Taylor & Francis
Taylor & Francis Group

Journal of Experimental & Theoretical Artificial Intelligence

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/teta20

Deep asset allocation for trend following investing

Saejoon Kim & Hyuksoo Kim

To cite this article: Saejoon Kim & Hyuksoo Kim (2021): Deep asset allocation for trend following investing, Journal of Experimental & Theoretical Artificial Intelligence, DOI: 10.1080/0952813X.2021.1908429

To link to this article: https://doi.org/10.1080/0952813X.2021.1908429

JOURNAL OF EXPERIMENTAL & THEORETICAL ARTIFICIAL INTELLIGENCE
https://doi.org/10.1080/0952813X.2021.1908429

Taylor & Francis
Taylor & Francis Group

ARTICLE

Deep asset allocation for trend following investing

Saejoon Kim and Hyuksoo Kim

Department of Computer Science and Engineering, Sogang University, Seoul, Korea

ABSTRACT

Trend following strategies are well-known to exhibit excellent excess return performance across a wide range of asset classes in various global markets. For the equity asset class in particular, while the securities selection part is relatively a straightforward procedure, the weight allocation part is more debatable and it has traditionally been identified with the equal-weighted allocation strategy. In this paper, we examine security's own return-based weight allocation strategy for trend following investing and find that this strategy generates superior returns to several well-established weight allocation schemes. In particular, if the true return of the holding period is used *ex ante* for weight allocation, it is found that this strategy can generate absolutely huge excess returns. Motivated by this finding, we investigate the efficacy of machine learning techniques for regression of securities' returns to improve the weight calculation in this framework. Empirical results indicate that deep learning provides the means of regression with which largest excess return gains are possible. In particular, it is demonstrated that the return-based weight allocation strategy defined by our proposed deep learning architecture produces substantial abnormal returns outperforming all other broadly recognised weight allocation schemes compared in this paper.

ARTICLE HISTORY
Received 26 October 2019
Accepted 22 March 2021

KEYWORDS
Trend following; momentum strategy; deep learning; autoencoders; marginalised denoising stacked autoencoders

Introduction

Trend following investing has existed as an investment strategy for over a century now (Hurst et al., 2017; Lim et al., 2018) and for a good reason. It has generated positive excess returns persistently and pervasively across a wide range of asset classes in different markets over an extensive time period (Babu et al., 2020). Its popularity is exhibited in the returns gained by various investing participants. For example, at the end of 2014, managed futures funds or commodity trading advisors (CTAs) who manage assets that are more than 11% of 2.8 trillion USD hedge fund industry use trend following as their predominant investment strategy (Baltas & Kosowski, 2017). The commonly employed form of trend following investing is the time series momentum strategy (Moskowitz et al., 2012). In Moskowitz et al. (2012), it was shown that time series momentum has generated positive excess returns for an extensive set of 58 futures including equity, bond and commodity futures, and currency forwards investigated since 1985 which was when all data considered in its work became available and had sufficient liquidity. The basic idea of time series momentum strategy is extremely simple: long or short an asset based on the sign of its own past return. So the strategy profits when the asset's return exhibits continuation of the same directional movement.

Trending of assets or the momentum effect has been acknowledged as a systematic driver of asset returns, and consequently, momentum has been conceived as the fourth factor (Carhart, 1997)

CONTACT Saejoon Kim saejoon@sogang.ac.kr Department of Computer Science and Engineering, Sogang University, Korea

in asset pricing models in addition to the three factors of Fama and French (Fama & French, 1996, 2012). Elucidation of economic rationale and human behaviour that attribute to the success of trend following has been well documented in various places (see, for example, Grundy & Martin, 2001; He & Li, 2015; Hurst et al., 2013 and references therein).

Documentation of the first comprehensive empirical study of the momentum effect was provided in (Jegadeesh & Titman, 1993) in which it was shown stocks that have outperformed others in the recent past continue to outperform in the near future. In a later study (Jegadeesh & Titman, 2001), they confirm that the momentum effect is a persistent phenomenon that is unique to this strategy not found in other factors. Persistent momentum effect in asset classes including equities, currencies, government bonds and commodity futures across various global markets has been exhibited in Asness et al. (2013) as well. Momentum effects observed in Asness et al. (2013) and Jegadeesh & Titman (1993, 2001) are more accurately called cross-sectional momentum which are characterised by relative returns instead of the absolute returns of the time series momentum. Nonetheless, the concept of momentum effect applies to and is robust in both versions. Comprehensive empirical evidence of the time series momentum effect has been presented in Moskowitz et al. (2012), and the means by which the effect of its occasional crashes can be mitigated has been demonstrated in Barroso & Santa-Clara (2015) and Daniel & Moskowitz (2016). Finally, time series momentum strategies evaluated globally on 67 markets across four major asset classes for over a century dating back to 1880 has validated that momentum is a true phenomenon (Hurst et al., 2017). It has remained as a powerful anomaly throughout the economic vicissitudes that span more than a century during which the Great Depression and the recent financial crisis of 2007–08 took place. In particular, it is the only known factor that has persisted throughout time unaffected by the finding of its risk premium (McLean & Pontiff, 2016).

Implicit in all of the above-mentioned trend following strategies is the unequivocally unvarying asset allocation scheme for which the equal-weighted strategy (Hurst et al., 2017; Jegadeesh & Titman, 2001) or in some cases the value-weighted one is the standard. The equal-weighted strategy has garnered a reputation as an effective asset allocation scheme as it was shown to be superior to most of the in-sample-optimised portfolio strategies in various places such as in (DeMiguel et al., 2009; Ernst et al., 2017; Hsu et al., 2018). The superiority of the equal-weighted scheme can be attributed to a number of reasons including the mean reversion property of assets and the absence of estimation error (Malladi & Fabozzi, 2017; Michaud, 1989). Regarding the latter, for example, if the size of the error of the out-of-sample estimate obtained during the in-sample portfolio optimisation becomes too large, then the resulting portfolio may become far from optimal with respect to the objective function under consideration. Such an unintended consequence is not a rare phenomenon as it was observed for the mean-variance optimised portfolio in (Michaud, 1989). In the equal-weighted strategy, such out-of-sample estimate error is non-existent which allows itself never to construct a portfolio that is far from the intended one. Furthermore, optimisation of asset weights can be a dangerous procedure as a variety of adverse effects can be introduced as noted in (Smith & Wallis, 2009). For these reasons, many portfolio optimisation strategies such as the (norm-unconstrained) mean variance or the value-weighted strategy have resorted to the equal-weighted scheme for asset allocation (Hsu et al., 2018; Malladi & Fabozzi, 2017). In this paper, we will restrict our argument to the confines of the more practically viable *long-only* trend following investing as the asset class under consideration is the equity. So, in the aforementioned equal-weighted trend following strategy, the portfolio consists of equal weights of securities that had positive return in the look-back period.

On the other hand, the logic behind the trend following strategy in particular suggests that past return information be also incorporated in the asset allocation part. Specifically, the trend following strategy is based on the premise that an asset's own past return performance is indicative of its future return performance. For the portfolio that is rewarded for this property, it would be more adequate if asset allocation is proportional to assets' past returns instead of being altogether equal if one wishes for higher returns.

Going one step further, it would even be better if the asset allocation scheme can be defined by assets' future returns. In this paper, we conduct a comprehensive empirical study to test this argument on whether return-based asset allocation scheme can generate superior returns compared to the commonly used equal-weighted strategy. As we will show in the next section, it follows that one can obtain higher returns by setting an asset's weight in a trend following portfolio proportional to the asset's realised return in the look-back period. Specifically, if $r_i\ (>0)$ denotes the return of asset i in the look-back period, then by setting $\frac{r_i}{\sum_i r_i}$, where the summation in the denominator is over all assets in the investable universe with positive returns in the look-back period, as the weight of asset i, improved return performance can be gained in comparison to the equal-weighted strategy. Furthermore, if $r_i\ (>0)$ denotes the return of asset i in the holding period, then using the same ratio as the weight for asset i results in absolutely huge performance gain. A qualitative aspect that is important about this finding is that this motivates us to accurately predict the returns of assets in the holding period for higher performance gain, and a main contribution of this work is the provision of solutions to this prediction problem.

To this end, we investigate in this paper algorithms from machine learning or artificial intelligence for time series regression to which the aforementioned prediction problem belongs. The motivation lies in recent results in the literature that have shown these algorithms to be preferable to the classical regression schemes such as linear regression. For example, support vector regression has proven to be very effective for prediction of financial time series in various places that include (Kim, 2018; Kim & Heo, 2017; Sapankevych & Sankar, 2009) and references therein. It should be noted that support vector regression is generally accepted as one of the best regression schemes within the realm of *shallow* machine learning which has been the dominant machine learning technology prior to the late 2000s. Afterwards, *deep* machine learning (Lecun et al., 2015) emerged as the 'new' technology where it first gained attention by producing unprecedented high accuracy results for image classification applications. The success and the defining characteristic of deep machine learning lie in learning deep hidden features which shallow machine learning techniques are incapable of. Lately, this characteristic has led to deep learning-based regression schemes finding success in the financial applications domain, and some of these examples include the prediction of foreign exchange rates in Shen et al. (2015), stock price forecasting in Bao et al. (2017), and return prediction of momentum strategies in Kim (2019). In this paper, we propose a novel deep learning methodology that has not been considered for financial applications previously, and this methodology will establish superior asset allocation for trend following investing to traditional approaches in terms of various forms of returns and return to risk ratios.

The main contribution of this paper is twofold. The first is the aforementioned construction of the new asset allocation scheme based on the assets' realised returns. Return-based asset allocation will be compared favourably with some of the more popular existing asset allocation schemes such as the equal-weighted, equal volatility-weighted and risk parity strategies. Specifically, return-based asset allocation scheme will be shown to yield higher annualised and risk-adjusted returns, and superior higher order moments of returns, in general, to the existing schemes. We note that all asset allocation schemes considered here outperform the market index by a sizeable margin even after accounting for transaction costs. In particular, in terms of cumulative returns near the end of the holding period, it will be shown that the amount of outperformance achieved by our proposed return-weighted asset allocation scheme to the traditional asset allocation schemes is approximately the same as that by the traditional allocation schemes to the market index.

