Forecasting Realized Volatility: A Hybrid Model Integrating BiLSTM with HAR-type Models

Sherry Luo[1] and Marwan Izzeldin[2]

[1] *Department of Accounting, Xi'an Jiaotong-Liverpool University, Email: Yi.Luo@xjtlu.edu.cn*
[2] *Department of Economics, Lancaster University*

Abstract

This paper proposes a hybrid methodology that combines both Heterogeneous Autoregressive (HAR)-type models and Deep Feedforward Neural Network (DFN) model as well as Bidirectional Long Short-term Memory (BiLSTM) model in predicting realized volatility. On the one hand, Neural Network architecture naturally deals with many important stylized facts of realized volatility (e.g., long-memory and nonlinearity etc), complementing the linear HAR-type models. On the other hand, the interpretation of results produced by Neural Networks can be improved from the aspect of high-quality input features generated by HAR-type models. Empirical results show that BiLSTM-based hybrid model outperforms all other models in the out-of-sample forecasting across all forecasting horizons. Additionally, both the performance of DFN-based and BiLSTM-based hybrid model significantly beat their single-model counterparts, indicating HAR-type components can be considered as effective features in Neural Networks.

JEL classification: C14, C58, G17, G32
Keywords: Realized Volatility, Neural Networks, Heterogeneous Autoregressive Model, Volatility Forecasting

1. Introduction

Accurate volatility forecasting is critical for various financial applications such as asset pricing, asset allocation and risk management. The key task of financial volatility modeling is to model the stylized facts of financial returns and volatility. Since the early work of Engle (1982) and Bollerslev (1986), Generalized AutoRegressive Conditional Heteroskedasticity (GARCH) family models have been extended to capture many important stylized facts such as leverage effects, long memory, leptokurtosis and volatility clustering. However, for GARCH-type models, problems arise in distributional assumption, estimation complexity and economic interpretation of estimated parameters. In addition, the failure to account nonlinearity distorts the definition of data generating process and imposes great challenges on model estimation. With the substantial development of computational resources, the Artificial Neural Network models (ANNs) have been extensively used to predict stock return volatility, showing superior forecasting performance against Financial Time Series models (see, e.g., Tino et al. (2001), Hamid and Iqbal (2004), Khan (2011), Nikolaev et al. (2013), Xiong et al. (2015), Doering et al. (2017), Zhou et al. (2019), Bucci (2020), Horvath et al. (2021)). One of the advantages of this approach is that it can well approximate a wide class of linear and nonlinear functions, forming the input-output map by learning the data rather than assuming the data generating process which complements the limitations of traditional Financial Time Series models.

However, ANNs have been criticized for the "black box" problem which makes the results interpretation very difficult since it does not provide any insights about the resulted input-output map (Lai et al. (2009)). To improve the interpretability of Neural Network models, many studies suggest hybrid modeling, integrating Financial Time Series models with ANNs. Donaldson and Kamstra (1997) develop seminonparametric GARCH-type models which uses ANN transformation to capture nonlinear relationship between past return innovations and future volatility. The logistic function, the most popular nonlinear transformation function used in the ANN literature, is applied to approximate the nonlinearities. The empirical results show that the GARCH-type models using logistic transformation generally outperform other alternatives in forecasting volatility of four stock market index. Another type of hybrid modeling is purely nonparametric approach through the usage of more informative input features (see, e.g., Sjöberg et al. (1995), Faraway and Chatfield (1998), Nanopoulos et al. (2001), Dash et al. (2007), Zheng et al. (2016), Chen et al. (2016), Petneházi (2019), Hewamalage et al. (2021)). Features generated by Financial Time Series models have become increasingly popular given their explicit economic interpretation and successful reproduction of important stylized facts of financial volatility. For example, Roh (2007) propose hybrid

models integrating Deep Feedforward Neural Network (DFN) with GARCH, EWMA and EGARCH model respectively. Rather than using only past volatility as input for DFN, features generated from those GARCH-type models provide valuable information on the characteristics of stock market volatility such as leverage effects, volatility clustering and excessive kurtosis. The empirical results show that hybrid models significantly outperform both single GARCH-type models and single DFN in forecasting volatility of KOSPI 200 index. This indicates that the predictability and interpretability of DFN can be enhanced by integrating features generated from GARCH-type models. Roh's study has been extended by integrating other GARCH-type models with various Neural Network models in forecasting stock index volatility (see, e.g., Tseng et al. (2008), Hajizadeh et al. (2012), Kristjanpoller et al. (2014), Maciel et al. (2016), Kim and Won (2018)).

Although GARCH-type models successfully reflect some empirical characteristics of in-sample volatility, such characteristics are hard to be reproduced in the out-of-sample volatility forecasting (Andersen et al. (2003)). Moreover, inappropriate distributional assumptions on financial returns and failure of reproducing other important stylized facts (i.e., long memory) may further deteriorate the quality of features generated by GARCH-type models and integrated into the ANNs. With the development of high-frequency data, Andersen and Bollerslev (1998) derive realized volatility (RV), which is a new nonparametric measure of return volatility and is defined as the sum of intraday squared returns. Barndorff-Nielsen and Shephard (2002) show that RV is model-free and unbiased estimate of ex post return variation. RV displays long-memory feature which has been previously modeled by AutoRegressive Fractionally Integrated Moving Average (ARFIMA) process (see, e.g., Andersen et al. (2001), Andersen et al. (2003)). However, ARFIMA lacks explicit economic interpretation and is nontrivial to be estimated. To improve this, Corsi (2009) proposes Heterogeneous AutoRegressive (HAR) model which is an additive cascade model of three volatility components defined over different time horizons, i.e., daily, weekly and monthly. HAR successfully reproduces the main features of financial returns and volatility (i.e., fat tails, self-similarity and long memory). In addition, compared with GARCH-type models, HAR achieves superior performance in forecasting RV across various financial assets with less estimated parameters. It is also easy to implement and interpret. Despite RV's prominent long-memory feature, it is also found to be subject to regime-switching and structural breaks which requires modeling of nonlinear effects. The nonlinearity has imposed great challenges in modeling and forecasting RV. McAleer and Medeiros (2008) extend the standard HAR into the Heterogeneous Autoregression with Multiple-Regime Smooth Transition (HARST) model, merging long memory and nonlinearities in RV. However, such method requires huge amount of data to determine the number of regimes before estimating the model and the sample log-likelihood can easily

trap in local maxima, leading to poor out-of-sample performance (Pavlidis et al. (2012)).

To facilitate better modeling of nonlinearities, semiparametric HAR-based hybrid models integrating Neural Networks have been developed. For example, Hillebrand and Medeiros (2010) propose the Neural Network HAR model (NNHAR) to forecast RV with an additional logistic transformation term to capture the nonlinearities in RV components. Surprisingly, NNHAR only outperforms the standard log-linear HAR during low-volatility periods and such advantages are eliminated by bagging the forecasts of log-linear HAR model. This indicates that bagging may be more efficient than directly modeling the nonlinearities. Fernandes et al. (2014) extend the NNHAR model into the NNHARX model by including macroeconomic variables to forecast implied volatility (IV). The empirical results are mixed and NNHARX only beats the standard HAR in 22-day forecasting window. This indicates that the strong persistence of IV may predominates the nonlinearities over shorter forecasting horizons. Instead of using the logistic function, Psaradellis and Sermpinis (2016) pioneer to use Support Vector Machines (SVM) with Radial Basis Function (RBF) kernel for nonlinear transformation in HAR to forecast IV. Moreover, unlike the previous studies where the hyperparameters (i.e., the number of hidden nodes in the nonlinear transformation term) are selected according to authors' discretion, the value of those parameters are optimized and determined by walk-forward validation. The proposed approach consistently outperforms standard HAR and NNHAR, showing significant improvement in the out-of-sample forecasting of IV.

The puzzling results of semiparametric HAR-based hybrid models may reflect the following limitations of such approach: (1) This kind of hybrid models combines the HAR components with an extra nonlinear transformation term (i.e., the logistic transformation term). Such technique is called the cooperative modular combination and aims for better representing the different behaviors of time series (Sharkey (2002)). Regarding the application, the original HAR is first applied to the time series and then its residuals are modeled through nonlinear transformation function. However, this implicitly assumes that the residuals of HAR include prominent nonlinear patterns that can be modeled by Neural Networks (Zhang and Qi (2005)). Violation of such assumptions may deteriorate the performance of semiparametric HAR-based hybrid models. In addition, the simple additive combination may underestimate the relationship between the HAR components and the nonlinear term (Taskaya-Temizel and Casey (2005)); (2) In viewing of (1), the choice of nonlinear transformation function may also be problematic in such hybrid models. The NNHAR model which uses the logistic function hardly improves the standard HAR while the model uses RBF kernal achieves significant improvement. This indicates that the nonlinear functions are chosen by authors rather than being reasonably optimized. Different nonlinear trans-

formation functions process the input information in a very different way which may lead to undesirable outcomes; (3) The functioning of Neural Networks rely on the configuration of input layer, hidden layer and output layer. Although the hidden layer handles nonlinear transformation and creation of high-dimensional features, the design of input layer is found to be an important factor which would affect the performance of subsequent layers. The semiparametric HAR-based hybrid models only rely on the function of hidden layer (i.e., nonlinear transformation) and disregard other components of ANNs which may induce performance degeneration (Zhang and Berardi (2001), Taskaya-Temizel and Casey (2005)).

