $\mathbf{r}_t$  are the input vector, output vector, update gate vector, and reset gate vector, respectively, and
- W, U, and b are forward matrices, recurrent matrices, and biases, respectively.

### 4 Experiment

4.1 Data

#### 4.1.1 Universe

The asset universe consists of the top 10 stocks in Standard and Poor's 500 index (S&P500):

• Apple (AAPL), Amazon (AMZN), Bank of America Corporation (BAC), Berkshire Hathaway Inc. Class B (BRK-B), General Electric Company (GE), Johnson&Johnson (JNJ), JPMorgan Chase & Co. (JPM), Microsoft Corporation (MSFT), AT&T Inc. (T), and Wells Fargo & Company (WFC).

We use data from January 1997 to December 2016 from Yahoo Finance. The daily stock dataset contains five attributes: open price, high price, low price, adjust close price, and volume (OHLCV). Figure 1 graphically shows the normalized closed price (i.e., subtract the mean from each original value and then divide by the standard deviation). We convert the daily OHLCV dataset to four different monthly OHLCV datasets by calculating the last, mean, maximum, and minimum values of the daily OHLCV dataset per month. Each monthly OHLCV dataset is used as a raw dataset for forecasting one-month-ahead return at the end of each calendar month.

# 4.1.2 Preprocessing

To achieve higher quality and reliable predictions, the five attributes are preprocessed as a percentage change,  $(x^{(t)} - x^{(t-1)})/x^{(t-1)}$ . All data were divided into a training dataset (70%) to set the model parameters and the test set (30%) for an out-of-sample model evaluation. The 30% of the training set is used as the validation to evaluate a given model during training.



Fig. 1 (Color online) Normalized stock prices for the 10 sample stocks over the test period

The statistical characteristics of the data used to train and test the deep learning model are shown in Table 1. We observe that the data are roughly in the range of -1 to 1 which is a usual range of features in deep learning, save for abnormal trading volume from max values of the volume data.

Table 1	Statistics	of dat	a (closed	prices	and	volume)	used	$\operatorname{to}$	train,	validate,	and	$\operatorname{test}$	the
RNN mo	dels												

	AAPL	AMZN	BAC	BRK-B	GE	JNJ	JPM	MSFT	Т	WFC
					Train	ı data				
Mean	0.01, 0.45	$0.01 \ 0.12$	$0.01 \ 0.124$	$0.00 \ 0.21$	$0.00 \ 0.17$	$0.007 \ 0.186$	$0.00 \ 0.16$	-0.00 0.13	-0.00 0.20	$0.01 \ 0.13$
Std.	0.16, 2.22	$0.20 \ 0.59$	$0.08 \ 0.549$	$0.06 \ 0.79$	$0.07 \ 0.74$	0.069 0.901	$0.11 \ 0.83$	$0.12 \ 0.73$	$0.10 \ 0.84$	$0.07 \ 0.62$
Min.	-0.57, -0.88	-0.41 - 0.75	-0.22 -0.76	-0.12 - 0.76	-0.17 -0.617	-0.157 -0.66	-0.28 -0.80	-0.34 -0.80	-0.18 - 0.76	-0.16 -0.84
Max	0.45, 16.68	$0.62 \ 1.71$	$0.17 \ 2.198$	$0.26\ 2.34$	$0.19\ 2.93$	0.174 5.603	$0.25 \ 5.17$	$0.40 \ 4.84$	$0.29 \ 4.56$	0.23, 2.93
					Validat	ion data				
Mean	0.06, 0.31	-0.00 0.11	$0.01 \ 0.152$	$0.00 \ 0.13$	$0.00 \ 0.082$	$0.00 \ 0.14$	$0.01 \ 0.21$	$0.00 \ 0.32$	$0.015 \ 0.20$	0.01, 0.12
Std.	0.12, 1.06	$0.14 \ 0.60$	$0.03 \ 0.822$	$0.02 \ 0.52$	$0.02 \ 0.393$	$0.027 \ 0.68$	$0.04 \ 0.77$	$0.05 \ 1.66$	$0.03 \ 0.878$	0.02, 0.58
Min.	-0.15, -0.72	-0.30 -0.70	-0.04 -0.68	-0.05 -0.50	-0.065 - 0.48	-0.04 -0.65	-0.06 -0.59	-0.11 - 0.79	-0.06 -0.73	-0.02, -0.67
Max.	0.35, 5.03	$0.36 \ 1.96$	$0.09 \ 3.62$	$0.05 \ 1.30$	$0.058\ 1.19$	$0.05\ 2.90$	$0.09\ 2.85$	$0.08 \ 8.50$	$0.08 \ 4.30$	0.07, 1.41
					Test	data				
Mean	0.03, 0.11	$0.04 \ 0.36$	$-0.00\ 0.22$	$0.00 \ 0.27$	-0.011 0.11	$0.00 \ 0.10$	$0.00 \ 0.20$	$0.00 \ 0.15$	$0.00 \ 0.05$	0.00, 0.23
Std.	0.12, 0.54	$0.14 \ 1.40$	$0.21 \ 0.68$	$0.06 \ 1.06$	$0.11 \ 0.51$	$0.04 \ 0.51$	$0.11 \ 0.75$	$0.08 \ 0.70$	$0.06 \ 0.40$	0.14, 0.80
Min.	-0.32, -0.82	-0.25 - 0.72	-0.53 -0.60	-0.14 -0.86	-0.27 -0.58	-0.12 -0.61	-0.23 -0.73	-0.16 -0.69	-0.15 - 0.63	-0.35, -0.74
Max.	0.23, 1.71	$0.54 \ 6.45$	$0.73\ 2.25$	$0.12 \ 5.64$	$0.25\ 1.87$	$0.07 \ 1.49$	$0.24 \ 2.39$	$0.24 \ 3.05$	$0.09\ 1.50$	0.40, 3.65