In the second contribution, we extend the result of the first one by addressing the weight selection or the asset allocation problem through the lens of deep learning. For this purpose, we establish some of the latest deep learning architectures (Bengio et al., 2006; Chen et al., 2014; Vincent et al., 2010) for the prediction of returns which is used to define the weight of each asset in our trend

following portfolio. In particular, it will be demonstrated that our proposed deep learning architecture, namely, marginalised denoising stacked autoencoder, is the only deep learning architecture that consistently shows performance advantage over the realised return-based asset allocation of the first contribution. In other words, no other machine learning architecture, both shallow and deep, showed consistent superior performance, measured in terms of annualised and risk-adjusted returns and higher order moments, to the realised return-based asset allocation scheme for all data sets considered. To the best of our knowledge, this is the first work that this architecture has been established for any type of financial applications domain. As will be presented later in the paper, this result can be attributed to the inherent structure of the marginalised denoising stacked autoencoder which affords itself the property of superior generalisability across different domains.

The outline of this paper is as follows. In the next section, we examine the performance characteristics of various asset allocation schemes defined for the trend following strategy. Motivated by the results shown in Section 2, in Section 3, we present the main contribution of this paper that is the provision of deep learning architecture and algorithm specifically designed to generate prediction of assets' returns. In Section 4, extensive empirical performance results of the trend following strategies defined by various asset allocation schemes are evaluated, and we conclude in the following section.

Trend following investing

In this section, we provide empirical evidence that return-weighted trend following portfolio can generate nontrivially larger returns than the traditionally weighted trend following portfolios such as the equal-weighted portfolio. We note that unless otherwise specified, all weighing schemes are in-sample-based. To substantiate our argument, we also experiment with the unimplementable out-of-sample (OOS) return-weighted trend following portfolio that will be demonstrated to produce extremely large returns. We have included this unimplementable portfolio to illustrate the amount of gains that can be achieved by correctly weighing the portfolio constituents by their actual future returns. The extremely large gains that are offered by this unrealisable portfolio has served as the motivation to conduct research of this paper.

Let us first define the trend following portfolios that we consider in this paper. All trend following portfolios in this paper consist of securities that had positive return in the past one-year period. The only difference between the various types of trend following portfolios lies in how the weights are endowed to each constituent. To describe the weighing schemes applied to these portfolio constituents, let us consider the following four types in this section:

(1) equal-weighted (EW)
(2) equal volatility-weighted (EVW)
(3) risk parity (RP)
(4) return-weighted (RW)

In the following, w_i denotes the weight of asset i. The equal-weighted scheme (EW) is defined by $w_i = \frac{1}{N_P}$ for all i where N_P denotes the portfolio cardinality, *i.e.*, the number of securities that had positive return in the past one-year period. The equal volatility-weighted scheme (EVW) is defined by $w_i = \frac{1/\sigma_i}{\sum_i 1/\sigma_i}$ where σ_i denotes the volatility of security i in the past one-year period and the summation over i in the denominator refers to summation over all securities in the trend following portfolio of size N_P. Portfolio optimisation by weighing securities inversely proportional to their volatilities, sometimes also known as the low-volatility portfolio, has received popularity in the recent years due to their consistent generation of favourable returns. (See (Kim, 2018; Yang et al., 2019) and references therein.) For this matter, we have included this weighing scheme for comparison. The risk parity scheme (RP) (Maillard et al., 2010; Qian, 2011) defines the portfolio where risk contribution to the portfolio by each security is the same for all portfolio constituents. The weights can be calculated by solving an optimisation problem and we used a simple approach provided in (Bai et al., 2016). It

was demonstrated in (Yang et al., 2019) that the risk parity scheme which is conceptually an improvement over the equal volatility-weighted scheme does indeed provide improved out-of-sample performance over the latter. The return-weighted scheme (RW) is defined by $w_i = \frac{r_i}{\sum_i r_i}$, where r_i denotes the return of security i in the past one-year period and the summation over i in the denominator is defined the same as above. Let us now describe the data sets on which we evaluate the trend following strategies. They are the following:

(1) Securities in S&P 500 ranging from 1997. 6. 3 to 2018. 11. 16.
(2) Securities in FTSE 100 ranging from 2001. 1. 3 to 2018. 12. 31.
(3) Securities in KOSPI 200 ranging from 2000. 5. 29 to 2019. 2. 8.

S&P 500 is from the US, FTSE 100 is from the UK, and KOSPI 200 is from South Korea. The source of S&P 500 data set was Yahoo! Finance (finance.yahoo.com), that of FTSE 100 was Investing.com (investing.com), and that of KOSPI 200 was Koscom (koscom.co.kr). The data sets were collected on the last day of the coverage period for each data set. In our experiments, we set the formation period equal to one year and the holding period equal to the following one month, and applied monthly rebalancing. To account for transaction costs incurred for each rebalancing, we subsumed 0.1% one-way transaction cost and 75% turnover which is typical for trend following-type strategies (Barroso & Santa-Clara, 2015; Hurst et al., 2017; Kim, 2019), and deducted accordingly for all trend following strategies considered in this paper. To compare how these portfolios fare with the market index (Mkt), we also included the performance measures for the market index which did not incur any transaction cost and therefore no deduction in return was made for this strategy only. Tables 1 ∼ 3 summarise the performance measure values of the aforementioned trend following strategies along with the market index on S&P 500, FTSE 100, KOSPI 200 data sets, respectively. Among the performance measures, 'Average Return,' 'Maximum Return' and 'Minimum Return' refer to the corresponding monthly returns, and for the calculation of *t*-statistic, we assumed risk-free rate of zero. Kurtosis and skewness are measures for the tailedness and the asymmetry, respectively, of the return distribution, and MDD represents the maximum drawdown which measures the largest peak-to-trough decline in the value of a portfolio. It is represented here as the percentage of the lowest portfolio value in comparison to the highest one during the largest peak-to-trough decline.

Table 1 shows that, in terms of annualised return only, RW outperforms EW which outperforms EVW and RP. However, in terms of risk-adjusted return measured through Sharpe ratio, RP performs the best. Higher order moments favoured RW as it had the smallest kurtosis and least negative skewnews. Table 2 shows a similar trend, except in this data set, RW also has the highest Sharpe ratio in addition to the highest annualised return. RW showed *positive* skewness and the smallest maximum drawdown. Table 3 shows a very similar trend as Table 2. RW outperforms EW, in terms of the annualised return, which outperforms RP and EVW. RW outperforms RP, in terms of the Sharpe ratio, which outperform EW and EVW. RW achieves the smallest kurtosis and the least negative skewness. In all tables shown, all trend following strategies show statistically significant, *e.g.*, *t*-

Table 1. Performance of trend following strategies using various weighting schemes on S&P 500 data.

	Mkt	EW	EVW	RP	RW
Annualised Return	0.035	0.097	0.086	0.093	0.118
Annualised Vol.	0.146	0.142	0.129	0.127	0.179
Average Return	0.004	0.009	0.008	0.008	0.011
t-statistic	1.338	3.149	3.077	3.322	3.108
Maximum Return	0.104	0.102	0.090	0.089	0.203
Minimum Return	−0.252	−0.269	−0.251	−0.250	−0.293
Sharpe Ratio	0.236	0.681	0.668	0.730	0.659
Treynor Ratio	0.035	0.110	0.108	0.120	0.124
Info. Ratio	0.000	0.927	0.732	0.773	0.711
Kurtosis	5.377	8.672	9.662	10.296	5.281
Skewness	−1.326	−1.715	−1.874	−1.940	−0.921
MDD	0.490	0.451	0.424	0.428	0.494

Table 2. Performance of trend following strategies using various weighting schemes on FTSE 100 data.

	Mkt	EW	EVW	RP	RW
Annualised Return	0.030	0.075	0.066	0.071	0.114
Annualised Vol.	0.152	0.138	0.133	0.131	0.174
Average Return	0.003	0.007	0.006	0.006	0.010
t-statistic	1.078	2.313	2.149	2.300	2.781
Maximum Return	0.138	0.125	0.110	0.110	0.247
Minimum Return	−0.194	−0.145	−0.148	−0.141	−0.143
Sharpe Ratio	0.201	0.538	0.496	0.538	0.656
Treynor Ratio	0.030	0.092	0.084	0.091	0.135
Info. Ratio	0.000	0.542	0.442	0.495	0.647
Kurtosis	3.730	1.596	1.729	1.479	2.849
Skewness	−0.688	−0.449	−0.604	−0.540	0.155
MDD	0.465	0.415	0.434	0.425	0.406

Table 3. Performance of trend following strategies using various weighting schemes on KOSPI 200 data.

	Mkt	EW	EVW	RP	RW
Annualised Return	0.084	0.183	0.172	0.180	0.265
Annualised Vol.	0.198	0.216	0.202	0.202	0.291
Average Return	0.008	0.016	0.015	0.016	0.023
t-statistic	2.021	3.534	3.523	3.655	3.807
Maximum Return	0.199	0.210	0.203	0.244	0.286
Minimum Return	−0.316	−0.296	−0.278	−0.277	−0.359
Sharpe Ratio	0.423	0.847	0.848	0.888	0.912
Treynor Ratio	0.084	0.205	0.203	0.221	0.242
Info. Ratio	0.000	0.697	0.634	0.632	0.889
Kurtosis	5.315	3.641	3.758	4.465	2.421
Skewness	−0.883	−0.363	−0.356	−0.126	−0.026
MDD	0.475	0.483	0.462	0.448	0.528

statistic greater than 2, superior performance to the market index, and the RW strategy was exhibited with the best overall performance among all four trend following strategies considered in this section.

Next we show the cumulative return results for the three data sets in Figure 1–3, respectively. Each of these figures contains two sets of figures (top and bottom) where one set spans over the entire coverage period and the other set spans ten years roughly centred around the time of global financial crisis of 2008. In each set, the one on the left shows the cumulative return curves for the five strategies shown in the previous tables. The one on the right shows the cumulative return curve of the out-of-sample (OOS) return-weighted scheme (RW-OOS). This scheme has the weight of each security defined the same as the return-weighted scheme (RW) except the return of the security is the actual return of the out-of-sample or holding period. Hence this weighing scheme provides the upper limit of return performances of portfolios that weigh securities based on their respective returns. Separation of the RW-OOS curve from the others was necessary as the return for the RW-OOS strategy completely dwarfed that of all other strategies.

