Electronic Research Archive

http://www.aimspress.com/journal/era

ERA, 31(7): 3814–3831.
DOI: 10.3934/era.2023194
Received: 01 March 2023
Revised: 20 April 2023
Accepted: 22 April 2023
Published: 08 May 2023

Research article

A mixture deep neural network GARCH model for volatility forecasting

Wenhui Feng[1], **Yuan Li**[2,*] **and Xingfa Zhang**[1]

[1] School of Economics and Statistics, Guangzhou University, Guangzhou 510006, China
[2] Institute of Applied Mathematics, Shenzhen Polytechnic, Shenzhen 518000, China

* **Correspondence:** Email: mathly@gzhu.edu.cn.

Abstract: Recently, deep neural networks have been widely used to solve financial risk modeling and forecasting challenges. Following this hotspot, this paper presents a mixture model for conditional volatility probability forecasting based on the deep autoregressive network and the Gaussian mixture model under the GARCH framework. An efficient algorithm for the model is developed. Both simulation and empirical results show that our model predicts conditional volatilities with smaller errors than the classical GARCH and ANN-GARCH models.

Keywords: volatility forecasting; deep autoregressive network; GARCH model

1. Introduction

Volatility forecasting is essential in asset pricing, portfolio allocation and risk management research. Early volatility forecasting was based on economic models. The most famous economic models are the ARCH model [1] and the GARCH model [2], which can capture volatility clustering and heavy-tail features. However, they fail to capture asymmetry, such as leverage effects. The leverage effect is due to the fact that negative returns have a more significant impact on future volatility than positive returns. To overcome this drawback, the exponential GARCH (EGARCH) model [3] and GJR model [4] were proposed. In the following years, new volatility models based on the GARCH model emerged, such as the stochastic volatility model [5] proposed by Hull and White and the realized volatility model [6] offered by Blair et al. They formed a class of GARCH-type volatility models for financial markets.

The traditional GARCH model has strict constraints and requires the financial time series to satisfy the stationarity condition. It usually assumes conditional variances have a linear relationship with previous errors and previous variances. However, many financial time series show certain nonstationary and nonlinear characteristics in practice. Consequently, some extended model from GARCH is necessary to study the volatility of these time series.

With the development of computer and big data technologies, machine learning brings new ideas to

volatility forecasting [7–21]. Especially artificial neural networks(ANN) have shown an outstanding performance. It derives its computational ideas from biological neurons and is now widely used in various fields.

In financial risk analysis, researchers have utilized neural networks to study the volatility of financial markets. Hamid and Iqbal [22] apply ANN to predict the S&P500 index implied volatility, finding that ANN's forecasting performance surpasses that of the American option pricing model. Livieris proposes an artificial neural network prediction model for forecasting gold prices and trends [23]. Additionally, Dunis and Huang [24] explore neural network regression (NNR), recurrent neural networks, and their collaborative NNR-RNN models for predicting and trading the volatility of daily exchange rates of GBP/USD and USD/JPY, with results indicating that RNNs have the best volatility forecasting performance. Beyond the direct application of neural networks, researchers have investigated a series of mixture models [25–31] that combine ANNs and GARCH models. Liu et al. [32] introduce a volatility forecasting model based on the recurrent neural network (RNN) and the GARCH model. Experiments reveal that such mixture models enhance the predictive capabilities of traditional GARCH models, capturing normality, skewness and kurtosis of financial volatility more accurately.

This study employs a mixture model (DeepAR-GMM-GARCH) that combines the deep autoregressive network, the Gaussian mixture model and the GARCH model for probabilistic volatility forecasting. First, we discuss the design of the mixture model; Second, the article presents the model's inference and the training algorithm of the model; Third, we conduct a simulation experiment using artificial data and compare the outcomes with traditional GARCH models, finding that our model yields smaller RMSE and MAE; Last, we investigate the correlation between the square of extreme values and the square of returns for the CSI300 index. The empirical data is partitioned into training and test sets. After training and testing, we analyze the prediction results and observe that our proposed model outperforms other models in both in-sample and out-of-sample analyses.