#### 4.2 Experimental Design

We build S-RNN, LSTM, and GRU architectures for one-month-ahead forecasts of stock returns. Based on the validation set evaluation, we carried out a grid search over their hyperparameters over the number of RNN hidden layers (1,2, or 3) and the number of hidden units per layer (8, 16, 32, 64, or 128), and whether dropout is used to avoid overfitting of the model. The whole networks was trained by a backpropagation algorithm by minimizing the the quadratic loss value,  $L = \frac{1}{2} \sum_{t}^{T} (r(t) - \hat{r}(t))^2$  (where  $\hat{r}(t)$  is the output of the last layer and r(t) is the corresponding target) on the validation set. The efficient ADAM (adaptive moment estimation) optimization algorithm [39] with a learning rate of 0.001 is used to fit the models in mini-batches of size 20. From the experiments, we specified the topology consisting of an input layer, an RNN layer with h = 36 hidden neurons, a 50% dropout layer, and an output layer with a linear activation function for regression.

The feature vectors to feed the models are overlapping sequences of 36 consecutive points (trading months in three years) for the preprocessed variables. The sequences themselves are sliding windows shifted by one month for each time  $t \geq 36$ , that is,  $\{\mathbf{x}_{t-35}, \mathbf{x}_{t-34}, \dots, \mathbf{x}_t\}$ .

The experimental set-up is implemented over a laboratory prototype, equipped with an Intel quad core i7-6700 processor at 3.4GHz, Nvidia GPU (i.e., GTX 1070), and 32GB of RAM running the Ubuntu 16.04.2 LTS x86-64 Linux distribution. Prediction models are evaluated using Keras 2.0.4 [40] and TensorFlow 0.11.0. In our approach, the stage of modeling and forecasting stock returns contributes significantly to the overall processing time and, for one asset, is obtained in an approximate processing time of 109 seconds.

# 4.3 Prediction accuracy

We evaluate the predictive ability of the three models using the hit ratio, which is defined as follows:

Hit ratio = 
$$\frac{1}{N} \sum_{i=1}^{N} P_i(i=1,2,\dots,N),$$
 (10)

where N is total trading months and  $P_i$  is the prediction result for the  $i^{th}$  trading day, defined as:

$$P_i = \begin{cases} 1 & \text{if } y_{t+1} \cdot \hat{y}_{t+1} > 0, \\ 0 & \text{otherwise,} \end{cases}$$

where  $y_{t+1}$  and  $\hat{y}_{t+1}$  are the realized return at the last business day of month t+1and the one-month-ahead return predicted at the last business day of month t, respectively.

Table 2 shows the mean and standard deviation (SD) of the hit ratios for the 10 assets and the use of the last business day and the LSTM model generates the best prediction accuracy value (0.604). Therefore, we will use the LSTM model and the last business day OHLCV of each month for building TBPs in the next subsections.