Across all data sets, the figures show EW, EVW and RP give very similar, if not nearly identical for some data set, performance over the entire coverage period and around the financial crisis period. Also across all data sets, RW outperforms the three strategies EW, EVW and RP, and the performance gap between RW and the group of three traditionally weighted strategies was roughly as large as that between this group and the market index. Moreover, this performance differential behaviour was consistent throughout the entire period including the period that covered the financial crisis of 2008 as is better illustrated in the bottom figure. To get a better view of this persistency, we reset the initial value of returns in the bottom figure in each of Figures 1 $\sim$ 3 to one.

In all figures shown, the RW-OOS strategy completely obliterates all other strategies in terms of cumulative returns. This finding along with that of the outperformance of the RW strategy to all other strategies considered here by a sufficiently large margin suggest that in order to execute effective

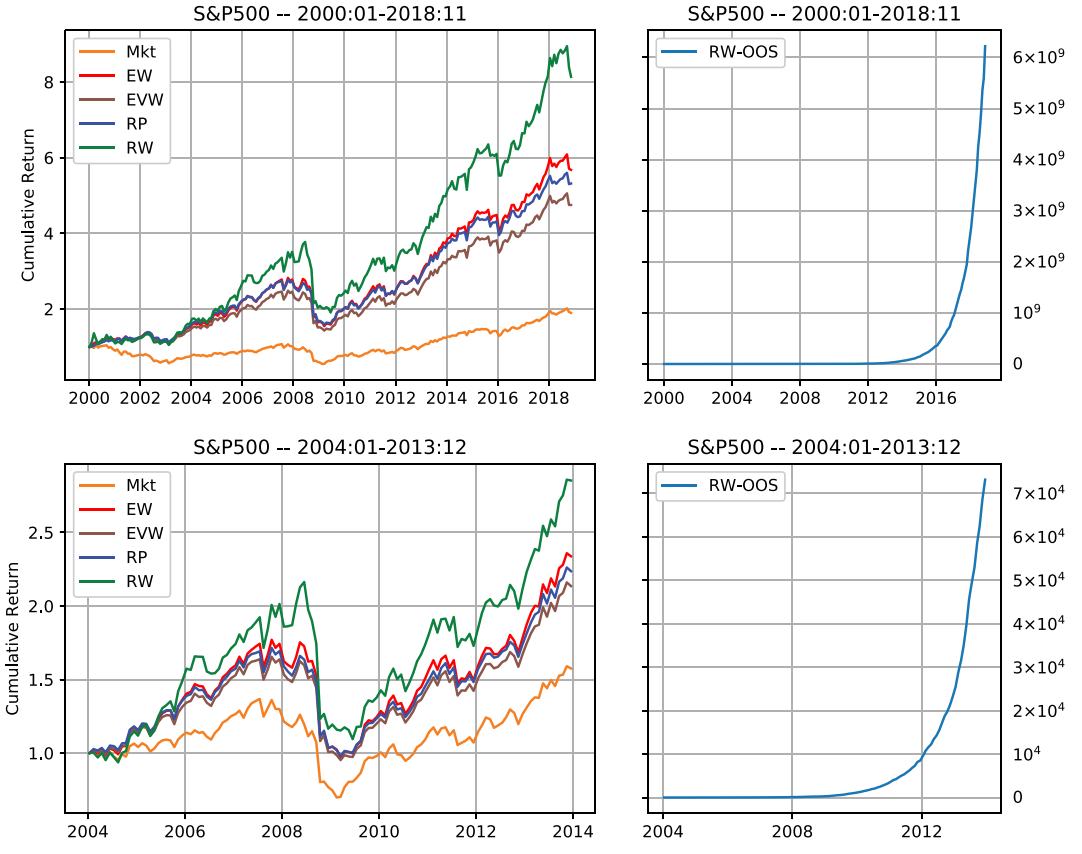

Figure 1. Cumulative return curves of trend following strategies for S&P 500 data.

asset allocation for trend following investing, one should consider the return-weighted strategy and, in particular, use estimates of OOS returns of portfolio constituents for this strategy. To this end, we investigate the efficacy of machine learning techniques for this purpose in the next section. In particular, we delve into the latest deep learning techniques designed specifically for regression of returns.

Deep learning for asset allocation

Deep learning is essentially achieved through feeding input data into a *deep neural network* (DNN), which consists of a sequence of layers of neural networks, for the target task. The multiple layers of neural networks process multiple levels of abstraction of the input data that enables DNN to perform *representation learning* (Bengio et al., 2013). This representation learning part of deep learning is deficient in the traditional 'shal- low' learning techniques, and it is this part that mainly gives deep learning techniques the outperformance advantage over the traditional shallow learning ones (Lecun et al., 2015). For this reason, empirical success offered by deep learning is perceived to have come from two sources: representation learning and effective optimisation. In this work, we will invoke the former to establish deep learning for asset allocation.

Elsewhere, it is well established that support vector regression (SVR), a form of shallow learning, may provide good predictions of financial time series (see, for example, Kim, 2018; Law & Shawe-Taylor, 2017; Sapankevych & Sankar, 2009 and references therein). Specifically, it is well documented in Kim (2018) and Sapankevych & Sankar (2009), among others, that SVR is the best performing

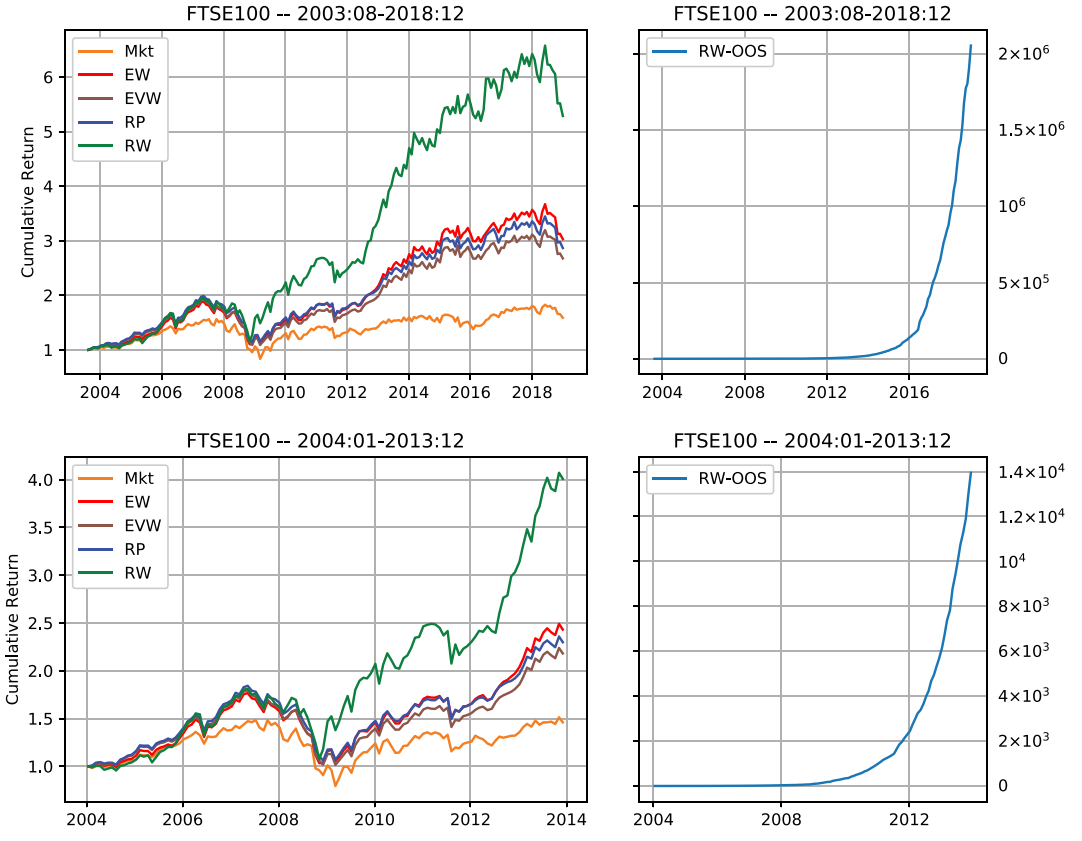

Figure 2. Cumulative return curves of trend following strategies for FTSE 100 data.

shallow learning technique for non-stationary time series prediction. In particular, it was demonstrated in Kim & Heo (2017) that SVR outperforms ARIMA models for the same problem in the framework of pairs trading.

On the other hand, the empirical success of shallow learning techniques, including SVR, is well known to heavily depend on the representation of the input. For this reason, feature selection has played an important role in the realm of shallow learning, however, with no guarantee that the generated features set well represents the raw input data. In other words, adequate representation of input data has been a main source of performance limitation for shallow learning techniques. To this end, with the advent of deep learning technology, we address the inherent problem of adequate input representation in SVRs through DNNs. Specifically, we invoke the representation learning characteristic of DNNs to generate the input representation for SVRs which will be used to make returns time series prediction.