To overcome above limitations, we propose the nonparametric ANN-based hybrid models which takes various input features generated from HAR-type models. This approach differs from the semiparametric HAR-based hybrid models in the following aspects. First, the proposed model can well approximate a wide range of linear and nonlinear functions between the input features and the target output without assuming that nonlinear patterns only exist in the residuals. Second, our model consists of the input layer, hidden layer and output layer which fulfills the standard structure of ANNs. The hyperparameters of the model are reasonably optimized through validation rather than being chosen subjectively. Third, to predict the target output, the model enables learning not only from the past values but also from the future values. This allows bidirectional learning and a much longer window to capture the long-term dependence in RV.

The main contributions of this chapter are as follows:

(1) Two nonparametric ANN-based hybrid models are proposed to forecast RV on the basis of different Neural Network architectures. To allow different pattern-learning abilities, Deep Feedforward Neural Network (DFN) and Bidirectional Long Short-term Memory Network (BiLSTM) are used as the Neural Network basis in the proposed hybrid models. In addition, rather than simply adding a nonlinear transformation term to the original HAR model, we relax the assumptions used in the semiparametric HAR-based hybrid model by embracing a full Neural Network setting.

(2) Despite significant advantages of applying a full Neural Network setting, ANNs are often criticized by lacking explicit economic explanations. We improve the interpretability of the proposed hybrid models by feeding features generated by HAR-type models. The standard HAR and HAR-TCJ proposed by Corsi et al. (2010) successfully reproduce several stylized facts of RV (i.e., self-similarity, presence of jumps etc.). Moreover, the parameters of HAR-type models measure the impact of different volatility components on future RV which makes the HAR-type components more informative than past RV alone. By taking the HAR-type components as input features, our proposed hybrid models outperform all benchmarks in forecasting RV with the BiLSTM-based hybrid model being the best performer across all

forecasting horizons in all subsamples.

The remainder of this chapter is structured as follows. Section 2 introduces the structure and design of the proposed model. Section 3 presents the data and experiment. Section 4 shows the out-of-sample prediction results. Finally, Section 5 concludes.

2. Models

2.1. HAR model

Our hybrid models are based on the HAR proposed by Corsi (2009). The HAR model successfully replicates main empirical features of financial returns such as long memory, fat tails and self-similarity. It has strong empirical performance in forecasting RV across various financial assets. In this study, we consider the RV estimator proposed by Andersen and Bollerslev (1998), which is equal to the square root of the sum of intraday squared returns:

$$RV_t = \sqrt{\sum_{j=1}^{M} r_{j,t}^2} \tag{1}$$

where $r_{j,t}$ stands for the intraday returns within each time interval in day t. To introduce the HAR model, we denote the average RV over the previous (h) days by:

$$RV_t^h = \frac{1}{h} \sum_{i=1}^{h} RV_{t-i+1} \tag{2}$$

Therefore, the weekly (5-day) and monthly (22-day) RV are the corresponding averages of daily RV, which is given by $RV_t^{(w)} = \frac{1}{5} \sum_{i=1}^{5} RV_{t-i+1}$ and $RV_t^{(m)} = \frac{1}{22} \sum_{i=1}^{22} RV_{t-i+1}$ respectively. The logarithmic form of HAR is given by:

$$log(RV_{t+1}) = \beta_0 + \beta_d log(RV_t) + \beta_w log(RV_t^{(w)}) + \beta_m log(RV_t^{(m)}) + \epsilon_{t+1} \tag{3}$$

where ϵ_{t+1} is a sequence of independently and identically distributed (i.i.d) innovations with zero mean. Coefficient estimates, β_0, β_d, β_w and β_m can be consistently obtained by a standard OLS regression. The HAR component, $\beta_d log(RV_t)$, $\beta_w log(RV_t^{(w)})$ and $\beta_m log(RV_t^{(m)})$, reflect the magnitude of the heterogeneous reaction of different market participants to a given price change. The Autocorrelation Function (ACF) plot and Partial Autocorrelation Function (PACF) plot for SPY's RV are provided in Figure 3.

Although HAR has very strong empirical performance in predicting RV, it fails to capture

any nonlinearity in the RV dynamics. As a result, some researchers suggested to use smooth transition or threshold model to estimate RV (McAleer and Medeiros (2008)). However, this class of models generally requires substantial amount of data to identify the states and shows poor out-of-sample forecasting results (Bucci (2020)).

2.2. HAR-TCJ model

We also consider an extension of the HAR, the HAR-TCJ model proposed by Corsi et al. (2010). By implementing a more robust test to detect jumps, the HAR-TCJ builds on the HAR-CJ model in Andersen et al. (2007) and better incorporates jumps in the estimation of volatility models. The logarithmic form of HAR-TCJ is defined as:

$$log(RV_{t+1}) = \beta_0 + \beta_{c,d} log(\hat{TC}_t) + \beta_{c,w} log(\hat{TC}_t^{(w)}) + \beta_{c,m} log(\hat{TC}_t^{(m)}) + \beta_{j,d} log(1 + \hat{TJ}_t) + \epsilon_{t+1} \quad (4)$$

where $\hat{TC}_t^{(w)}$ and $\hat{TC}_t^{(m)}$ are the weekly and monthly averages of $\hat{TC}_t$ as in Eq.

The jump component is estimated based on the Threshold Bi-power variation (TBPV) measure given by:

$$\hat{TJ}_t = I_{\{C-Tz>\Phi_\alpha\}}(RV_t - TBPV_t)^+ \quad (5)$$

where $C - Tz$ is the test statistics of the jump test based on the corrected TBPV and Φ_α is the cumulative distribution function of the normal distribution at level $\alpha = 99.9\%$, see more details in Corsi et al. (2010). The TBPV measure is derived as:

$$TBPV_t = \mu^{-2} \sum_{j=2}^{M} |r_{t,j-1}||r_{t,j}| I_{\{r_{t,j-1}^2 \leqslant v_{j-1}\}} I_{\{r_{t,j}^2 \leqslant v_j\}} \quad (6)$$

with $\mu = (2/\pi)^{0.5}$. The threshold v_t is equal to $c_v^2 \hat{V}_t$ with $\hat{V}_t$ being the local variance estimator and $c_v = 3$ suggested by Corsi et al. (2010). The continuous part of variation is therefore given by:

$$\hat{TC}_t = RV_t - \hat{TJ}_t \quad (7)$$

The HAR-TCJ components, $\beta_{c,d} log(\hat{TC}_t)$, $\beta_{c,w} log(\hat{TC}_t^{(w)})$, $\beta_{c,m} log(\hat{TC}_t^{(m)})$ and $\beta_{j,d} log(1 + \hat{TJ}_t)$, provide more detailed features of realized volatility by accommodating the robust jump effects.

2.3. Neural Network models

ANNs are powerful nonparametric tools, mimicking the structure of the human brain and aiming for modeling and predicting the unobserved function underlying the observed data (Arnerić et al. (2014)). Empirical research suggests that ANNs are particularly suitable for forecasting Financial Time Series that exhibit nonlinear behaviors, like stock market volatility or stock market returns (Maheu and McCurdy (2002)), by learning and mapping nonlinear structure that linear model cannot process. In this way, Neural Network models can be implemented without any assumptions on the underlying data generating process.

The learning ability of ANNs can be improved by increasing the number of hidden layers and hidden nodes. However, the training algorithm can be very slow and the out-of-sample regularization can be poor when the entire network becomes too complicated. Some rules have been proposed in the literature to find the optimal number of hidden layers and hidden nodes so that the good learning performance can be generalized to the testing sample (Sheela and Deepa (2013)). However, there is no uniform solution to this issue. Stinchcombe and White (1992) prove that Neural Network with a single hidden layer and sufficient hidden nodes is a universal approximator, meaning that such network is sufficient in approximating a wide range of linear and nonlinear functions. Following Bucci (2020), a single hidden layer Neural Network is assumed throughout this chapter, leaving the number of hidden nodes to be determined separately.