The key offerings presented in this article can be summarized as follows: Initially, this article introduces a novel conditional volatility probability prediction model, which addresses the leptokurtic and heavy-tail traits of conventional financial volatility. This model is built upon a deep autoregressive network combined with a Gaussian mixture distribution. Subsequently, we incorporate extreme values into the mixture model via the neural network. It is discovered that the inclusion of extreme values enhances the accuracy of volatility predictions.

The structure of this paper is as follows: Section 2 outlines the GARCH model and the deep autoregressive network. Section 3 delves into the mixture model, elaborating on inference, prediction and the relevant algorithm. Section 4 encompasses the simulation studies that we propose. Lastly, Section 5 focuses on the empirical analysis of our proposed model.

2. The GARCH model and the deep autoregressive network

2.1. The GARCH model

Scholars usually believe that stock price or stock index returns are nonlinear, asymmetric, heavy-tailed, returns are generally uncorrelated. Aggregation characterises volatility, which was first found by Engle (1982) and Bollerslev (1986) in ARCH and GARCH models. The GARCH model is defined

as follows:

$$\begin{aligned} r_t &= \varepsilon_t \sqrt{h_t}, \\ h_t &= \alpha_0 + \sum_{i=1}^{q} \alpha_i r_{t-i}^2 + \sum_{j=1}^{p} \beta_j h_{t-j}, \end{aligned} \tag{2.1}$$

where h_t is the conditional heteroscedastic variance of return series r_t.

Although there are many criteria of GARCH(p,q) models to find p and q, it is sufficient to apply the GARCH(1,1) model to characterize the conditional volatilities.

2.2. The DeepAR model

The DeepAR model [33], illustrated in Figure 1, is a time series forecasting model that employs a deep autoregressive recurrent network architecture. Distinct from other time series forecasting models, DeepAR generates probabilistic predictions.

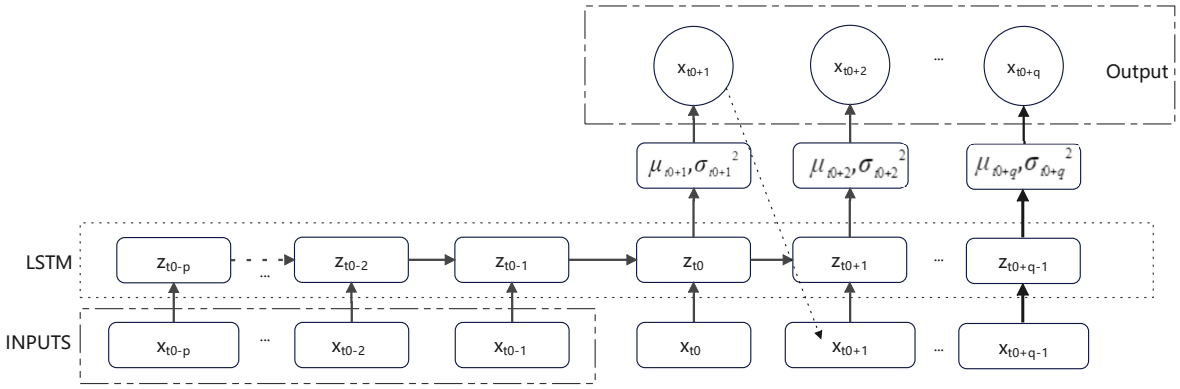


Figure 1. The network structure of the DeepAR model with p inputs and q outputs.