While various realisations of DNNs exist that comprise, for example, layers of recurrent neural networks, convolutional neural networks and restricted Boltzmann machines, we consider in this paper a particular form of DNN called stacked autoencoders (SAEs) that is particularly suited for effective regression of time series data (Kim, 2019). The main goal of SAEs is the generation of deep features of the input data such that these generated features which, by construction, are good concise representation of the input data can help better solve the target task. SAEs were first conceived in Bengio et al. (2006) and have since been utilised for a wide range of applications with great successes which motivates us to investigate them in our setting of returns time series prediction. To list a couple

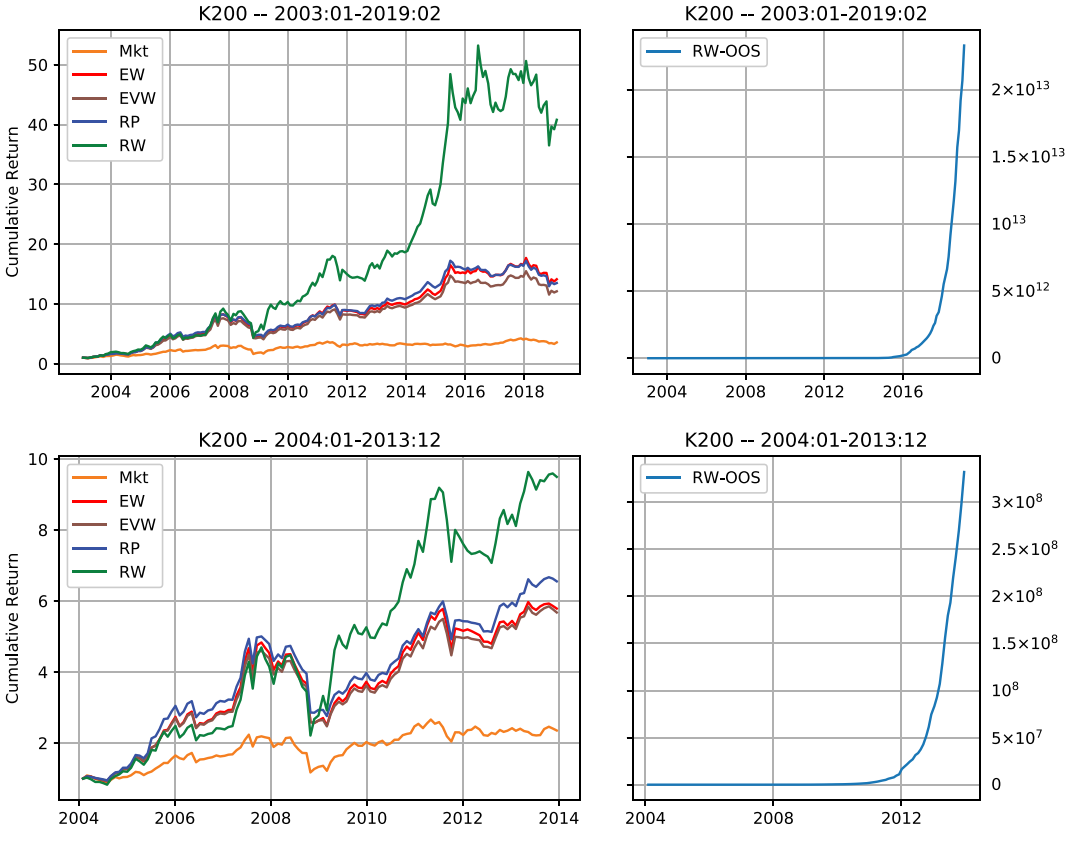

Figure 3. Cumulative return curves of trend following strategies for KOSPI 200 data.

that is relevant to our work, they have been employed for stock price forecasting in Bao et al. (2017) and for constructing portfolios that replicate stock indices in Kim & Kim (2020).

In the next subsection, we present a brief description of stacked autoencoders. Then, we introduce marginalised denoising stacked autoencoders (mdSAEs) for the first time in the financial applications domain that will play the key role in the provision of consistent outperformance of a machine learning-based asset allocation scheme to the traditional non-machine learning-based ones. Specifically, it will be shown that only mdSAE-based asset allocation scheme, among all shallow and deep machine learning-based schemes tested, consistently produced superior results in all of the asset allocation schemes of Section 2 across data sets of different economic regions. In the following subsection, overall deep learning architecture that establishes our proposed return regression technique is provided.

Stacked autoencoder

Stacked autoencoder (SAE) (Bengio et al., 2006) is simply a set of autoencoders that are stacked or cascaded in a sequential manner where an autoencoder is a single hidden layer neural network designed to reconstruct the input at the output. An example of an autoencoder is shown in Figure 4. In the figure, for an input $x \in \mathbb{R}^d$, $y \in \mathbb{R}^{d'}$ is the hidden layer representation, and $z \in R^d$ is the reconstruction of x where f is some (typically) nonlinear activation function. They are related by the following:

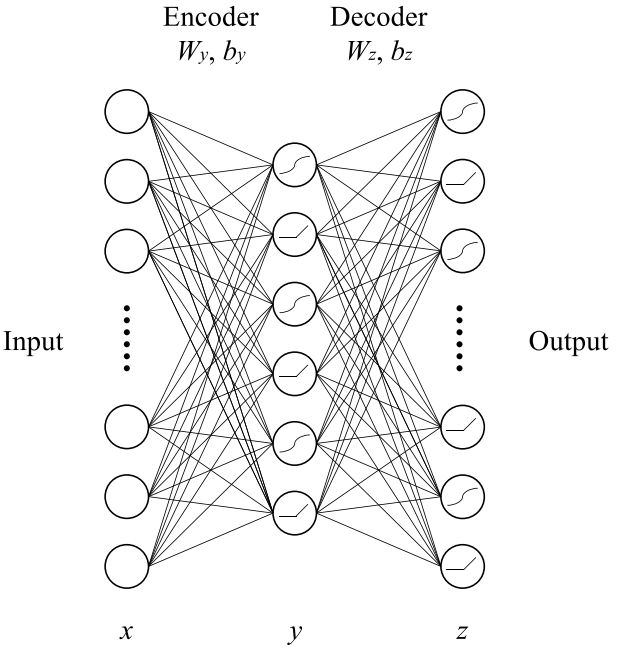

Figure 4. Autoencoder.

$$y = f(W_y x + b_y), \tag{1}$$

$$z = f(W_z y + b_z) \tag{2}$$

where W_y, W_z are matrices and b_y, b_z are vectors to be found. The objective of the autoencoder is the minimisation of the reconstruction error $L(x, z)$, such as the squared error loss $||x - z||^2$, by carefully selecting the hyperparameters of the autoencoder W_y, W_z, b_y, b_z. Calculation of these hyperparameters which is served through the back-propagation algorithm constitutes the pre-training and fine-tuning phases of the autoencoder, and if hyperparameters that yield small reconstruction error can be found, then y can be interpreted as a good abstraction or features set for x. The reason for using a nonlinear function, *e.g.*, the rectified linear unit (ReLU) or the hyperbolic tangent function (tanh), for $f(x)$ is to avoid the trivial autoencoders in which the identify mapping ($d = d'$ case) or the principal component analysis ($d' < d$ case) is performed when x is transformed to y and then to z.

In an SAE, the hidden layer representation y of the first (or the lowest level) autoencoder is used as the input to the next (or the second level) autoencoder in the cascade. Pre-training and fine-tuning of the second autoencoder is then processed which is done identically as but independently of the first autoencoder. As in the case of the first autoencoder, pre-training and fine-tuning phases generate the hidden layer representation of the second autoencoder. The same procedure is carried out repeatedly until the final autoencoder of the cascade is reached. Specifically, the hidden layer representation of the second level autoencoder is used as the input to the next (or the third level) autoencoder which can then be pretrained, and so forth. Multiple layers of autoencoders give SAE the ability to extract multiple levels of abstraction of the input data that can efficaciously be used for a target task. In particular, the hidden layer of an autoencoder in the cascade can be interpreted to possess better concise description than that of a lower level autoencoder of the original input x (Bengio et al., 2006). A defining characteristic of SAE in which the overall network is trained hierarchically one autoencoder at

a time endows the hidden layer of the final autoencoder the capacity to uncover deep latent representation of x (Kim & Kim, 2020).

Marginalised denoising stacked autoencoder

Marginalised denoising stacked autoencoder (mdSAE) Chen et al., 2014) improves on SAE in two aspects. First, it extends SAE by introducing artificial noise to the input data to obtain denoising stacked autoencoder (dSAE) (Vincent et al., 2010). dSAE is founded on the idea that the hidden layer representation of an autoencoder that induces low reconstruction error may not necessarily represent a good abstraction of the actual (clean) input if input was initially tainted with noise. To cope with such cases, dSAE was proposed to 'denoise' the noise-tainted input by minimising the reconstruction error $L(x, z)$ in which x, the clean input, and z, the reconstruction of x, are related through x', the artificially noise-tainted version of x, by the following:

$$y = f(W_y x' + b_y), \tag{3}$$

$$z = f(W_z y + b_z). \tag{4}$$

Comparing Equations (3) and (4) with Equations (1) and (2) the only difference lies in x in Equation (1) being replaced by x' in Equation (3). That is, SAE and dSAE both aim to minimise $L(x, z)$, the reconstruction error between the clean input and its reconstructed version, however, dSAE generalises to encompass the case that the input may be noise-tainted. A common method of choice in producing x' for real-valued input data x is by adding a zero-mean, variable-variance Gaussian noise to x. Vincent et al. (2010) have shown that adding artificial noise to the original input x to get x' and then denoising x' to minimise reconstruction error $L(x, z)$ improves on SAE through empirical and geometrical interpretational results.

Second, notice that the artificially noise-tainted version of the input in the dSAE is obtained through many training epochs of SAE each of which corresponds to generating a particular x' from the input x and the noise distribution. Clearly, more training epochs of SAE are necessary to construct more stable dSAE which limits its practicality due to the high computational cost. To circumvent this problem, mdSAE is built on top of dSAE by marginalising out the noise variable in dSAE in an efficient manner to effectively execute infinitely many epochs of SAE in one epoch of mdSAE. This efficient marginalisation was based on the fact that the empirical average of the reconstruction error is the expected average under the noise corruption distribution as the number of training epochs goes to infinity.

Hence, by construction, mdSAE is superior to dSAE in terms of accuracy for a target task, and it has been verified empirically to run much faster than dSAE of comparable performance (Chen et al., 2014). Moreover, as SAEs are known for their good domain adaption property, mdSAEs can be superior to dSAE for a general set of tasks. Calculation of the hyperparameters W_y, W_z, b_y, b_z that includes marginalising out the noise variable constitutes the pre-training and fine-tuning phases of mdSAE.