Table 2 Mean and SD of hit ratios for the ten assets, respectively

	Last	Mean	Max	Min
S-RNN	0.559, <b>0.040</b>	0.555, 0.066	0.555,  0.059	0.555, 0.048
LSTM	<b>0.604</b> , 0.042	0.536,  0.083	0.569,  0.048	0.584,  0.039
GRU	0.573,  0.053	0.550,  0.079	0.586,  0.049	0.575,  0.051

# 4.4 Role of Threshold and TBP

We present the three type of TBP imposing the positive and negative threshold levels ( $\theta^+$  and  $\theta^-$ , respectively) on the predicted returns. Given the one-month-ahead return prediction  $\hat{r}_i$  for asset  $S_i(i = 1, 2, ..., n)$ , the TBPs are defined as the subset of the universe:

- Long only TBP:  $\{S_i \in \text{Universe} \mid \hat{r}_i \geq \theta^+\}$
- Short only TBP:  $\{S_i \in \text{Universe} \mid \hat{r}_i < -\hat{\theta}^-\}$
- Long-short TBP:  $\{S_i \in \text{Universe} \mid \hat{r}_i \geq \theta^+ \text{ and } \hat{r}_i < -\theta^-\}$

To illustrate, the thresholds are used to classify assets as long and short positions. A long (short) equity portfolio consists of assets whose prediction is higher (lower) than  $\theta^+$  ( $\theta^-$ ). Here, the thresholds are exogenous variables, and as explained in the next sections, we can determine proper threshold values for the target portfolio through backtesting.

# 4.5 Portfolio Weight

As classical portfolio models, the TBPs are built on the following underlying assumptions for evaluating performance. (i) all stocks are infinitely divisible; (ii) there are no restrictions on buying and selling any selected portfolio; (iii) there is no friction (transactions costs, taxation, commissions, liquidity, etc.); and (iv) it is possible to buy and sell stocks at closing prices at any time t.

We adapt the periodic rebalancing strategy: the investor adjusts the weights in his portfolio on the last business day of every month, as academic research typically assumes monthly portfolio rebalancing. Throughout this study, we provide the results of experiments on the long TBP (TBP in short), and other TVPs can be easily built by adjusting the thresholds. Let  $w_i$  denote the TBP weight on the  $i^{\text{th}}$  asset. The TBPs ( $w_i \ge 0$ ) are subjected to the budget constraint  $\sum_i^P w_i = 1$ , where P is the number of assets in the TBP. For all the TBPs,  $w_i$  is defined as  $|w_i| = 1/P$ , that is, equally-weighted TBPs.

#### 4.6 Simulation Results

# 4.6.1 Experiment 1: Performance of TBPs

Table 3 provides the mean and SD (standard deviation) values of the monthly returns of the individual assets, the EWP of the universe, and the TBPs with different thresholds. The explanation is as follows:

- AAPL and AMZN achieve higher returns with higher volatility.
- The EWP achieves lower volatility by diversification effect, which leads higher Sharpe ratio.
- The EWP and TBPs overly outperform individual stocks in terms of the Sharpe ratio.
- As shown in the TBPs, an increase in  $\theta$  results in an increase in the return and volatility of TBPs.

Asset/Portfolios	Threshold	Mean	$^{\mathrm{SD}}$	Mean/SD	Average assets
AAPL		0.02	0.075	0.271	
AMZN		0.023	0.08	0.299	
BAC		0.013	0.102	0.128	
BRK		0.01	0.0378	0.275	
GE		0.01	0.053	0.202	
JNJ		0.0129	0.036	0.353	
JPN		0.014	0.075	0.19	
MSFT		0.017	0.062	0.275	
Т		0.01	0.042	0.239	
WFC		0.011	0.048	0.237	
EWP		0.014	0.039	0.368	
	0	0.015	0.04	0.381	9.072
	0.005	0.017	0.044	0.381	6.637
TBP	0.01	0.02	0.052	0.383	4.376
1.01	0.015	0.02	0.06	0.347	2.855
	0.02	0.018	0.069	0.269	2.173

Table 3 Performance of the individual assets, EWP, and TBPs



Fig. 2 (Color online) Distributions of TBP monthly returns

The remarkable fact is that the EWP serves as a benchmark for evaluating the TBPs, in the sense that there is a (roughly) consistent up–right shift from the point of the EWP on the risk–return plane: as  $\theta$  increases, the return increases from 0.014 (EWP) to 0.015 ( $\theta = 0.000$ ), and then to 0.018 ( $\theta = 0.02$ ); the risk increases from 0.039 (EWP) to 0.04 ( $\theta = 0$ ), and then to 0.069 ( $\theta = 0.02$ ). In the next section, this relationship is more clearly elucidated as the risk–return frontier.