To allow different pattern learning behavior, two representative Neural Network models, DFN and BiLSTM, are used as Neural Network basis in the proposed hybrid models. DFN is one of the earliest developed ANNs and it consists of input layer, hidden layer and output layer. Fixed amount of information is input into DFN at time t to predict the target variable value at $t+1$. Clearly, DFN is a short-memory model as it does not capture any possible long-range dependence in the data. The workings of BiLSTM, however, relies on memory cells which controls the amount of information flow through the input gate, forget gate and output gate. BiLSTM takes input at each time point until time t and only the important information will be processed into the next cell state. Therefore, BiLSTM captures both short-term and long-term memory in the data. These two Neural Network basis are further enhanced by the proposed hybrid models in the following section.

2.3.1. Deep Feedforward Neural Network model

DFN is a classical deep learning model. It assumes that information moves forward from the input layer to the output layer and uses the output of the previous layer as the input of the current layer. The calculation continues until the output layer is reached. The output

of a typical DFN with single hidden layer is given by:

$$\hat{RV}_t = F(\beta_0 + \sum_{j=1}^{q} f(x_t w_j')\beta_j), \tag{8}$$

where F is the activation function of the output layer, β_0 is the bias to be added to the final output, f is the activation function of hidden layer, $x_t = \{1, x_{1,t}, ..., x_{i,t}\}$ is the $1 * (1 + 1)$ vector of features at time t, $w_j = \{w_{1,j}, ..., w_{i+1,j}\}$ is the $1*(i+1)$ vector of weights, measuring the connection between the features and hidden node j, total number of hidden nodes is q.

F and f can be chosen from a variety of functions in the empirical research. Given the nature of forecasting a real number (RV) in this study, we choose the identity function as the activation function of the output layer, that is, $F(a) = a$. In this sense, Eq.(8) can be written as:

$$\hat{RV}_t = \beta_0 + \sum_{j=1}^{q} f(x_t w_j')\beta_j. \tag{9}$$

For the activation function of the hidden layer f, any continuous, nonlinear, differentiable and monotonic function can be applied. Common choices are sigmoid function and hyperbolic tangent function. However, these two functions suffer from the vanishing gradient problem and are generally slow in computing. Therefore, we choose the RELU function, that is, $f(a) = max(0, a)$, to allow faster learning and better performance.

2.3.2. Bidirectional Long Short-term Memory Network model

DFN is perceived by researchers as static network given the fixed amount of information used to predict the target output variable at time t. Although DFN may be able to capture the temporal dependence contained in the data, it has no memory. The Long Short-term Memory (LSTM) Neural Network, a representative recurrent Neural Network model proposed by Hochreiter and Schmidhuber (1997), is proposed to overcome this problem by allowing information feedback. Instead of using fixed amount of information as does in DFN, LSTM utilizes all available input information up to the time t to predict the target output. It becomes extremely popular in predicting financial time series where nonlinear time dependence and long memory are prominent (see, e.g., Schittenkopf et al. (2000), Tino et al. (2001), Bucci (2020)). Compared with DFN, LSTM is characterized by using memory cells instead of hidden units to process information. Figure 1 shows the structure of a LSTM memory cell.

LSTM controls information flow through the functioning of three gates on the cell state c_t: an input gate (i_t), a forget gate (f_t) and an output gate (o_t). Each gate performs different

roles, as follows.

The input gate i_t controls the extent of information to be added to the cell, where

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \tag{10}$$

The forget gate f_t controls the extend of information to be discarded from the cell, where

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \tag{11}$$

The output gate o_t controls the extent of output information from the cell, where

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \tag{12}$$

and σ_g is the gate activation function which is by default the sigmoid function. The response of the gate activation function is between 0 and 1, controlling the amount of information flow in that gate.

A nonlinear function, the hyperbolic tangent function ($tanh$), is applied to generate a vector of candidate values, $\tilde{c}_t$, to update the cell state c_t as follows:

$$\tilde{c}_t = tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{13}$$

The current cell state c_t is now updated by the operation of input gate and forget gate, as follows:

$$c_t = i_t \odot \tilde{c}_t + f_t \odot c_{t-1} \tag{14}$$

The output of LSTM memory cell at time t is then given by:

$$h_t = o_t \odot tanh(c_t) \tag{15}$$

where $\odot$ is the Hadamard production function.

Finally, the memory cell output h_t is projected to obtain the predicted output $\hat{RV}_t$, which is,

$$\hat{RV}_t = W_t h_t \tag{16}$$

where W_t is a projection matrix used to reduce the dimension of h_t.

However, for the highly time-dependent financial time series, future input information coming up later than time t can also be helpful in predicting the target value at time t. The Bidirectional Long Short-term Memory (BiLSTM) Neural Network is therefore proposed by Schuster and Paliwal (1997) to include both the past and the future input information of

a specific time frame to predict the target output. BiLSTM computes the forward output sequence $\overrightarrow{h}$ using input information from time $t = 1$ to T and the backward output sequence $\overleftarrow{h}$ using the information from time $t = T$ to 1. Fig presents the structure of BiLSTM Neural Network. Both the forward and the backward output sequence are calculated using the same LSTM updating equations specified in Eq.(10) - (15). The predicted output $\hat{RV}_t$ is now given by:

$$RV_t = \sigma(\overrightarrow{h}, \overleftarrow{h}) \tag{17}$$

where σ is a function merging the forward and the backward output sequences. It can be a summation function, a multiplication function, a concatenating function or an average function. We choose the multiplication function for σ as it achieves the minimum validation loss discussed below.

As the above DFN, we use a single BiLSTM layer for the hidden layer and the number of hidden units is optimized separately. In addition, recurrent layer dropout regularization is applied to prevent overfitting. The BiLSTM is implemented using the "keras" package in R. Figure 2 presents the basic structure of DFN, LSTM and BiLSTM.

[INSERT FIGURE 2 ABOUT HERE]

2.4. Proposed hybrid models - BiLSTM with HAR-type models

Financial time series models such as GARCH-type models haven been integrated with different Neural Network models to predict financial volatility. Empirical results show that adding GARCH-type components significantly improve the volatility forecasts under the Neural Network structure, indicating those components are good feature representations of volatility. However, standard GARCH fails to reproduce the long-memory property which is an important stylized facts of financial returns. Extension of standard GARCH such as FIGARCH uses fractional difference operators to obtain long-memory volatility but it is difficult to estimate and lacks explicit economic interpretation (Corsi (2009)). Such problems may negatively affect the interpretability of forecasting results generated by Neural Network models if those components are fed into Neural Networks as features.

HAR model is proposed by Corsi (2009) to address above issues of GARCH-type models and significantly improves volatility forecasting by successfully reproducing the main stylized facts of financial returns (i.e., self-similarity, long-memory and fat tails). This indicates that volatility is better represented under the HAR structure. It also maintains a simple structure and its parameters have clear economic interpretation. The standard HAR model is then extended by incorporating jump component in forecasting volatility. Corsi et al. (2010) propose a new jump test based on the Threshold Bi-power variation and decompose the

volatility into a continuous part and a jump part. Empirical results show that jumps have significant impact in forecasting future volatility, suggesting jumps serve to an important feature of financial volatility and volatility is better modelled by such decomposition.

As mentioned in the introduction, HAR model considers different volatility components realized over different time horizons and HAR-TCJ model looks into a more detailed decomposition of financial volatility. Their empirical results show significant improvement in forecasting volatility, indicating the characteristics of volatility are well featured by HAR-type components. Therefore, adding the HAR-type components into Neural Networks should improve the volatility forecasts compared to the Neural Networks without those input. To verify this hypothesis, we propose a hybrid model integrating HAR-type components into Neural Networks. We now explain in detail how information increases as HAR-type components are added.

In Eq.(3) of the standard HAR model, β_d measures the impact of volatility realized by actions of high-frequency traders (i.e., daily traders) on future volatility. β_w and β_m indicate the effect of volatility realized by behaviors of medium-frequency traders (i.e., weekly trades) and low-frequency traders (i.e., monthly traders) on future volatility respectively. By decomposing the volatility into a continuous part and jump part, the jump coefficient ($\beta_{j,d}$) of HAR-TCJ model in Eq.(4) provides extra information for the impact of jumps on future volatility. These parameters multiplied by the corresponding volatility component form the HAR-type components and they are more informative than the past volatility alone. The HAR-type components are then added into Neural Networks as feature input. Neural networks transform these features into higher dimensional space in the hidden layer and learn the input-output map via the training data. DFN structure is applied to improve nonlinear pattern learning and BiLSTM is used to detect long-term dependence in the financial time-series data. Figure 4 presents the architecture of the proposed hybrid model.