Deep features for asset allocation

SAEs and mdSAEs described in the previous two subsections extract deep features from or perform representation learning of the input data. We now relate this obtained deep features set to the input to SVR for our target task which is return regression. Glorot et al. (2011) and Lee et al. (2009) have previously shown evidence that associating features extracted from stacked autoencoders as input to support vector machines is an efficacious construct of machine learning as state-of-the-art empirical performance on sentiment analysis benchmarks was observed. We modify and extend their results in this work to using mdSAE and regression for return prediction. When designing an SVR architecture, there are three types of hyperparameters one needs to select which are the

regularisation constant, the tube size, and the variance of the kernel function if the Gaussian kernel is being used. They can be selected using the validation period which is normally set as some last segment in the formation period.

The overall architecture of our deep learning-based return regression scheme is shown in Figure 5 in which the architecture consists of three layers of autoencoders followed by an SVR unit. As described previously, the input data is fed into the input nodes of the first autoencoder, *i.e.*, the autoencoder at the lowest level or layer 1. The hidden layer nodes of the autoencoder at the highest level, *i.e.*, the autoencoder at layer 3, become the input to the SVR which executes return regression. These hidden layer nodes at layer 3 represent the deepest level of abstraction of the input data in this architecture. Hereafter, when we refer to return regression by SAE or mdSAE, it is assumed that SAE or mdSAE is followed by SVR in the regression architecture. After fine-tuning the SAE or mdSAE in the architecture, the hyperparameters of it are fully established and the only unknown part in the overall architecture at this point is the set of hyperparameters in SVR and the number of nodes in each layer in the autoencoder both of which will be determined during the validation period. Once this is determined, input test data can be fed into the deep learning architecture for return regression.

Performance results

In this section, we present with the empirical results of the return-weighted trend following strategies which will confirm the main result of this paper. The strategies are as follows:

(1) RW-LB – look-back period-based return-weighted asset allocation
(2) RW-SVR – SVR-predicted return-weighted asset allocation
(3) RW-SAE – SAE-predicted return-weighted asset allocation
(4) RW-mdSAE – mdSAE-predicted return-weighted asset allocation

RW-LB is the scheme we introduced in Section 2, *i.e.*, RW-LB of this section is RW of Section 2, and RW-SVR, -SAE, -mdSAE are the machine learning-based schemes where the return is predicted using the method of the respective suffixes. Therefore, in this section, RW-LB is the baseline strategy to beat whereas in Section 2, EW served as the baseline strategy which our proposed RW-LB had outperformed. We note that RW-SVR which uses a stand-alone SVR for return regression is included in our experiments to examine how it compares with SVR that takes deep features generated by the

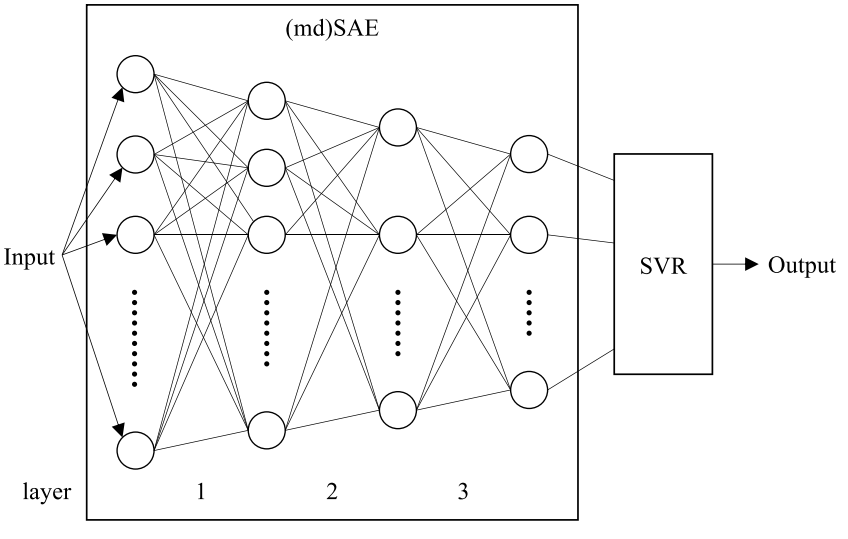

Figure 5. Deep learning architecture for return regression.

deep learning models of the previous section as inputs. It will be demonstrated shortly that these deep features played the key role in establishing more accurate return time series prediction as the only difference between RW-SVR and RW-(md)SAE was the input data to SVR. In other words, the box in the middle shown in Figure 5 is missing in RW-SVR. We also note that some of the securities with positive return in the look-back period may have negative return prediction in the holding period. Therefore, the portfolio size of machine learning-based schemes may be slightly smaller than that of RW-LB.

In accordance with Section 2, the portfolio rebalancing frequency is set to one month. Therefore, if the portfolio strategy is created today, the goal of machine learning-based strategies is to predict the monthly return of the day that is one month in the future from today. More formally, let us denote the monthly return of day i as r_i, and let $i = i^*$ be the day of the portfolio creation, so that the goal is to predict r_{i^*+21}. To train the machine learning-based schemes, the following set of input-output pairs was used on all schemes:

$$\text{input}: \ (r_i, r_{i-4}, \cdots, r_{i-248}) \tag{5}$$

$$\text{output}: \ r_{i+21}$$

for $i = i^* - 21 - j$ where $j = 0, 1, 2, \cdots, 99$. So, we have a total of 100 training vectors per test vector. The input vector in Equation (5) was designed to cover approximately one year which is the same as the formation period length of RW-LB. We empirically found that this length was sufficiently long to capture any pattern in the monthly returns that may exist, and any extraneous information would be suppressed during training of the machine learning scheme.

To test the machine learning-based scheme, *i.e.*, to predict r_{i^*+21}, the input of Equation (5) with $i = i^*$ is fed into the machine learning scheme which outputs the prediction. As is normally done for raw input data before being processed into a machine learning model, both shallow and deep, standardised scaling was applied to the raw input so that values taken by each input node in a machine learning architecture follows the Gaussian distribution with zero-mean and unit-variance. Furthermore, the most recent 10% of training data was used as the validation set to pick the optimal values of the remaining hyperparameters for regression. Specifically, the validation set is used to pick the hyperparameters for SVR and to pick the architecture configuration described below. In SVR, the triple $(C, \in, \gamma)$ in which C is the regularisation constant, $\in$ is the tube size and γ is the parameter of the Gaussian kernel is the set of hyperparameters to be optimised. For this triple, we tried $C \in \{10^i, \ 5 \cdot 10^i : -2 \leq i \leq 3\}$, $\varepsilon \in \{10^i, 5.10^i : -5 \leq i \leq -1\}$, $\gamma \in \{10^i, \ 5 \cdot 10^i : -5 \leq i \leq -1\}$, and the accuracy measure used in the validation set was the mean squared error (MSE).

For SAE and mdSAE, two main hyperparameters relate to the topology of the network which are the number of hidden layers and the number of nodes in each layer. There is no known efficient way to configure these hyperparameters, and for this reason, one can only obtain these hyperparameters through trial-and-error. We tried many combinations and chose three as the number of hidden layers (as shown in Figure 5), and the following set of candidates as the numbers of nodes for each layer:

(1) 63–63–32–32
(2) 63–63–32–16
(3) 63–32–32–16
(4) 63–32–16–16

The first number 63 shown in all architecture configurations corresponds to the number of input values as indicated by Equation (5). The last numbers in the architecture configurations represent the number of values that are input to SVR for regression. So in the first architecture, 32 values from the hidden nodes of the last layer are input to SVR for regression whereas in the next three architectures, 16 hidden node values from the last layer in stacked autoencoders are input to SVR. Of the four architecture configurations listed above, the one that yields the most accurate regression is chosen (in the validation period) as the final architecture configuration for the SAE- or mdSAE-based

regression. For the activation function that appears in Equations (1) $\sim$ (4), we tried ReLU and tanh which are two of the most widely used activation functions. Moreover, Adam (Kingma & Ba, 2015) was used as the learning algorithm for SAE and mdSAE with its hyperparameters set to $\beta_1 = 0.9, \beta_2 = 0.999$ and $\in= 10^{-8}$. The learning rates of Adam were $10^{-i}, i \in \{3,4,5,6\}$ for pre-training and 10^{-i} and $5 \cdot 10^{-i}, i \in \{3,4,5,6\}$ for fine-tuning, and the batch size we tried was 64 for pre-training and 10 for fine-tuning. The noise variance for mdSAE was set to 0.01, 0.005, 0.001, 0.0005 and 0.0001.

Tables 4, 6 and 8 show the results of various performance measures of the trend following strategies defined by the return-weighted schemes on the S&P 500, FTSE 100 and KOSPI 200 data, respectively. We have included Mkt, EW and RP strategies for reference in these tables. Tables 5, 7 and 9 show the performance differential between our proposed scheme of RW-mdSAE, indicated by $*$ in the tables, and the baseline strategies for the S&P 500, FTSE 100 and KOSPI 200 data, respectively. As mentioned before in Section 2, all returns shown are net of transaction fees except those of the market index which does not incur such fees.

First, Table 4 shows that, with respect to annualised return, RW-mdSAE outperforms the other two machine learning-based return-weighted schemes both of which outperform the baseline strategy RW-LB. Similar relative performances of the four return-weighted strategies are shown for the Sharpe ratio. Recall for this data set, RP showed the best Sharpe ratio performance in Table 1. However, in Table 4, RW-mdSAE beat RP also in Sharpe ratio by 0.834 to 0.730. Other machine learning-based methods also show superior Sharpe ratios as well as Treynor and information ratios to those of RP. RW-mdSAE has the smallest kurtosis of 3.26 compared to those of 8.672 and 5.281 for EW and RW-LB, respectively. RW-mdSAE has the second least negative skewness of −0.836 followed by RW-SVR with −0.752. For comparison, EW and RW-LB had skewness of −1.715 and −0.921, respectively. RW-mdSAE also had the smallest maximum drawdown of 0.369 compared to 0.451 and 0.494 of EW and RW-LB, respectively. In fact, all three machine learning-based return-weighted strategies have, in general, shown superior performance to the baseline strategy of RW-LB and all traditionally weighted strategies shown in the Table.

Table 5 summarises the amount of outperformance provided by our proposed strategy. For the S&P 500 data set, we used 63–63–32–16 architecture with tanh function as the activation function for SAE, and 63–32–16–16 architecture with ReLU function as the activation function with noise level set to 0.001 for mdSAE.