The EWP has been frequently used as a proxy for the risk-return ratio of the financial market, by both academia and the financial industry [41, 42]. It is more diversified than a value-weighted portfolio, which is heavily concentrated into just the largest companies, so that it is being widely traded in the real financial industry (e.g., the NASDAQ-100 Equal Weighted Index allots the same weight to each stock in the index). Therefore, the fact that such a well-known EWP serves as a benchmark for TVPs allows us to more quantitatively characterize TBPs.

θ	No. of correct forecasts	No. of total forecasts	Accuracy
0	343	562	0.61
0.0025	321	521	0.616
0.005	225	405	0.629
0.0075	204	326	0.625
0.01	171	272	0.628
0.0125	134	216	0.62
0.015	110	177	0.621
0.0175	95	152	0.625
0.02	83	134	0.619
0.0225	74	117	0.632
0.025	67	108	0.62

Table 4 Prediction accuracy values for assets whose predictive return is higher than  $\theta$  over the test period

We examine the relationship between the magnitude of predictive returns and the prediction accuracy. Table 4 shows the correct forecasts among all forecasts whose value is larger than  $\theta$ . For the different  $\theta$ s, the prediction accuracy ranges over 0.61 ~ 0.63, independent of  $\theta$ .

We calculate the accumulated returns of TVPs by using the closing prices on the last trading day of each month. We rebalance all portfolios on the last trading day of each month based on the one-month-ahead prediction; we then reinvest according to a weight vector that divides the accumulated wealth equally among the constituents. The accumulated return  $R_t$  is defined as:

$$R_t = \prod_{i=0}^t (1+r_i), \tag{11}$$

where  $r_i$  is the arithmetic return at time *i*. This is a standard performance measure for comparing investments, and it relates the wealth at time *t*,  $W_t$ , to the initial wealth,  $W_0$ , as  $W_t = W_0 \times R_t$ . All experiments in this study used an initial wealth value of  $W_0 = 1$ . Figure 4 shows the cumulative returns of the individual assets, EWP, and TBPs.

# 4.7 Experiment 2: Robustness Test

As a further check, we conduct a similar experiment with the whole of the study period (i.e., January 1, 2006 to December 31, 2014). Figure 4 graphically shows the cumulative returns over the test period (i.e., 30% of the period). Over the test period, the market is more volatile than the previous one, and the LSTM-based predictors shows a poor predictive accuracy value of 0.495. Much of our analysis generated results similar to those in earlier sections, but interestingly, there is a positive relation between the magnitude of predictive return and the prediction accuracy: that is, the accuracy consistently increases from 0.529 ( $\theta = 0.00$ ) to 0.611 at ( $\theta = 0.225$ ), as shown in Table 5.



Fig. 3 (Color online) Cumulative returns of individual assets, EWP, and TBPs



Fig. 4 (Color online) Experiment 2 results: Cumulative returns of individual assets, EW, and TBPs

### **5** Applications

Regarding the practical use of TBPs, we provide illustrations on how to manage them over multiple-periods and how to incorporate them into an MPT optimization portfolio.

# 5.1 TBP Management

Figure 5 is a scatter plot of the risk-return profiles of the TBPs that we are constructed in experiment 1, at different  $\theta_s$  (0.000, 0.0025, ..., 0.025) over the last 10 months, along with the lines fitted to the polynomial of degree 3. We will refer to the line as the "TBP frontiers." Note that the points indicate the realized monthly returns and risk of the (predictive) TBPs built using one-month-ahead

θ	No. of correct forecasts	No. of total forecasts	Accuracy
0	142	268	0.529
0.0025	124	238	0.521
0.005	115	218	0.527
0.0075	104	192	0.541
0.01	96	172	0.558
0.0125	79	143	0.552
0.015	73	130	0.561
0.0175	63	108	0.583
0.02	57	95	0.6
0.0225	52	85	0.611
0.025	46	73	0.605

Table 5 Experiment 2 results: Prediction accuracies for the assets whose predictive return is higher than  $\theta$  over the test period



Fig. 5 (Color online) The realized monthly return versus risk of the TBPs, which are constructed using one-month-ahead return predictions, at  $\theta = 0.000, 0.0025, \ldots, 0.025$  on the last 10 months of the test period of experiment 1. T is the last business month.

predictive returns. The TBP lines are concave, moving upward and to the right as  $\theta$  increases, thus indicating that the greater the amount of risk by the increase of  $\theta$ , the greater the realized returns. This characteristics allows for the design of a TBP with a target risk-return.