[INSERT FIGURE 4 ABOUT HERE]

3. Data and experiment

In the empirical analysis, we choose one stock each from ten business sectors which gives us the price data of following ten stocks: Citigroup Inc. (C), Microsoft (MSFT), Freeport-McMoran (FCX), Pfizer (PFE), General Electric (GE), The Home Depot (HD), AT&T (T), Exxon-Mobil (XOM), Duke Energy (DUK) and Wal-Mart (WMT). We also include SPY which is the ETF of S&P 500 index to see results in more aggregate level. All data are obtained from Tick Data Inc. and the sampling frequency is 5-min. Our sample covers the period from Jan 03, 2000 to Dec 31, 2020. Descriptive statistics are provided in Table 1.

The experiment is to forecast RV over daily ($h = 1$) and weekly ($h = 5$) horizons. It is comprised of the following two steps.

Step 1: Feature construction and model building

- Data preparation: The daily, weekly and monthly RV in the standard HAR model are calculated using Eq. (1) - (2). The continuous part and jump part of RV in the HAR-TCJ model are computed by Eq.(5) - (7). Table 1 provides the descriptive statistics of daily RV and daily $\hat{TC}$ for each of the considered stocks.
- Data division: To examine the forecasting performance of different models under distinct market outlook, we divide our sample into three subsamples containing approximately equal observations: the pre-crisis sample (Jan 03, 2000 to Aug 31, 2007), the post-crisis sample (Sep 04, 2007 to April 30, 2014) and the last sample (May 01, 2014 to Dec 31, 2020). In addition, two thirds of each sample forms the training set and the remainder enters into the testing set [1].
- Feature construction: To construct the HAR-type components, we estimate the parameters of the log-form HAR-type models using a rolling window of 1000 days [2]. The estimated parameters are then multiplied by the corresponding volatility components which gives us the HAR-type components.
- Feature scaling: Given the huge difference between the value range of each feature input, we standardize all features to avoid big features dominating the features with smaller values.
- Model building: Single model and hybrid model are performed to forecast RV. Single model consists of HAR, HAR-TCJ, DFN and BiLSTM where Neural Networks only take past RV and returns as input. Hybrid model contains DFN-based and BiLSTM-based hybrid models where HAR-type components are integrated as input features of Neural Networks. Details are provided in Table 2.

Step 2: Model training and testing sample forecasting

Although we apply the Neural Network models with single hidden layer in this chapter, it leaves us other tuning parameters to be determined before training (i.e., number of hidden nodes, dropout rate, batch size etc.). Tuning parameter selection is vital for the success of Neural Network models. For instance, increasing the number of hidden nodes improves the learning ability of Neural Networks characterized by the superior training sample performance. However, the results can be hardly generalized to the testing sample due to

[1] This split of training and testing example is conventional in the empirical literature and it is found that this split ratio subjects to less overfitting problem (Bucci (2020))

[2] We conduct the structural break test and find that 1000 days subject to at least one structural break for all chosen stocks.

the problem of overfitting, leading to poor testing sample performance. We conduct the walk-forward validation to select the tuning parameters where the training sample is further divided into several small training and validation sample on a rolling window basis [3]. The optimal tuning parameters for single Neural Network models and hybrid models are selected separately according to the minimum average validation loss measured by Mean Squared Errors across all chosen stocks. The search space for each tuning parameter and the selected tuning parameter values are reported in Table 3 - 4.

The single and hybrid Neural Network models are then trained and optimized by gradient-based algorithm under the chosen tuning parameters. The testing sample performance is measured by Mean Squared Errors (MSE) and Diebold-Mariano (DM) test is performed to test the significance of performance difference of different models.

[INSERT TABLE 2 ABOUT HERE]

[INSERT TABLES 3 - 4 ABOUT HERE]

4. Results

Table 5 - 7 present the out-of-sample forecasting performance of different models for the pre-crisis, post-crisis and last subsample respectively. We report the results by comparing and contrasting different models from a holistic perspective. We then look into the subsample results.

In general, at both the stock and SPY level, our proposed HAR-BiLSTM hybrid model consistently outperforms other benchmark models, achieving the minimum average forecasting loss across all forecasting horizons in all subsamples. HAR-DFN hybrid model ranks second in forecasting daily and weekly RV which also suggests the superiority of integrating Neural Networks with HAR-type models. DFN, as a short-memory Neural Network model, fails to build feedback loop between the input and hidden-layer output. BiLSTM, however, memorizes past input information for a long time and feed back (forward too as indicated by its name bidirectional) the current output through the functioning of input gate, forget gate and output gate, allowing long-memory modeling and therefore stronger pattern detection ability. In all subsamples, the DM test results show that on average the difference of forecasting performance between HAR-BiLSTM and HAR-DFN is not obvious in forecasting daily RV while becoming much more significant in forecasting weekly RV. This may suggest a more complex and longer-term dependent relationship between input variables and target output in longer forecasting horizon. Also, this reconfirms the empirical finding that long-memory

[3]The selected rolling window length is 250 days. Each small training sample contains 500 observations and the subsequent 250 observations go into the small validation sample.

Neural Network models have superior performance in capturing long-term dependence in the data (see e.g., Kim and Won (2018),Bucci (2020)). Moreover, the proposed hybrid models outperform the standard HAR and HAR-TCJ by a very large extent in forecasting both daily and weekly RV. The performance difference is especially bigger in forecasting weekly RV. This indicates that strong nonlinearity exist in HAR-type components (daily, weekly and monthly RV, continuous components and jump component) which is beyond linearity modelling and requires nonlinear modeling algorithms.

Surprisingly, in all subsamples, single DFN and single BiLSTM, which takes past RV as input features, often underperform standard HAR and HAR-TCJ. However, such performance difference becomes much less in forecasting weekly RV. This might be attributed to the strong persistence of RV in short forecasting horizon. In other words, daily RV is highly persistent and such persistence is well modeled by the standard HAR. However, strong nonlinearity appears in longer forecasting horizon and Neural Network models become more powerful in capturing the nonlinear effects. In addition, superior forecasting performance of the proposed hybrid models indicates that HAR-type components are more informative input features than past RV. Corsi (2009) shows that the HAR model successfully reproduce the main stylized facts of RV such as self-similarity and long-memory. By inputting the HAR-type components into Neural Network basis, we actually provide: (1) main stylized facts of RV (self-similarity and long-memory); (2) components of RV (continuous part and jump part); (3) magnitude of volatility induced by different market participants (daily, weekly and monthly RV); (4) the impact of actions taken by high-frequency, medium-frequency and low-frequency traders on future RV (estimated parameters of HAR-type model). With more effective information, the performance of proposed hybrid models is significantly improved compared with single Neural Network models. This indicates that feature quality is a key factor which largely affect the performance of Neural Network models. This empirical finding may be meaningful in the following aspects: (1) components generated by classical Financial Time Series models may produce high-quality input features for Neural Network models and enhance their learning ability. This is achieved by successful and in-depth modeling of the statistical properties of target variables, offering Neural Network models a deeper and more micro vision; (2) Neural Network models, however, detect and model the nonlinearity that is beyond the scope of linear Financial Time Series models. This is particularly helpful when forecasting horizon is longer or large amount of exogenous variables are added.

On average, both HAR-DFN and HAR-BiLSTM perform better in the post-crisis period compared with other subsamples. This might be attributed to the unexpected events happened in the testing period of pre-crisis and last subsample. Specifically, the testing period of pre-crisis subsample is very close to the official beginning of the Great Recession in 2007

while that of last sample covers the burst of Covid-19 crisis in 2020. As those events are not included by the training samples, the abbreviate data behavior during the crisis periods may not be sufficiently learned by the hybrid models, leading to relatively higher loss in pre-crisis and last subsample. The testing period of post-crisis sample, however, is a relatively ordinary period compared with that of pre-crisis and the last subsample in which case the training sample performance is generalized well to the testing sample. Despite these, the proposed HAR-BiLSTM hybrid model still perform the best in all subsamples across all forecasting horizons, showing the superiority of long-memory Neural Network model.

[INSERT TABLES 5 - 7 ABOUT HERE]

5. Conclusion

This chapter proposes a new nonparametric Neural Network-based hybrid models to forecast realized volatility. Unlike semiparametric HAR-type hybrid models, where nonlinear transformation term is added to the linear part, we retain a full Neural Network setting which enables more powerful pattern detection and learning. Two representative Neural Network models, Deep Feedforward Neural Network (DFN) and Bi-directional Long Short-term Memory (BiLSTM) model, are used as Neural Network basis in this chapter. Compared with DFN, BiLSTM processes information both backwards and forwards which allows dynamic bidirectional learning.

By inputting HAR-type components, the proposed hybrid models consistently outperform other benchmark models in all subsamples across all forecasting horizons. The best performer is HAR-BiLSTM hybrid model, followed by the HAR-DFN. This suggests the superiority of long-term dependence detection model and the better learning ability of BiLSTM model. In addition, HAR-type components are stronger predictors of future RV than the past RV given the underperformance of single DFN and single BiLSTM. This finding may advocate the usage of HAR-type components instead of raw RV or return data in the Neural Network models.