Table 4. Performance of trend following strategies using return-weighted schemes on S&P 500 data.

	Mkt	EW	RP	RW-LB	-SVR	-SAE	-mdSAE
Annualised Return	0.035	0.097	0.093	0.118	0.135	0.131	0.139
Annualised Vol.	0.146	0.142	0.127	0.179	0.179	0.169	0.167
Average Return	0.004	0.009	0.008	0.011	0.012	0.012	0.012
t-statistic	1.338	3.149	3.322	3.108	3.471	3.549	3.770
Maximum Return	0.104	0.102	0.089	0.203	0.200	0.148	0.144
Minimum Return	−0.252	−0.269	−0.250	−0.293	−0.273	−0.286	−0.244
Sharpe Ratio	0.236	0.681	0.730	0.659	0.753	0.775	0.834
Treynor Ratio	0.035	0.110	0.120	0.124	0.138	0.139	0.158
Info. Ratio	0.000	0.927	0.773	0.711	0.892	0.948	0.912
Kurtosis	5.377	8.672	10.296	5.281	4.207	5.399	3.260
Skewness	−1.326	−1.715	−1.940	−0.921	−0.752	−1.091	−0.836
MDD	0.490	0.451	0.428	0.494	0.597	0.536	0.369

Table 5. Outperformance of RW-mdSAE in S&P 500 data.

	* − Mkt	* − EW	* − RW-LB
Annualised Return	0.104	0.042	0.021
Sharpe Ratio	0.598	0.153	0.175
Treynor Ratio	0.123	0.048	0.034

Table 6. Performance of trend following strategies using return-weighted schemes on FTSE 100 data.

	Mkt	EW	RP	RW-LB	-SVR	-SAE	-mdSAE
Annualised Return	0.030	0.075	0.071	0.114	0.112	0.099	0.129
Annualised Vol.	0.152	0.138	0.131	0.174	0.162	0.179	0.170
Average Return	0.003	0.007	0.006	0.010	0.010	0.009	0.011
t-statistic	1.078	2.313	2.300	2.781	2.878	2.424	3.118
Maximum Return	0.138	0.125	0.110	0.247	0.229	0.253	0.213
Minimum Return	−0.194	−0.145	−0.141	−0.143	−0.127	−0.205	−0.162
Sharpe Ratio	0.201	0.538	0.538	0.656	0.688	0.553	0.754
Treynor Ratio	0.030	0.092	0.091	0.135	0.141	0.105	0.148
Info. Ratio	0.000	0.542	0.495	0.647	0.646	0.609	0.835
Kurtosis	3.730	1.596	1.479	2.849	3.034	5.466	3.200
Skewness	−0.688	−0.449	−0.540	0.155	0.451	0.086	0.654
MDD	0.465	0.415	0.425	0.406	0.344	0.433	0.227

Table 6 of the FTSE 100 data set shows that, with respect to annualised return, RW-mdSAE outperforms the baseline strategy of RW-LB, however, the other two machine learning-based strategies do not. With respect to Sharpe ratio, all machine learning-based strategies, among which RW-mdSAE is the best, outperform RP and RW-LB. In general, with respect to all performance measures, RW-mdSAE outperforms the baseline strategy of RW-LB and the traditionally weighted strategies. However, for the other two machine learning-based strategies, this was not the case where only one of RW-SVR and RW-SAE produced superior result to the baseline strategy. Note that all return-weighted strategies produced positive skewness in contrast to all traditionally weighted strategies that produced negative skewness. Table 7 gives a summary of the amount of outperformance provided by our proposed strategy. For the FTSE 100 data set, 63–32–32–16 architecture with tanh function as the activation function was used for SAE, and 63–63–32–32 architecture with ReLU function as the activation function with noise level set to 0.1 was used for mdSAE.

Table 7. Outperformance of RW-mdSAE in FTSE 100 data.

	* − Mkt	* − EW	* − RW-LB
Annualised Return	0.099	0.054	0.015
Sharpe Ratio	0.553	0.216	0.098
Treynor Ratio	0.118	0.056	0.013

Table 8. Performance of trend following strategies using return-weighted schemes on KOSPI 200 data.

	Mkt	EW	RP	RW-LB	-SVR	-SAE	-mdSAE
Annualised Return	0.084	0.183	0.180	0.265	0.230	0.267	0.279
Annualised Vol.	0.198	0.216	0.202	0.291	0.269	0.269	0.302
Average Return	0.008	0.016	0.016	0.023	0.020	0.023	0.024
t-statistic	2.021	3.534	3.655	3.807	3.588	4.033	3.832
Maximum Return	0.199	0.210	0.244	0.286	0.274	0.284	0.422
Minimum Return	−0.316	−0.296	−0.277	−0.359	−0.265	−0.255	−0.317
Sharpe Ratio	0.423	0.847	0.888	0.912	0.854	0.993	0.922
Treynor Ratio	0.084	0.205	0.221	0.242	0.239	0.276	0.260
Info. Ratio	0.000	0.697	0.632	0.889	0.694	0.887	0.849
Kurtosis	5.315	3.641	4.465	2.421	1.618	2.045	3.366
Skewness	−0.883	−0.363	−0.126	−0.026	0.286	0.425	0.525
MDD	0.475	0.483	0.448	0.528	0.447	0.360	0.565

Table 9. Outperformance of RW-mdSAE in KOSPI 200 data.

	* − Mkt	* − EW	* − RW-LB
Annualised Return	0.195	0.096	0.014
Sharpe Ratio	0.499	0.075	0.010
Treynor Ratio	0.176	0.055	0.018

Lastly, Tables 8 and 9 display the results for the KOSPI 200 data set which show a similar trend as in Tables 6 and 7, respectively. Broadly speaking, RW-mdSAE outperforms the baseline strategy of RW-LB in most of the performance measures, however, RW-SVR and RW-SAE do not necessarily perform as well. On the other hand, for this data set, the deep learning-based RW-SAE seemed to perform consistently better than the shallow learning-based RW-SVR. For the KOSPI 200 data set, 63–63–32–32 architecture with tanh function as the activation function was used for SAE, and 63–32–32–16 architecture with ReLU function as the activation function with noise level set to 0.001 was used for mdSAE.

Next, we present the cumulative return curves in Figures 6 ∼ 8 for the three data sets, respectively. As in previously shown figures of Section 2, each figure contains two subfigures in which the top one spans the entire coverage period and the bottom one spans the period around the recent

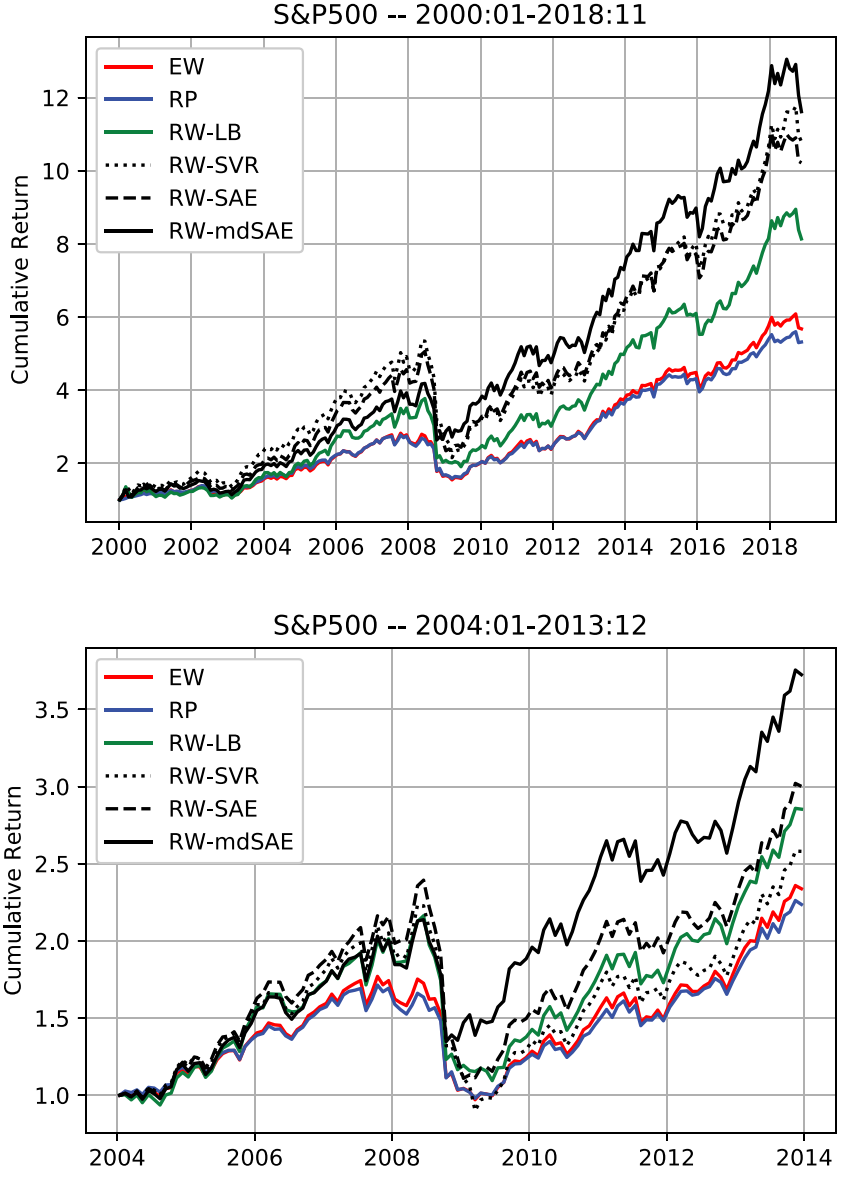

Figure 6. Cumulative return curves of return-weighted trend following strategies for S&P 500 data.

global financial crisis. The cumulative returns across the three data sets reveal that RW-mdSAE is distinctively the best asset allocation scheme amongst all asset allocation schemes considered in this paper. Figure 6 shows the curves for the S&P 500 data set and it is shown that both deep learning-based schemes of RW-SAE and RW-mdSAE clearly outperform the baseline strategy RW-LB in both subfigures. In Figures 7 and 8, which show the curves for the FTSE 100 and KOSPI 200 data sets, respectively, only RW-mdSAE outperforms the baseline strategy in all of the subfigures.