To illustrate, let us suppose that an investor at time T-9 hopes to build a TBP to achieve a target risk-return at time T-8. If the target is the monthly risk of 0.05 and the monthly return of 0.016, the investor can estimate the  $\hat{\theta}$ , which corresponds to the target from the TBP frontier drawn at T-9; the investor can then build a TBP with the target, using the predictive return generated and

the estimated  $\hat{\theta}$  at time T - 9. Then, at time T - 8, the investor will obtain an approximate return of 0.017 and risk of 0.05, as seen in the TBP frontier moving upward over the period from time T - 9 to T - 8. This difference, 0.017 - 0.015, is the estimation risk of the TBP. As seen in the continual shift of TBPs as time passes, to maintain a target risk-return, the corresponding  $\hat{\theta}$  needs to be updated. The estimation risk of TBP can be quantified by calculating the average of the differences of realization and expectation for both return and risk over a past period. There is the estimation risk, but it seem to be sufficiently small to classify TBPs as having different risk aptitude. The TBP frontier can be more broad, and combined with riskless assets. In summary, we illustrate the TBP management process:

- Step 1. Set a investment universe (stocks, bonds, ETFs, etc.)
- Step 2. Build forecasting models for future stock return or price
- **Step 3.** Select a trading position (long-only, short-only, or long-short) and a weighting method (equal-weighted, prediction-weighed, etc.)
- Step 4. (Backtest) Draw the TBP frontiers at different thresholds
- **Step 5.** Select a target risk–return value and find its corresponding  $\hat{\theta}$  on the TBP frontier
- **Step 6.** (Actual investment) Invest in the TBP with the  $\hat{\theta}$
- **Step 7.** (Realization) Estimate the TBP at  $\hat{\theta}$ , and reinvest in the TBP with the updated  $\hat{\theta}$  from the realized TBP frontier

#### 5.2 Mean-Variance Portfolio

MPT is a mathematically elegant framework for building a portfolio with specific risk-return level. However, it is well known that it is more difficult to estimate the means than the covariances of asset returns [43], and errors in the estimates of means will have a greater impact on portfolio weights than errors in the estimates of covariances. Furthermore, as mean-variance optimization is extremely sensitive to expected returns, any errors therein might make outcomes far from optimal [44, 45]. For this reason, although theoretical and empirical academic studies have examined various MPT aspects, its real-life practical applications have mostly focused on minimum variance portfolios. This estimation error invariably leads to inefficient portfolios, which can be explained by considering the following three sets of portfolios [46].

- True efficient frontier (TEF): the efficient frontier based on true (but unknown parameters)
- Estimated frontier (EF): the frontier based on the estimated (and hence incorrect) parameters
- Actual frontier (AF): the frontier comprising the true portfolio mean and variance points corresponding to portfolios on the estimated frontier

The use of thresholds on predicted returns can help mitigate inefficiency by screening a subset to be predicted more accurately, as shown in Table 5. Figure 6 shows a schematic scenario that  $EF_T$ , which is estimated for the assets screened by a threshold, is located more closely to the TEF. (Here, for simplicity, we ignore the shift in EF due to the change in the asset number.)



Fig. 6 (Color online) Schematic scenario for shifting EF to  $\mathrm{EF}_T$  by screening the universe at a threshold

#### 6 Conclusion

This study proposes a novel framework by which to construct portfolios that target specific risk-return levels. We evaluated the RNN networks while examining the hit ratios of the one-month-ahead forecasts of stock returns, and then constructed TBPs by imposing thresholds for the potential return. The TBPs are more datadriven in building a portfolio than in existing methods, in the sense that they are constructed purely on the basis of data-driven models by a deep learning technique, in absence of any financial mathematics or knowledge. We showed that the EW of the universe plays the role of the reference portfolio to TBPs, and thus serves to quantitatively characterize the TBP. The TBP frontiers show that the threshold is a parameter used to control tradeoff between the risk and the return of portfolios. We discussed how to practically manage TBPs to maintain a target risk-return over multiple-periods; we also discussed the benefit of incorporating TBPs into MPT.