Consider a time series $[x_1, \ldots, x_{t_0}, x_{t_0+1}, \ldots, x_T] := x_{1:T}$. Given its past time series $[x_1, \ldots, x_{t_0-2}, x_{t_0-1}] := x_{1:t_0-1}$, our objective is to predict the future time series $[x_{t_0}, \ldots, x_{t_0+(T-1)}, x_{t_0+T}] := x_{t_0:t_0+T}$. The DeepAR model constructs the conditional distribution $P_\Theta(x_{t_0:T}|x_{1:t_0-1})$ using a latent factor z, which is implemented by a deep recurrent network architecture. This conditional distribution, $P_\Theta(x_{t_0:T}|x_{1:t_0-1})$, comprises a product of likelihood factors (z)

$$\begin{aligned} P_\Theta\left(\mathbf{x}_{t_0:T} \mid \mathbf{x}_{1:t_0-1}\right) &= \prod_{t=t_0}^{T} P_\Theta\left(x_t \mid \mathbf{x}_{1:t-1}\right) \\ &= \prod_{t=t_0}^{T} p\left(x_t \mid \theta\left(\mathbf{z}_t, \Theta\right)\right). \end{aligned} \tag{2.2}$$

The likelihood $p(x_t|\theta(z_t))$ is a fixed distribution with parameters determined by a function $\theta(z_t, \Theta)$ of the network output z_t. As suggested by the model's authors, Gaussian likelihood is appropriate for real-valued data.

3. The mixture model

Forecasting the probability of volatility in finance and economics is an important problem. There are mainly two methods to tackle this. First, statistical models such as ARCH and the GARCH models are usually adopted. These models are specifically designed to capture the dynamic nature of volatility over time and help to predict future levels of volatility based on past patterns.

Another strategy involves using machine learning models, such as neural networks, which can analyze vast amounts of data and uncover patterns that may not be readily apparent to human analysts. A case in point is the DeepAR model is a series-to-series probabilistic forecasting model. The advantages of the DeepAR model are: it makes probabilistic forecasting and allows to introduce additional covariates. Due to these advantages, it can be used to predict financial volatility(h_t) based on the series r_t^2. However, the DeepAR model usually assumes that $p(x_t|\theta(z_t))$ (given in (2.2)) follows a Gaussian distribution, which may be unreasonable due to the non-negative, leptokurtic and heavy-tail characteristics of traditional financial volatility. To avoid this problem, people use the gaussian mixture distribution to describe the density of $p(ln(x_t)|\theta(z_t))$, see references [34]. Motivated by the above results, this paper propose an improved mixture model: DeepAR-GMM-GARCH.

The conditional distribution of $ln(h_t)$ can be expressed as:

$$P(ln(h_t)|r^2_{1:t-1}, \mathbf{x}_{1:t-1}), \tag{3.1}$$

where h_t represents the future volatility at time t, $[r_1, ..., r_{t-2}, r_{t-1}] := r_{1:t-1}$ denotes the past return series during the $[1 : t-1]$ period, and $\mathbf{x}_{1:t-1}$ refers to the covariate, which is observable at all times. The past time horizon is represented by $[1 : t-1]$.