Finally, in Figure 9, we illustrate the portfolio sizes of RW-LB and RW-mdSAE as a function of time for the three data sets. Recall that the portfolio size of RW-LB is the number of securities that had positive return in the preceding one-year period, and therefore one can observe in the top subfigure for the S&P 500 data set the precipitous decline and ascent of the curve during and immediately after the global financial crisis, respectively. The RW-mdSAE portfolio is a subset of the RW-LB portfolio in

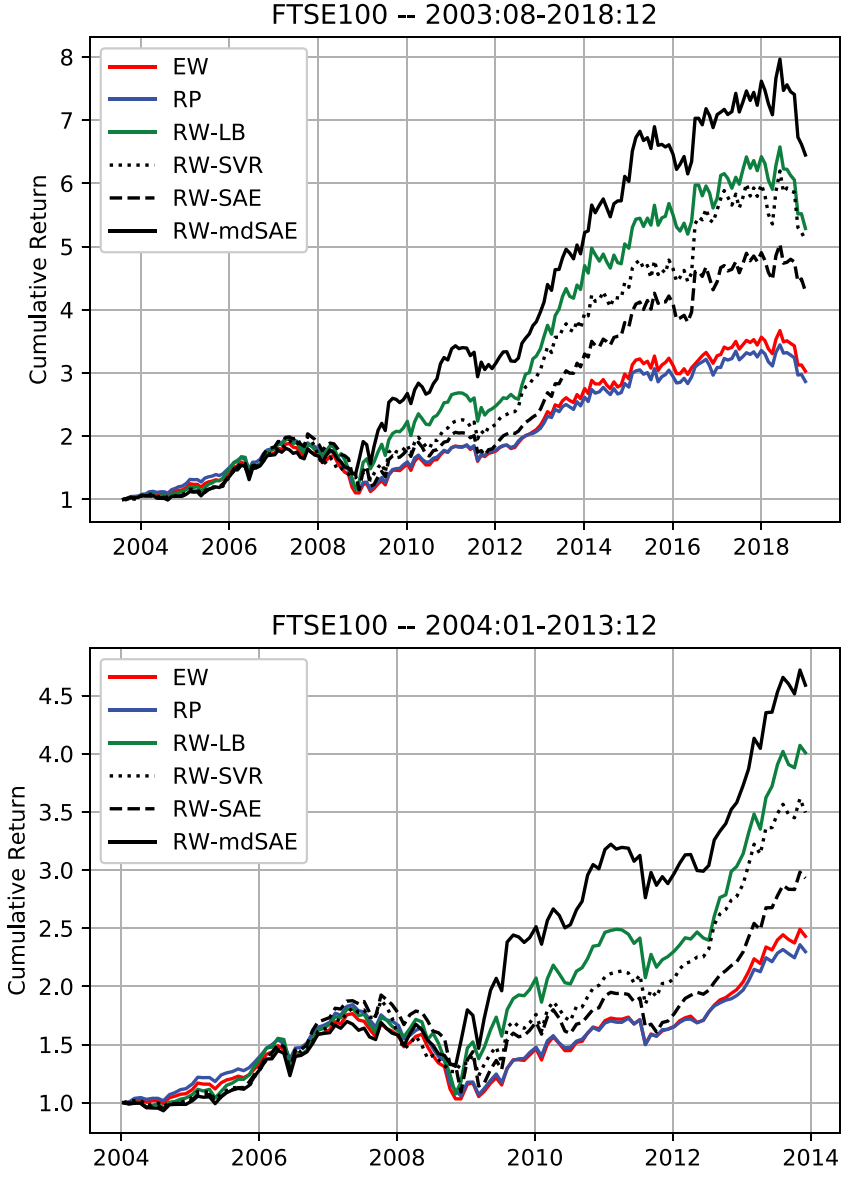

Figure 7. Cumulative return curves of return-weighted trend following strategies for FTSE 100 data.

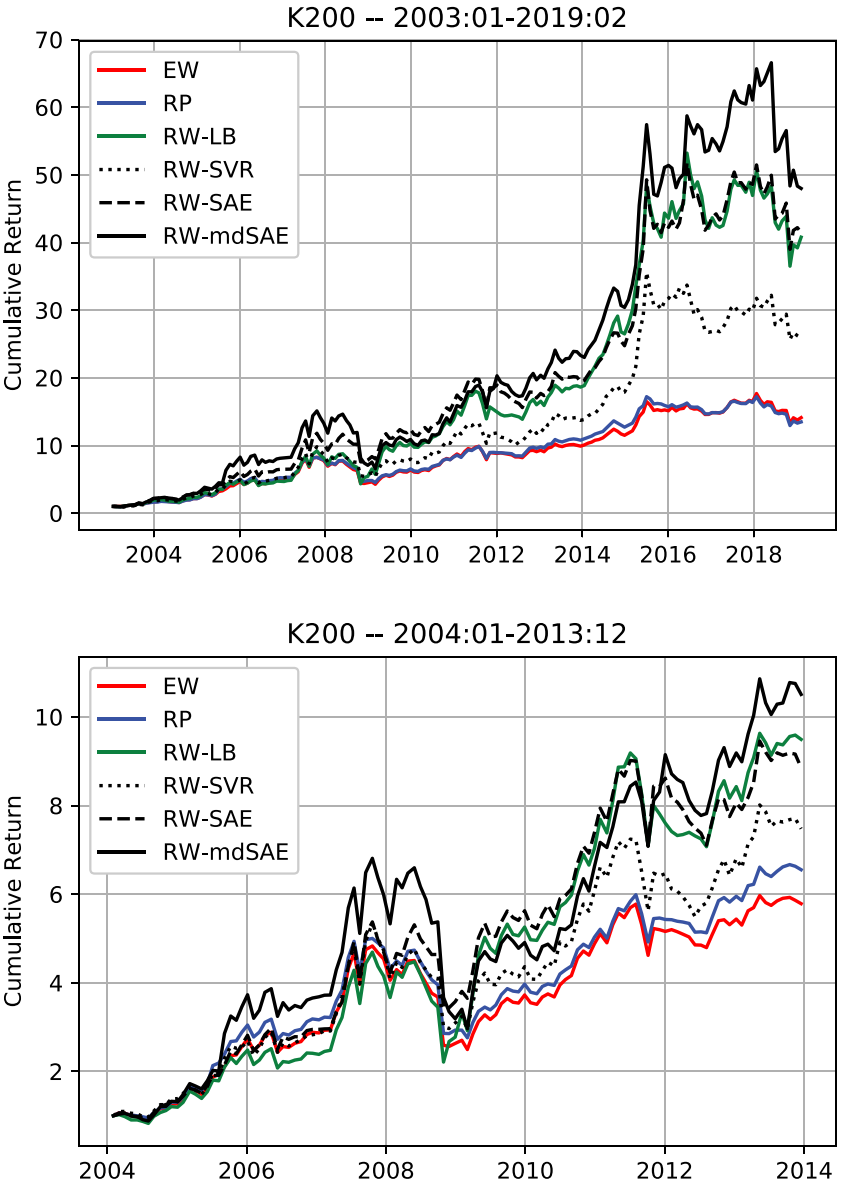

Figure 8. Cumulative return curves of return-weighted trend following strategies for KOSPI 200 data.

which the constituents are additionally required to satisfy positive predicted one-month return. For this reason, the RW-mdSAE curve always lies below the RW-LB curve, mostly by nontrivial amount across all three data sets, and this can translate to further advantage in smaller transaction costs for our proposed RW-mdSAE strategy.

Conclusion

In this paper, we have considered trend following investing of equities evaluated through a diverse set of asset allocation strategies. Our work has contributed to this topic of asset allocation in trend following investing by establishing a return-weighted strategy that has

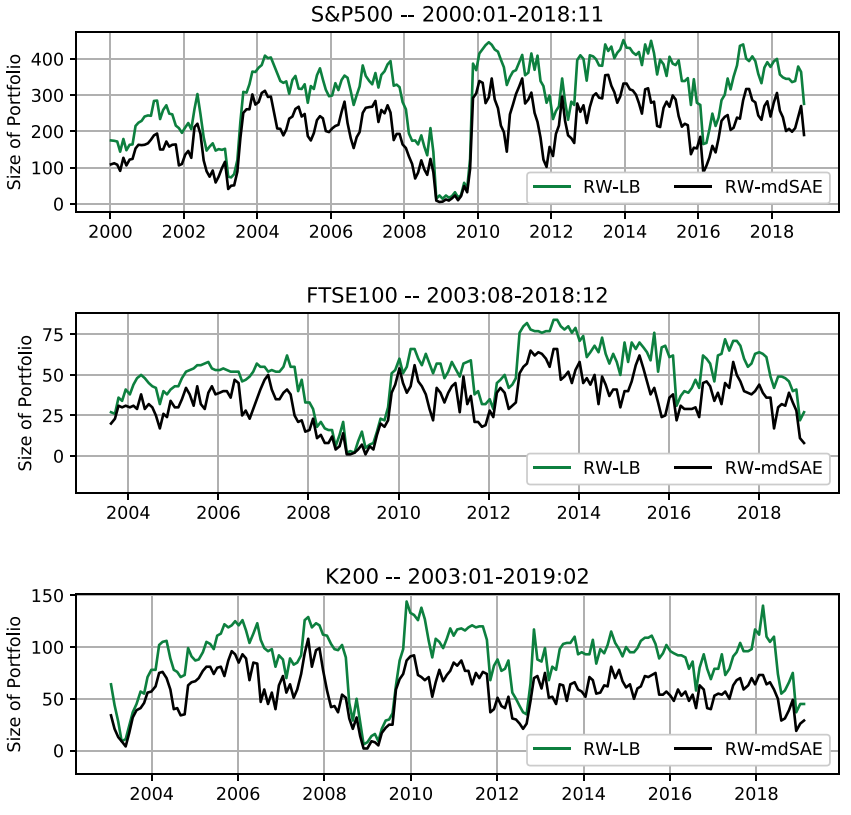

Figure 9. Portfolio sizes of RW-LB and RW-mdSAE strategies.

provided significant benefits in comparison to the traditionally weighted ones such as the equal-weighted, equal volatility-weighted and risk parity. More specifically, the return-weighted asset allocation strategy using assets' realised returns in the previous one-year period has generated abnormal absolute return advantage compared to the traditionally weighted ones, and this did not come at the cost of higher risk as similar advantage in Sharpe ratio was also observed. Our empirical findings have shown that our proposed strategy generates favourable performance statistics gauged through various measures that were persistent over time and pervasive across different markets.