The TBP is promising, since it provides a simple and straightforward way to build portfolios with target risk-returns, using predictions alone. Any prediction model can be basically applied to construct TBPs. As predictors become more accurate, TBPs can achieve greater returns, given a certain level of risk. In this respect, we believe that the TBP is a valuable application of machine learning to modern-day investment practice.

Acknowledgements This work was partly supported by the ICT R&D program of MSIP/IITP [2017-0-00302, Development of Self Evolutionary AI Investing Technology] and the ICT R&D program of MSIP/IITP [2014-0-00616, Building an Infrastructure of a Large Size Data Center].

#### References

1. G. S. Atsalakis, K. P. Valavanis, Surveying stock market forecasting techniques -Part II: Soft computing methods. Expert Systems with Applications 36(3):59325941 (2009)

- 2. Dixon, M., Klabjan, D., Bang, J. H. (2015) Implementing deep neural networks for financial market prediction on the Intel Xeon Phi. https://ssrn.com/abstract=2627258
- 3. Huck, N. (2009) Pairs selection and outranking: An application to the S&P100 index. Journal of Operational Research 196(2):819–825
- 4. Huck, N. (2010) Pairs trading and outranking: The multi-step-ahead forecasting case. Journal of Operational Research 207(3):1702–1716
- 5. Krauss, C., Do, X. A., Huck, N. (2017) Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P500. 259(2):689-702
- 6. Moritz, B., Zimmermann, T. (2014) Deep conditional portfolio sorts: The relation between past and future stock returns. Working paper
- Sermpinis, G., Theofilatos, K. A., and Karathanasopoulos, A. S., Georgopoulos, E. F., Dunis, C. L. (2013) Forecasting foreign exchange rates with adaptive neural networks using radial-basis functions and Particle Swarm Optimization. European Journal of Operational Research 225(3):528-540
- 8. Takeuchi, L. and Lee, Y.-Y. (2013) Applying deep learning to enhance momentum trading strategies in stocks. Working paper
- Cavalcante, R. C., Brasileiro, R. C., Souza, V. L. F., Nbrega, J. P., Oliveira, A. L. I. (2016) Computational Intelligence and Financial Markets: A Survey and Future Directions. Expert Systems with Applications 55:194–211
- Aggarwal, S., Aggarwal, S (2017) Deep Investment in Financial Markets using Deep Learning Models. International Journal of Computer Applications 162(2):40–43
- 11. Gao, T., Li, X., Chai, Y., Tang, Y. (2016) Deep learning with stock indicators and two-dimensional principal component analysis for closing price prediction system. IEEE, Software Engineering and Service Science (ICSESS), 2016 7th IEEE International Conference on
- 12. Zhang, Y. (2015) Using Financial Reports to predict Stock Market Trends With Machine Learning Techniques. Oxford University
- Fischer, T., Krauss, C. (2017) Deep learning with long short-term memory networks for financial market predictions. FAU Discussion Papers in Economics 11/2017, Erlangen. Available at http://hdl.handle.net/10419/157808.
- 14. Pang, X., Zhou, Y., Wang, P. et al. J Supercomput (2018). https://doi.org/10.1007/s11227-017-2228-y