Moreover, on average, our proposed hybrid models achieve less forecasting loss in predicting weekly RV than predicting daily RV. This indicates that daily RV is highly persistent and such persistence is well modeled by the standard HAR. However, strong nonlinearity appears in longer forecasting horizon and Neural Network models become more powerful in capturing the nonlinear effects.

Appendix A. Tables

Table 1: Summary statistics

	MSFT	C	DUK	FCX	HD	PFE	T	WMT	XOM	GE	SPY
	RV_t										
Mean	2.56	6.08	2.28	7.81	2.92	2.23	2.46	1.95	2.07	3.39	1.01
Std.	4.02	24.87	5.79	12.09	5.20	3.40	4.57	3.33	4.54	7.21	2.36
Skew.	6.06	19.31	15.66	6.18	7.69	7.28	9.42	7.06	12.43	8.98	9.79
Kurt.	59.27	567.11	375.27	59.84	97.08	88.96	194.58	87.41	250.98	133.38	146.79
	$\hat{TC}_t$										
Mean	2.35	5.29	1.91	6.58	2.56	1.90	2.17	1.70	1.89	2.96	0.94
Std.	3.85	21.26	4.50	11.04	4.62	2.81	4.27	3.03	4.30	6.38	2.17
Skew.	6.28	23.62	11.91	7.03	8.01	7.06	10.48	7.79	13.65	10.11	10.31
Kurt.	64.13	896.82	222.39	76.74	108.07	88.51	243.52	110.79	302.83	179.15	173.07

Note: This table reports descriptive statistics of realized volatility and the continuous part of variation for SPY and other selected stocks. Descriptive statistics includes mean (Mean.), standard deviation (Std.), Skewness (Skew.) and Kurtosis (Kurt.).

Table 2: Input variables of different models

Model	Return	Lagged RV	HAR Components	HAR-TCJ Components
Single SDN	Y	Y	N	N
Single BiLSTM	Y	Y	N	N
HAR-DFN	Y	N	Y	Y
HAR-BiLSTM	Y	N	Y	Y

Note: This table presents the input variables of single models and hybrid models applied in this study. Y indicates the corresponding variable is added to the model while N suggests the opposite.

Table 3: Selected tuning parameter values for $h = 1$

Model	Parameters	Value	Search Space
Single DFN	Number of hidden nodes	4	[1, 2*Number of input features]; Increment = 1
	Hidden activation function	RELU	RELU, Tanh
	Batch size	20	[20, 32]; Increment = 1
	Dropout rate	0.2	[0.2, 0.5]; Increment = 0.1
	Number of epochs	Early callback	Maximum number of epochs is 150 with early callback to reduce processing time
Single BiLSTM	Number of hidden nodes	4	[1, 2*Number of input features]; Increment = 1
	Dropout rate	0.2	[0.2, 0.5]; Increment = 0.1
	Number of epochs	150	Maximum number of epochs is 150
	Merge mode	Multiplicative	Multiplicative, Average, Sum, Concatenate
HAR-DFN	Number of hidden nodes	8	[1, 2*Number of input features]; Increment = 1
	Hidden activation function	RELU	RELU, Tanh
	Batch size	26	[20, 32]; Increment = 1
	Dropout rate	0.2	[0.2, 0.5];Increment = 0.1
	Number of epochs	Early callback	Maximum number of epochs is 150 with early callback to reduce processing time
HAR-BiLSTM	Number of hidden nodes	8	[1, 2*Number of input features]; Increment = 1
	Dropout rate	0.3	[0.2, 0.5]; Increment = 0.1
	Number of epochs	150	Maximum number of epochs is 150
	Merge mode	Multiplicative	Multiplicative, Average, Sum, Concatenate

Note: This table reports the tuning parameter values used in different models for $h = 1$. These parameters are selected according to the minimum average validation loss. The search space for each parameter is determined by empirical research convention.

Table 4: Selected tuning parameter values for $h = 5$

Model	Parameters	Value	Search Space
Single DFN	Number of hidden nodes	8	[1, 2*Number of input features]; Increment = 1
	Hidden activation function	RELU	RELU, Tanh
	Batch size	32	[20, 32]; Increment = 1
	Dropout rate	0.3	[0.2, 0.5]; Increment = 0.1
	Number of epochs	Early callback	Maximum number of epochs is 150 with early callback to reduce processing time
Single BiLSTM	Number of hidden nodes	8	[1, 2*Number of input features]; Increment = 1
	Dropout rate	0.3	[0.2, 0.5]; Increment = 0.1
	Number of epochs	150	Maximum number of epochs is 150
	Merge mode	Multiplicative	Multiplicative, Average, Sum, Concatenate
HAR-DFN	Number of hidden nodes	16	[1, 2*Number of input features]; Increment = 1
	Hidden activation function	RELU	RELU, Tanh
	Batch size	32	[20, 32]; Increment = 1
	Dropout rate	0.3	[0.2, 0.5]; Increment = 0.1
	Number of epochs	Early callback	Maximum number of epochs is 150 with early callback to reduce processing time
HAR-BiLSTM	Number of hidden nodes	8	[1, 2*Number of input features]; Increment = 1
	Dropout rate	0.3	[0.2, 0.5]; Increment = 0.1
	Number of epochs	150	Maximum number of epochs is 150
	Merge mode	Multiplicative	Multiplicative, Average, Sum, Concatenate

Note: This table reports the tuning parameter values used in different models for $h = 5$. These parameters are selected according to the minimum average validation loss. The search space for each parameter is determined by empirical research convention.

Table 5: Out-of-sample forecast losses: pre-crisis sample

	MSFT	C	DUK	FCX	HD	PFE	T	WMT	XOM	GE	Average	SPY
Horizon = 1												
HAR	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
HAR-TCJ	1.007	0.993	0.978	0.990	1.007	0.983	0.996	0.991	0.996	0.980	0.992	1.004
DFN	1.706 ***	1.361 ***	1.134 ***	1.158 ***	1.255 ***	1.609 ***	1.705 ***	1.198 ***	1.270 ***	1.514 ***	1.391	1.280 ***
BiLSTM	1.542 ***	1.589 ***	1.331 ***	1.248 ***	1.212 ***	1.081 **	1.797 ***	1.124 ***	1.664 ***	1.498 ***	1.309	1.483 ***
HAR-DFN	0.967 *	0.968 *	0.970 *	0.993	1.005	0.979	1.071	0.998	1.001	0.903 **	0.986	0.881 ***
HAR-BiLSTM	0.963 *	0.875 ***	0.609 ***	0.954 **	0.955 **	0.858 ***	1.017	0.973	1.003	0.819 ***	**0.903**	**0.879 *** **
Horizon = 5												
HAR	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
HAR-TCJ	0.996	0.996	0.995	0.986	0.983	0.985	0.988	0.996	1.009	1.008	0.994	1.010
DFN	1.041 ***	1.185 ***	1.187 ***	1.013 ***	1.006	1.038 ***	1.136 ***	0.995	1.047 ***	1.020 ***	1.067	1.038 ***
BiLSTM	1.027 ***	1.045 ***	1.084 ***	1.008	1.000	1.001	1.119 ***	0.991	1.017 ***	1.001	1.029	1.019 ***
HAR-DFN	0.936 **	0.945 **	0.925 **	1.000	0.977	0.993	0.951 *	0.967 *	0.996	0.922 **	0.961	0.679 ***
HAR-BiLSTM	0.886 ***	0.887 ***	0.910 **	0.929 **	0.929 **	0.990	0.813 ***	0.959 *	0.849 ***	0.888 ***	**0.904**	**0.690 *** **

Note: The table reports the ratio between the MSE of different models and the MSE of the standard HAR model for the pre-crisis sample. Figures lower than one indicates the corresponding model achieves lower MSE than the standard HAR while figures higher than one suggests the opposite. Column Average reports the average MSE of the selected 10 individual stocks. The lowest loss ratio in each column of the Average and the SPY is highlighted in bold.*, ** and *** indicate that the relative forecasting differences between the corresponding model and the standard HAR are significant at 10%, 5% and 1% level using the Diebold–Mariano test (Newey–West heteroscedasticity consistent covariance matrix estimator). Pre-crisis sample covers the period from Jan 03, 2000 to Aug 31, 2007 with 1927 observations in total. Two thirds of the pre-crisis sample are the training sample and the remainder forms the testing sample.