In the second part of our contribution, motivated by the immense benefit that can be achieved if the actual return of the holding period can be served for the return-weighted asset allocation strategy, we have established deep learning architectures designed specifically for the regression of returns in this setting. Our findings indicate that, amongst a number of qualified machine learning architectures, only a latest deep learning architecture, namely, the marginalised denoising stacked autoencoder, beats the realised return-weighted strategy consistently over a wide range of economic periods in various global markets. As a result, the return-weighted asset allocation strategy defined by our proposed deep learning architecture produces substantial abnormal returns advantage over all traditionally weighted strategies for trend following investing. Our work differs from previous works on return regression using deep learning, apart from numerous implementational details, in the adoption of the aforementioned latest deep learning architecture in conjunction with SVR in the last regression layer. For future works, we plan to investigate the identification of the set of invariant and disentangled deep

features for the characterisation of underlying drivers of improved return regression.

Acknowledgments

This work was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2017R1D1A1B03032722); and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2020R1F1A1A0106053811).

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was supported by the National Research Foundation of Korea [2020R1F1A1A0106053811,NRF-2017R1D1A1B03032722].

References

Asness, C. S., Moskowitz, T. J., & Pederson, L. H. (2013). Value and momentum everywhere. *Journal of Finance*, *68*(4), 929–985. https://doi.org/10.1111/jofi.12021

Babu, A., Levine, A., Ooi, Y. H., Pedersen, L. H., & Stamelos, E. (2020). Trends everywhere. *Journal of Investment Management*, *18*(1), 52–68.

Bai, X., Scheinberg, K., & Tutuncu, R. (2016). Least-squares approach to risk parity in portfolio selection. *Quantitative Finance*, *16*(3), 357–376. https://doi.org/10.1080/14697688.2015.1031815

Baltas, N., & Kosowski, R. (2017). Demystifying time-series momentum strategies: Volatility estimators, trading rules and pairwise correlations. In *SSRN* (pp. 2140091).

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

Barroso, P., & Santa-Clara, P. (2015). Momentum has its moments. *Journal of Financial Economics*, *116*(1), 111–120. https://doi.org/10.1016/j.jfineco.2014.11.010

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation Learning: A review and new perspectives. *IEEE, Trans. On Pattern Analysis and Machine Intelligence*, *35*(8), 1798–1828. https://doi.org/10.1109/TPAMI.2013.50

Bengio, Y., Lamblin, P., Popovici, D., & Larochelle, H. (2006). Greedy layer-wise training of deep networks. In *Neural information processing systems* edited by Scholkopf, B. and Platt, J.C, (pp. 153–160).

Carhart, M. (1997). On the persistence in mutual fund performance. *Journal of Finance*, *52*(1), 57–82. https://doi.org/10.1111/j.1540-6261.1997.tb03808.x

Chen, M., Weinberger, K., Sha, F., & Bengio, Y. (2014) Marginalized denoising auto-encoder for nonlinear representations. *International Conference on Machine Learning*, Beijing, China.

Daniel, K., & Moskowitz, T. J. (2016). Momentum crashes. *Journal of Financial Economics*, *122*(2), 221–247. https://doi.org/10.1016/j.jfineco.2015.12.002

DeMiguel, V., Garlappi, L., & Uppal, R. (2009). Optimal versus naive diversification: How inefficient is the 1/N portfolio strategy? *Review of Financial Studies*, *22*(5), 1915–1953. https://doi.org/10.1093/rfs/hhm075

Ernst, P., Thompson, J., & Miao, Y. (2017). Portfolio selection: The power of equal weight. In *arxiv.org/pdf/1602.00782*.

Fama, E. F., & French, K. R. (1996). Multifactor explanations of asset pricing anomalies. *Journal of Finance*, *51*(1), 55–84. https://doi.org/10.1111/j.1540-6261.1996.tb05202.x

Fama, E. F., & French, K. R. (2012). Size, value, and momentum in international stock returns. *Journal of Financial Economics*, *105*(3), 457–472. https://doi.org/10.1016/j.jfineco.2012.05.011

Glorot, X., Bordes, A., & Bengio, Y. (2011) Domain adaptation for large-scale sentiment classification: A deep learning approach. *International Conference on Machine Learning*, Bellevue, WA, USA.

Grundy, B., & Martin, J. S. (2001). Understanding the nature of the risks and the source of the rewards to momentum investing. *Review of Financial Studies*, *14*(1), 29–78. https://doi.org/10.1093/rfs/14.1.29

He, X.-Z., & Li, K. (2015). Profitability of time series momentum. *Journal of Banking and Finance*, *53*, 140–157. https://doi.org/10.1016/j.jbankfin.2014.12.017

Hsu, P.-H., Han, Q., Wu, W., & Cao, Z. (2018). Asset allocation strategies, data snooping, and the 1/N rule. *Journal of Banking and Finance*, *97*, 257–269. https://doi.org/10.1016/j.jbankfin.2018.09.021

Hurst, B., Ooi, Y. H., & Pedersen, L. H. (2013). Demystifying Managed Futures. *Journal of Investment Management*, *11*(3), 42–58.

Hurst, B., Ooi, Y. H., & Pedersen, L. H. A century of evidence on trend-following investing. (2017). *Journal of Portfolio Management*, *44*(1), 15–29. Fall. https://doi.org/10.3905/jpm.2017.44.1.015

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

Jegadeesh, N., & Titman, S. (2001). Profitability of momentum strategies: An evaluation of alternative explanations. *Journal of Finance*, *56*(2), 699–720. https://doi.org/10.1111/0022-1082.00342

Kim, S. (2018). Volatility forecasting for low-volatility portfolio selection in the US and the Korean equity markets. *Journal of Experimental and Theoretical Artificial Intelligence*, *30*(1), 71–88. https://doi.org/10.1080/0952813X.2017.1354083

Kim, S. (2019). Enhancing the momentum strategy through deep regression. *Quantitative Finance*, *19*(7), 1121–1133. https://doi.org/10.1080/14697688.2018.1563707

Kim, S., & Heo, J. (2017). Time series regression-based pairs trading in the Korean equities market. *Journal of Experimental and Theoretical Artificial Intelligence*, *29*(4), 755–768. https://doi.org/10.1080/0952813X.2016.1259265

Kim, S., & Kim, S. (2020). Index tracking through deep latent representation learning. *Quantitative Finance*, *20*(4), 639–652. https://doi.org/10.1080/14697688.2019.1683599

Kingma, D. P., & Ba, J. (2015) Adam: A method for stochastic optimization. *International Conference on Learning Representations*. San Diego, USA.

Law, T., & Shawe-Taylor, J. (2017). Practical Bayesian support vector regression for financial time series prediction and market condition change detection. *Quantitative Finance*, *17*(9), 1403–1416. https://doi.org/10.1080/14697688.2016.1267868

Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep Learning. *Nature*, *521*, 436–444.

Lee, H., Largman, Y., Pham, P., & Ng, A. Y. (2009). Unsupervised feature learning for audio classification using convolutional deep belief networks. In *Neural information processing systems*, pp. 1096–1104.

Lim, B. Y., Wang, J., & Yao, Y. (2018). Time-series momentum in nearly 100 years of stock returns. *Journal of Banking and Finance*, *97*, 283–296. https://doi.org/10.1016/j.jbankfin.2018.10.010

Maillard, S., Roncalli, T., & Teiletche, J. (2010). The properties of equally weighted risk contribution portfolios. *Journal of Portfolio Management*, *36*(4), 60–70. https://doi.org/10.3905/jpm.2010.36.4.060

Malladi, R., & Fabozzi, F. J. (2017). Equal-weighted strategy: Why it outperforms value-weighted strategies? Theory and evidence. *Journal of Asset Management*, *18*(3), 188–208. https://doi.org/10.1057/s41260-016-0033-4

McLean, D., & Pontiff, J. (2016). Does academic research destroy stock return predictability? *Journal of Finance*, *71*(3), 5–31. https://doi.org/10.1111/jofi.12365

Michaud, R. O. (1989, January-February). The markowitz optimization enigma: Is 'Optimized' optimal? *Financial Analysts Journal*, *45*(1), 31–42. https://doi.org/10.2469/faj.v45.n1.31

Moskowitz, T. J., Ooi, Y. H., & Pedersen, L. H. (2012). Time series momentum. *Journal of Financial Economics*, *104*(104), 228–250. https://doi.org/10.1016/j.jfineco.2011.11.003

Qian, E. (2011). Risk parity and diversification. *The Journal of Investing*, *20*(1), 119–127. https://doi.org/10.3905/joi.2011.20.1.119

Sapankevych, N. I., & Sankar, R. (2009). Time series prediction using support vector machines: A survey. *IEEE Computational Intelligence Magazine*, *4*(2), 24–38. https://doi.org/10.1109/MCI.2009.932254

Shen, F., Chao, J., & Zhao, J. (2015). Forecasting exchange rate using deep belief networks and conjugate gradient method. *Neurocomputing*, *167*, 243–253. https://doi.org/10.1016/j.neucom.2015.04.071

Smith, J., & Wallis, K. F. (2009). A simple explanation of the forecast combination puzzle. *Oxford Bulletin of Economics and Statistics*, *71*(3), 331–355. https://doi.org/10.1111/j.1468-0084.2008.00541.x

Vincent, P., Larochelle, H., Lajole, I., Bengio, Y., & Manzagol, P.-A. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, (11), 3371–3408.

Yang, K., Qian, E., & Belton, B. (2019, July). Protecting the downside of trend when it is not your friend. *Journal of Portfolio Management*, 1–13.