- 15. Ziniu Hu, Weiqing Liu, Jiang Bian, Xuanzhe Liu, and Tie-Yan Liu. (2018) Listening to chaotic whispers: A deep learning framework for news-oriented stock trend prediction. In Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, pp. 261-269.
- Singh, R. Srivastava, S. (2017) Stock Prediction using Deep Learning, Multimedia Tools and Application, 76 (18), 18569–18584.
- 17. Markowitz, H. (1952) Portfolio Selection. 7(1):77-91
- Fama, E. and French, K. (1992) The cross-section of expected stock returns. Expert Systems with Application 47:427–465
- Black, F., Litterman, R. (1992) Global portfolio optimization. Financial Analysts Journal 48(5):28–43
- 20. Michaud, R. (1998) Efficient Asset Management: A Practical Guide to Stock Portfolio Optimization and Asset Allocation. Boston: Harvard Business School Press
- 21. Haugen, R., Baker, N. (1991) Pairs selection and outranking: An application to the S&P100 index. Journal of Operational Research 17(3):35–40
- 22. Choueifaty, Y., Coignard, Y. (2008) Toward maximum diversification. The Journal of Portfolio Management 35(1):40–51
- 23. Qian, E (2005) Risk parity portfolios: Efficient portfolios through true diversification. Research Paper https://www.panagora.com/assets/PanAgora-Risk-Parity-Portfolios-Efficient-Portfolios-Through-True-Diversification.pdf
- 24. Qian, E (2005) Risk parity portfolios: The next generation. Research Paper https://www.panagora.com/assets/PanAgora-Risk-Parity-The-Next-Generation.pdf
- 25. Faber, M. T. (2014) A Quantitative Approach to Tactical Asset Allocation. The Journal of Wealth Management 9(4): 69–79
- 26. Keller, W. J., Keuning, J. W. (2014) Momentum, Markowitz, and Smart Beta, A Tactical, Analytical and Practical Look at Modern Portfolio Theory. https://ssrn.com/abstract=2759734 or http://dx.doi.org/10.2139/ssrn.2759734
- 27. Keller, W. J., Bulter, A. (2014) A Century of Generalized Momentum; From Flexible Asset Allocations (FAA) to Elastic Asset Allocation (EAA). https://ssrn.com/abstract=2543979 or http://dx.doi.org/10.2139/ssrn.2543979
- 28. Keller W. J., Keuning J. W. (2009) Protective Asset Allocation (PAA): A Simple Momentum-Based Alternative for Term Deposits. Expert Systems with Applications 36(3):5932–5941

- 29. Campbell, J. Y., Sanford J. G., Jiang, W. (1993) Trading volume and serial correlation in stock returns. Quarterly Journal of Economics 108: 905–939
- 30. Choueifaty, Y., Coignard, Y. (2000) Trading volume and cross-autocorrelations in stock returns. The Journal of Finance 55:913–935
- Bao, W., Yue, J., Rao, Y. (2017) A deep learning framework for financial time series using stacked autoencoders and long-short term memory. PLoS ONE 12(7): e0180944. https://doi.org/10.1371/journal.pone.0180944.
- 32. Freitas, F.D., De Souza, A.F., de Almeida, A.R. (2009), Prediction-based portfolio optimization model using neural networks, Neurocomputing 72: 2155-2170.
- 33. Mishra, S., K., Panda, G., Majhi, B. (2016) Prediction based mean-variance model for constrained portfolio assets selection using multiobjective evolutionary algorithms. Swarm and evolutionary computation. 28: 117-130
- 34. Ganeshapillai, G., Guttag, J., Lo, A. (2013) Learning connections in financial time series, in: ICML.
- Elman, Jeffrey L. (1990) Finding Structure in Time. Cognitive Science 14:179– 211
- Bengio, Y. and Simard, P., Frasconi, P. (1994) Learning Long-Term Dependencies with Gradient Descent is Difficult. IEEE Transactions on Neural Networks 5(2):157–166
- Hochreiter, S., Schmidhuber, J. (1997) Long short-term memory. Neural Computation 9(8): 1735–1780
- Cho, K., van Merrienboer B., Gülçehre Ç., Bougares, F., Schwenk, H., Bengio, Y. (2014) Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. CoRR abs/1406.1078
- Diederik P. K., Jimmy B. (2014) Adam: A Method for Stochastic Optimization. http://arxiv.org/abs/1412.6980
- 40. Chollet, F. (2016) Keras: Deep learning library for theano and tensorflow. https://keras.io, 2016.
- 41. Jegadeesh, N., Titman, S. (2001) Profitability of momentum strategies: an evaluation of alternative explanations. Journal of Finance 56: 699–720.
- 42. Y. Plyakha, R. Uppal, Vilkov, G. (2012) Why does an equal-weighted portfolio outperform value-and price-weighted portfolios? Available at SSRN 1787045.
- 43. Merton, R. C. (1980) On estimating the expected return on the market: an explatory investigation. Journal of financial economics 8: 323–361

- 44. Jorion, P.(1985) International portfolio diversification with estimation risk. Journal of business 58(3): 259-278
- 45. Best, M., Grauer, R. (1992) Positively weighted minimum-variance portfolios and the structure of asset expected returns. The journal of financial and quantitative analysis 27(4): 513–537
- 46. Broadie, M., (1993) Computing Efficient Frontiers using Estimated Parameters. Annals of Operations Research, 45: 21–58.