Table 6: Out-of-sample forecast losses: post-crisis sample

	MSFT	C	DUK	FCX	HD	PFE	T	WMT	XOM	GE	Average	SPY
Horizon = 1												
HAR	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
HAR-TCJ	1.187 ***	1.145 ***	0.952 **	1.032 ***	1.481 ***	1.028 ***	1.064 ***	0.948 **	1.042 ***	1.058 ***	1.094	1.026 ***
DFN	1.023 **	1.020 **	1.199 ***	1.759 ***	1.089 ***	1.216 ***	1.063 ***	1.245 ***	1.108 ***	1.001	1.172	1.011
BiLSTM	1.165 ***	1.138 ***	1.233 ***	1.131 ***	1.029 ***	1.065 ***	1.118 ***	1.224 ***	1.131 ***	1.033 ***	1.127	1.025 ***
HAR-DFN	0.987	0.954 **	0.949 **	0.939 **	0.876 ***	0.897 ***	0.811 ***	0.972	0.998	0.741 ***	0.912	0.956 **
HAR-BiLSTM	0.979	0.896 ***	0.917 **	0.967 *	0.790 ***	0.817 ***	0.792 ***	0.825 ***	0.924 **	0.720 ***	**0.863**	**0.899 *****
Horizon = 5												
HAR	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
HAR-TCJ	0.964 *	0.973	0.979	1.002	1.052 ***	0.984	1.011	0.988	0.813 ***	1.020 **	0.979	1.003
DFN	0.987	1.044 ***	1.055 ***	1.094 ***	1.052 ***	1.020 **	1.092 ***	1.041 ***	1.066 ***	1.085 ***	1.054	1.102 ***
BiLSTM	1.089 ***	0.996	1.050 ***	0.994	1.049 ***	0.982	1.023 **	1.019	1.038 ***	1.020 **	1.026	1.099 ***
HAR-DFN	0.993	0.825 ***	0.707 ***	0.886 ***	0.801 ***	0.743 ***	0.886 ***	0.934 **	0.995	0.706 ***	0.848	0.948 **
HAR-BiLSTM	0.960 *	0.768 ***	0.685 ***	0.713 ***	0.794 ***	0.636 ***	0.743 ***	0.652 ***	0.890 ***	0.704 ***	**0.755**	**0.811 *****

Note: The table reports the ratio between the MSE of different models and the MSE of the standard HAR model for the post-crisis sample. Figures lower than one indicates the corresponding model achieves lower MSE than the standard HAR while figures higher than one suggests the opposite. Column Average reports the average MSE of the selected 10 individual stocks. The lowest loss ratio in each column of the Average and the SPY is highlighted in bold.*, ** and *** indicate that the relative forecasting differences between the corresponding model and the standard HAR are significant at 10%, 5% and 1% level using the Diebold–Mariano test (Newey–West heteroscedasticity consistent covariance matrix estimator). Post-crisis sample covers the period from Sep 04, 2007 to April 30, 2014 with 1676 observations in total. Two thirds of the post-crisis sample are the training sample and the remainder forms the testing sample.

Table 7: Out-of-sample forecast losses: last sample

	MSFT	C	DUK	FCX	HD	PFE	T	WMT	XOM	GE	Average	SPY
Horizon = 1												
HAR	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
HAR-TCJ	1.027 ***	1.025 ***	0.988	0.994	0.982	1.008	0.956 **	0.996	0.974	1.012	0.996	0.964
DFN	0.975	1.105 ***	1.021 ***	1.079 ***	1.302 ***	1.143 ***	0.950 **	0.992	1.072 ***	1.080 ***	1.072	1.126 ***
BiLSTM	1.037 ***	1.006	1.166 ***	1.003	1.225 ***	1.130 ***	0.944 **	0.999	1.011 ***	0.985	1.051	1.216 ***
HAR-DFN	0.960 *	0.932 **	0.987	0.974	0.989	0.985	0.919 **	0.942 **	0.965	0.922 **	0.958	0.929 **
HAR-BiLSTM	0.948 **	0.844 ***	0.953 **	0.927 **	0.937 **	0.975	0.910 **	0.936 **	0.957 **	0.888 ***	**0.928**	**0.910 ** **
Horizon = 5												
HAR	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
HAR-TCJ	0.989	0.999	1.000	1.001	0.989	1.002	1.000	1.008	1.000	0.987	0.998	1.005
DFN	1.019 ***	1.980 ***	1.021 ***	1.072 ***	1.125 ***	1.045 ***	1.010	1.014	1.082 ***	1.030 ***	1.140	0.999
BiLSTM	0.988	1.227 ***	1.040 ***	1.070 ***	1.008	1.009	1.002	0.956 **	1.047 ***	1.027 ***	1.037	1.023 ***
HAR-DFN	0.889 ***	0.856 ***	0.995	0.883 ***	0.725 ***	0.815 ***	0.894 ***	0.852 ***	0.982	0.877 ***	0.877	0.874 ***
HAR-BiLSTM	0.851 ***	0.841 ***	0.854 ***	0.864 ***	0.705 ***	0.803 ***	0.799 ***	0.743 ***	0.706 ***	0.734 ***	**0.790**	**0.760 *** **

Note: The table reports the ratio between the MSE of different models and the MSE of the standard HAR model for the last sample. Figures lower than one indicates the corresponding model achieves lower MSE than the standard HAR while figures higher than one suggests the opposite. Column Average reports the average MSE of the selected 10 individual stocks. The lowest loss ratio in each column of the Average and the SPY is highlighted in bold.*, ** and *** indicate that the relative forecasting differences between the corresponding model and the standard HAR are significant at 10%, 5% and 1% level using the Diebold–Mariano test (Newey–West heteroscedasticity consistent covariance matrix estimator). The last sample covers the period from May 01, 2014 to Dec 31, 2020 with 1681 observations in total. Two thirds of the last sample are the training sample and the remainder forms the testing sample.

Appendix B. Figures

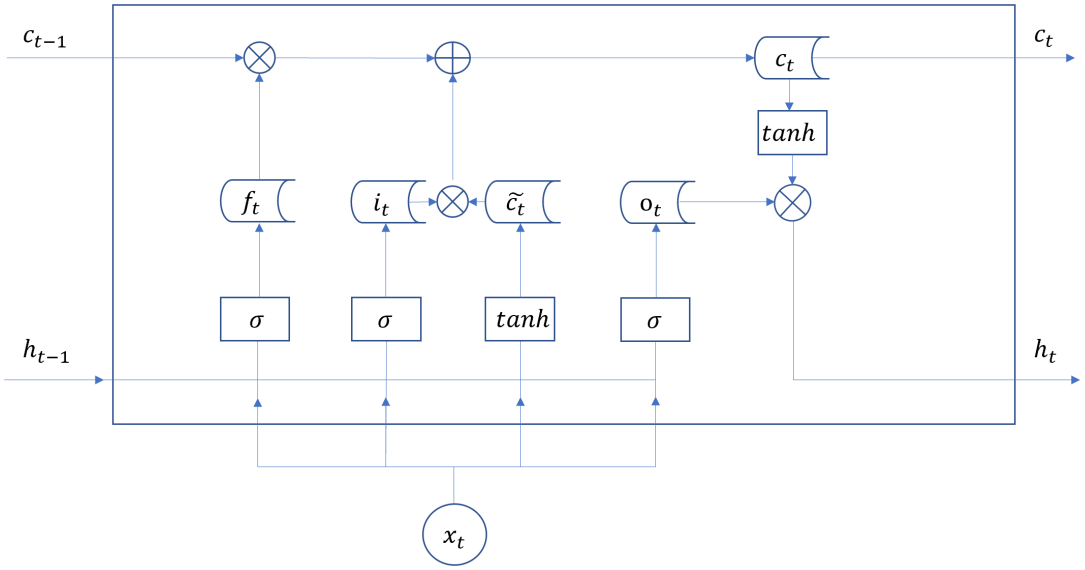
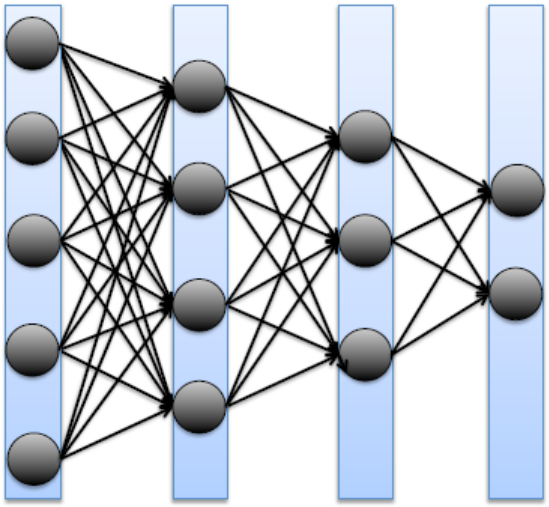
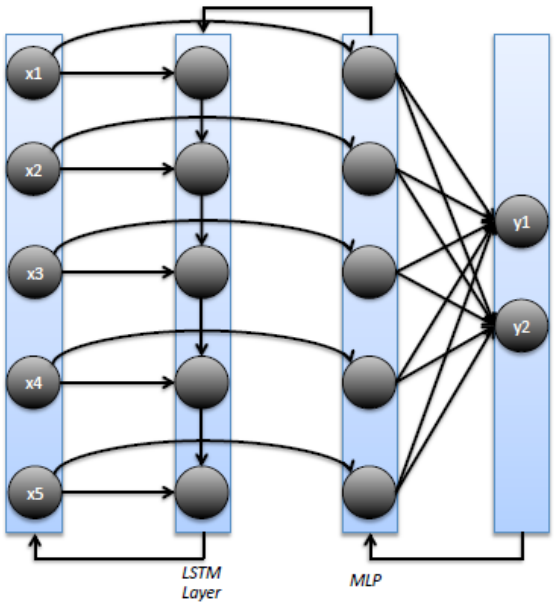


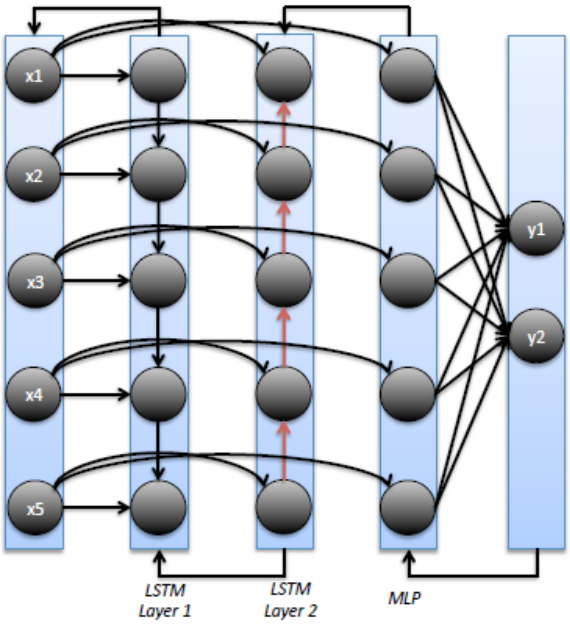
Fig. 1. LSTM memory cell



(a) Feedforward Neural Network

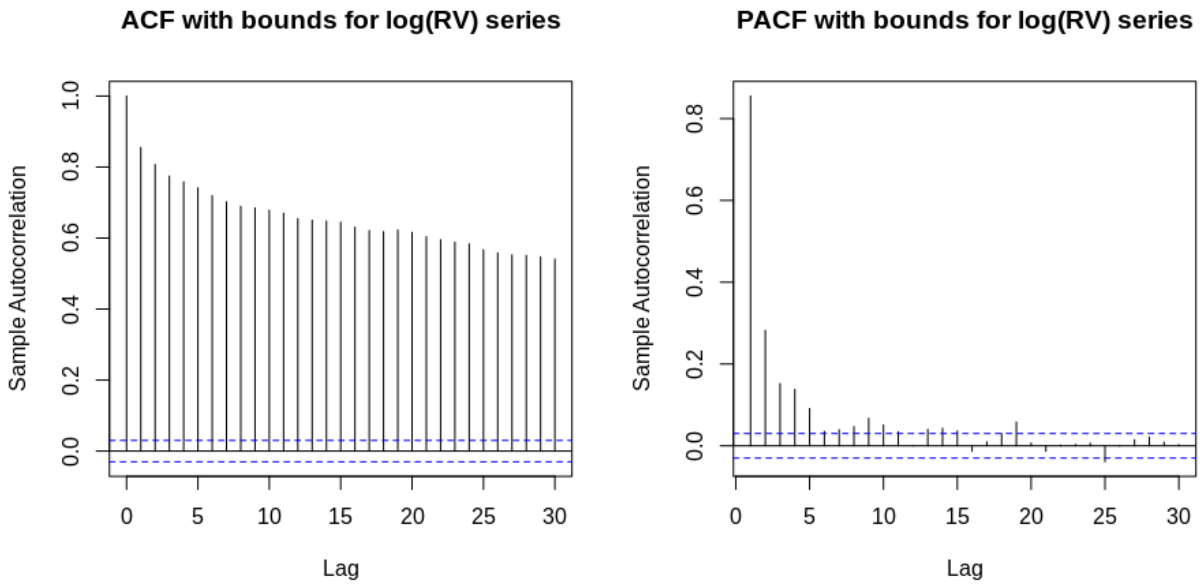


(b) Long Short-term Memory Network



(c) Bidirectional Long Short-term Memory Network

Fig. 2. Structure of Neural Network models



(a) ACF plot for $log(RV)$

(b) PACF plot for $log(RV)$

Fig. 3. ACF and PACF plot of SPY's RV

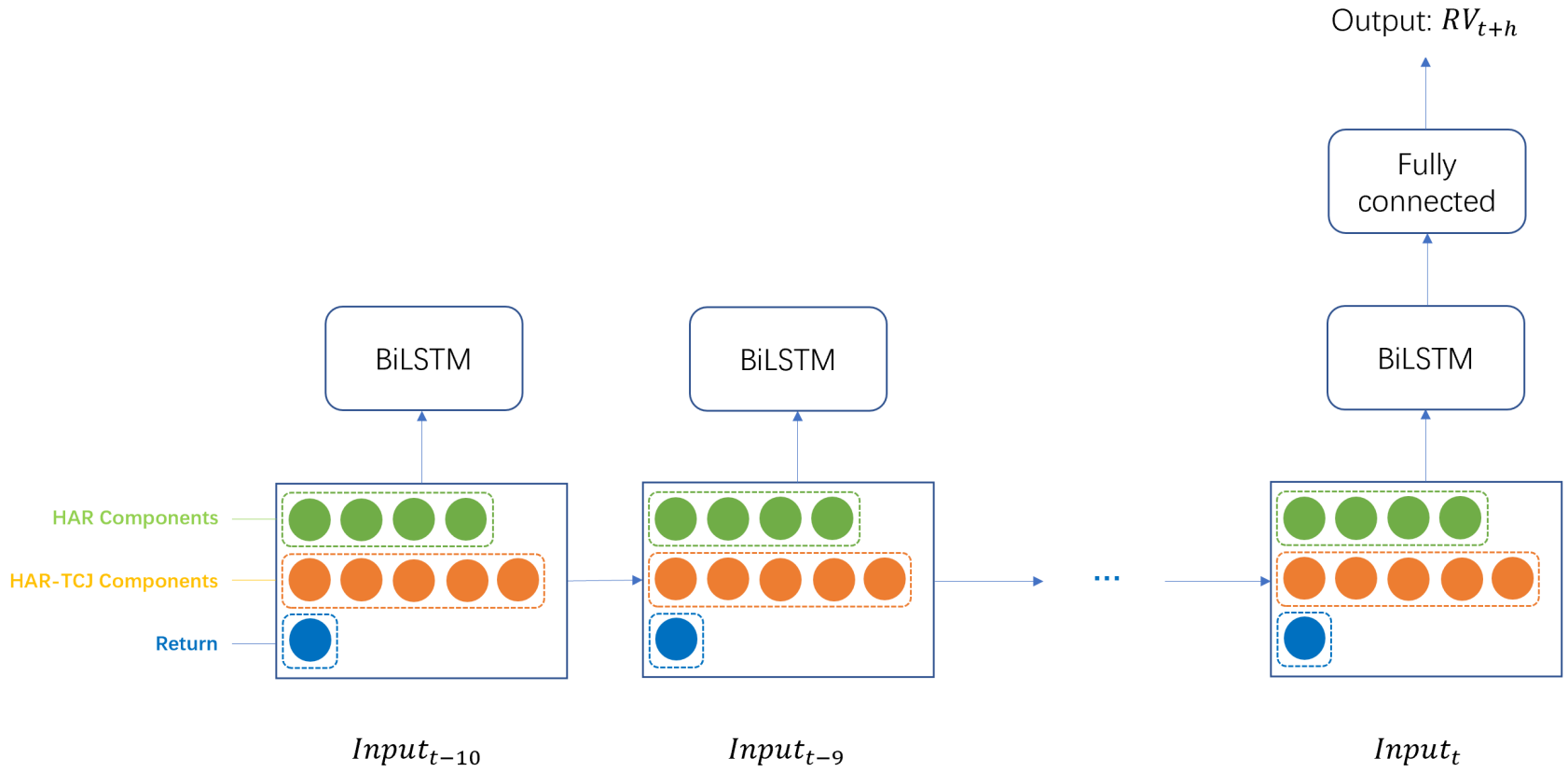


Fig. 4. Structure of the proposed HAR-BiLSTM hybrid model

Note: This figure shows the structure of the proposed HAR-BiLSTM model. To predict the target output RV_{t+h}, HAR-type components are integrated into BiLSTM network as input variables from $t-10$ to t. A fully connected dense layer is added to the BiLSTM network at time t to project results into correct dimension.

References

Andersen, T. G., Bollerslev, T., 1998. Answering the skeptics: Yes, standard volatility models do provide accurate forecasts. International economic review pp. 885–905.

Andersen, T. G., Bollerslev, T., Diebold, F. X., 2007. Roughing it up: Including jump components in the measurement, modeling, and forecasting of return volatility. The review of economics and statistics 89, 701–720.

Andersen, T. G., Bollerslev, T., Diebold, F. X., Ebens, H., 2001. The distribution of realized stock return volatility. Journal of financial economics 61, 43–76.

Andersen, T. G., Bollerslev, T., Diebold, F. X., Labys, P., 2003. Modeling and forecasting realized volatility. Econometrica 71, 579–625.

Arnerić, J., Poklepović, T., Aljinović, Z., 2014. Garch based artificial neural networks in forecasting conditional variance of stock returns. Croatian Operational Research Review pp. 329–343.

Barndorff-Nielsen, O. E., Shephard, N., 2002. Econometric analysis of realized volatility and its use in estimating stochastic volatility models. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 64, 253–280.

Bollerslev, T., 1986. Generalized autoregressive conditional heteroskedasticity. Journal of econometrics 31, 307–327.

Bucci, A., 2020. Realized volatility forecasting with neural networks. Journal of Financial Econometrics 18, 502–531.

Chen, J.-F., Chen, W.-L., Huang, C.-P., Huang, S.-H., Chen, A.-P., 2016. Financial time-series data analysis using deep convolutional neural networks. In: *2016 7th International conference on cloud computing and big data (CCBD)*, IEEE, pp. 87–92.

Corsi, F., 2009. A simple approximate long-memory model of realized volatility. Journal of Financial Econometrics 7, 174–196.

Corsi, F., Pirino, D., Reno, R., 2010. Threshold bipower variation and the impact of jumps on volatility forecasting. Journal of Econometrics 159, 276–288.

Dash, P. K., Nayak, M., Senapati, M. R., Lee, I. W., 2007. Mining for similarities in time series data using wavelet-based feature vectors and neural networks. Engineering Applications of Artificial Intelligence 20, 185–201.

Doering, J., Fairbank, M., Markose, S., 2017. Convolutional neural networks applied to high-frequency market microstructure forecasting. In: *2017 9th computer science and electronic engineering (ceec)*, IEEE, pp. 31–36.

Donaldson, R. G., Kamstra, M., 1997. An artificial neural network-garch model for international stock return volatility. Journal of Empirical Finance 4, 17–46.

Engle, R. F., 1982. Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. Econometrica: Journal of the econometric society pp. 987–1007.

Faraway, J., Chatfield, C., 1998. Time series forecasting with neural networks: a comparative study using the air line data. Journal of the Royal Statistical Society: Series C (Applied Statistics) 47, 231–250.

Fernandes, M., Medeiros, M. C., Scharth, M., 2014. Modeling and predicting the cboe market volatility index. Journal of Banking & Finance 40, 1–10.

Hajizadeh, E., Seifi, A., Zarandi, M. F., Turksen, I., 2012. A hybrid modeling approach for forecasting the volatility of s&p 500 index return. Expert Systems with Applications 39, 431–436.

Hamid, S. A., Iqbal, Z., 2004. Using neural networks for forecasting volatility of s&p 500 index futures prices. Journal of Business Research 57, 1116–1125.

Hewamalage, H., Bergmeir, C., Bandara, K., 2021. Recurrent neural networks for time series forecasting: Current status and future directions. International Journal of Forecasting 37, 388–427.

Hillebrand, E., Medeiros, M. C., 2010. The benefits of bagging for forecast models of realized volatility. Econometric Reviews 29, 571–593.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural computation 9, 1735–1780.

Horvath, B., Muguruza, A., Tomas, M., 2021. Deep learning volatility: a deep neural network perspective on pricing and calibration in (rough) volatility models. Quantitative Finance 21, 11–27.

Khan, M. A. I., 2011. Financial volatility forecasting by nonlinear support vector machine heterogeneous autoregressive model: evidence from nikkei 225 stock index. International Journal of Economics and Finance 3, 138.

Kim, H. Y., Won, C. H., 2018. Forecasting the volatility of stock price index: A hybrid model integrating lstm with multiple garch-type models. Expert Systems with Applications 103, 25–37.

Kristjanpoller, W., Fadic, A., Minutolo, M. C., 2014. Volatility forecast using hybrid neural network models. Expert Systems with Applications 41, 2437–2442.

Lai, Y., Chen, C. W., Gerlach, R., 2009. Optimal dynamic hedging via copula-threshold-garch models. Mathematics and Computers in Simulation 79, 2609–2624.

Maciel, L., Gomide, F., Ballini, R., 2016. Evolving fuzzy-garch approach for financial volatility modeling and forecasting. Computational Economics 48, 379–398.

Maheu, J. M., McCurdy, T. H., 2002. Nonlinear features of realized fx volatility. Review of

Economics and Statistics 84, 668–681.

McAleer, M., Medeiros, M. C., 2008. A multiple regime smooth transition heterogeneous autoregressive model for long memory and asymmetries. Journal of Econometrics 147, 104–119.

Nanopoulos, A., Alcock, R., Manolopoulos, Y., 2001. Feature-based classification of time-series data. International Journal of Computer Research 10, 49–61.

Nikolaev, N., Tino, P., Smirnov, E., 2013. Time-dependent series variance learning with recurrent mixture density networks. Neurocomputing 122, 501–512.

Pavlidis, E. G., Paya, I., Peel, D. A., 2012. Forecast evaluation of nonlinear models: The case of long-span real exchange rates. Journal of Forecasting 31, 580–595.

Petneházi, G., 2019. Recurrent neural networks for time series forecasting. arXiv preprint arXiv:1901.00069 .

Psaradellis, I., Sermpinis, G., 2016. Modelling and trading the us implied volatility indices. evidence from the vix, vxn and vxd indices. International Journal of Forecasting 32, 1268–1283.

Roh, T. H., 2007. Forecasting the volatility of stock price index. Expert Systems with Applications 33, 916–922.

Schittenkopf, C., Dorffner, G., Dockner, E. J., 2000. Forecasting time-dependent conditional densities: a semi non-parametric neural network approach. Journal of Forecasting 19, 355–374.

Schuster, M., Paliwal, K. K., 1997. Bidirectional recurrent neural networks. IEEE transactions on Signal Processing 45, 2673–2681.

Sharkey, A. J., 2002. Types of multinet system. In: *International Workshop on Multiple Classifier Systems*, Springer, pp. 108–117.

Sheela, K. G., Deepa, S. N., 2013. Review on methods to fix number of hidden neurons in neural networks. Mathematical problems in engineering 2013.

Sjöberg, J., Zhang, Q., Ljung, L., Benveniste, A., Delyon, B., Glorennec, P.-Y., Hjalmarsson, H., Juditsky, A., 1995. Nonlinear black-box modeling in system identification: a unified overview. Automatica 31, 1691–1724.

Stinchcombe, M., White, H., 1992. Using feedforward networks to distinguish multivariate populations. In: *[Proceedings 1992] IJCNN International Joint Conference on Neural Networks*, IEEE, vol. 1, pp. 788–793.

Taskaya-Temizel, T., Casey, M. C., 2005. A comparative study of autoregressive neural network hybrids. Neural Networks 18, 781–789.

Tino, P., Schittenkopf, C., Dorffner, G., 2001. Financial volatility trading using recurrent neural networks. IEEE transactions on neural networks 12, 865–874.

Tseng, C.-H., Cheng, S.-T., Wang, Y.-H., Peng, J.-T., 2008. Artificial neural network model of the hybrid egarch volatility of the taiwan stock index option prices. Physica A: Statistical Mechanics and its Applications 387, 3192–3200.

Xiong, R., Nichols, E. P., Shen, Y., 2015. Deep learning stock volatility with google domestic trends. arXiv preprint arXiv:1512.04916 .

Zhang, G. P., Berardi, V. L., 2001. Time series forecasting with neural network ensembles: an application for exchange rate prediction. Journal of the operational research society 52, 652–664.

Zhang, G. P., Qi, M., 2005. Neural network forecasting for seasonal and trend time series. European journal of operational research 160, 501–514.

Zheng, Y., Liu, Q., Chen, E., Ge, Y., Zhao, J. L., 2016. Exploiting multi-channels deep convolutional neural networks for multivariate time series classification. Frontiers of Computer Science 10, 96–112.

Zhou, Y.-L., Han, R.-J., Xu, Q., Jiang, Q.-J., Zhang, W.-K., 2019. Long short-term memory networks for csi300 volatility prediction with baidu search volume. Concurrency and Computation: Practice and Experience 31, e4721.