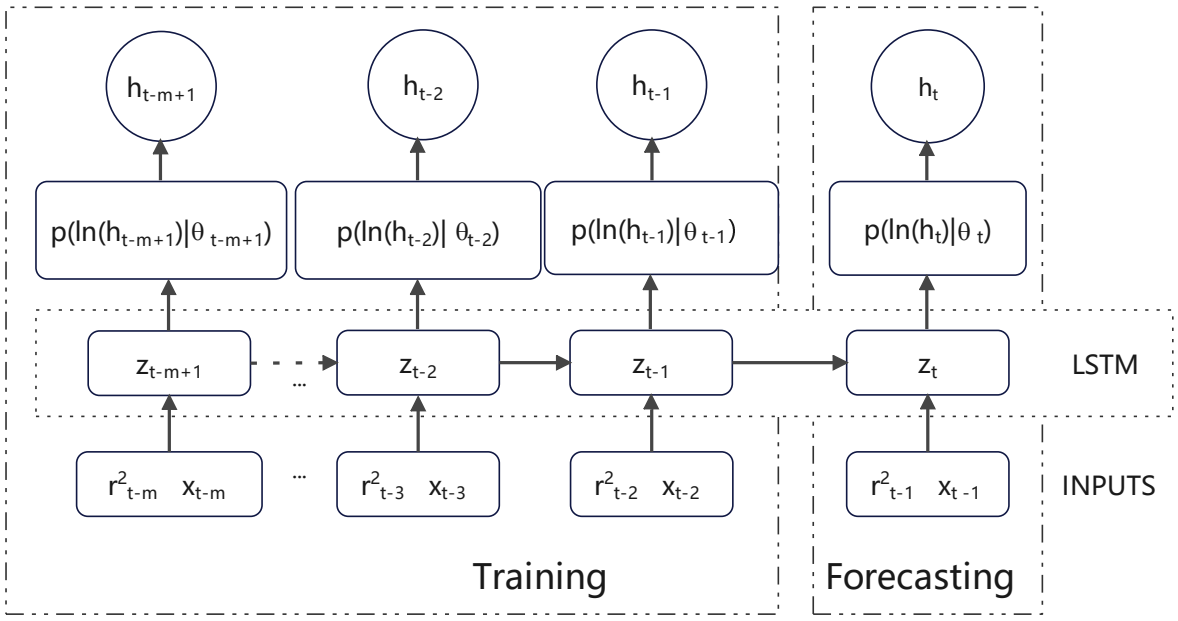


Figure 2. The network structure of the DeepAR-GMM-GARCH model with m inputs and one output.

The proposed hybrid model assumes that the conditional density for logarithm of the volatility is given by $p(ln(h_t)|r^2_{1:t-1}, \mathbf{x}_{1:t-1})$, which includes a set of latent factors, denoted as z_t. A recurrent neural

network with hyperparameters(Θ_1), specifically an LSTM, encodes the squared returns r_t^2, the input features $\mathbf{x}_t$ and the previous latent factors z_{t-1}, generating the updated latent factors z_t. The likelihood $p(ln(h_t)|\theta(z_t))$ follows a Gaussian mixture distribution with parameters determined by a function $\theta(z_t, \Theta_2)$ of the network output z_t. The network architecture of the DeepAR-GMM-GARCH model is depicted in Figure 2.

3.1. The DeepAR-GMM-GARCH model

Due to the complex interplay between volatility and the factors that influence it, this paper's central model component declares that the volatility h_t of a time series at time t is derived from the latent variable z_{t-1} at time $t-1$, the square of return r_{t-1}^2 and the covariates $\mathbf{x}_{t-1}$. $p(\ln(h_t)|\theta(z_t))$ follows a Gaussian mixture distribution composed of K components. In the empirical analysis, $\mathbf{x}_{t-1}$ will be substituted with a vector of extreme values. A nonlinear mapping function g is used to establish this relationship. The DeepAR-GMM-GARCH model proposed in this paper is as follows.

$$
\begin{aligned}
&z_t = g(z_{t-1}, r_{t-1}^2, x_{t-1}, \Theta_1),\\
&\mu_{k,t} = log(1 + exp(w_{k,\mu}^T z_t + b_{k,\mu})),\\
&\sigma_{k,t} = log(1 + exp(w_{k,\sigma}^T z_t + b_{k,\sigma})),\\
&\pi_{k,t} = log(1 + exp(w_{k,\pi}^T z_t + b_{k,\pi})),\\
&P(ln(h_t)|z_t, \Theta_2) \sim \sum_{i=1}^{K} \pi_k N(\mu_{k,t}, \sigma_{k,t}),\\
&r_t = \varepsilon_t \sqrt{h_t},\\
&\sum_{i=1}^{K} \pi_k = 1.
\end{aligned}
\tag{3.2}
$$

The model can be viewed as a structure for nonlinear volatility prediction models since the conditional distribution of the perturbation ε_t in the model can be selected as $N(0,1)$ and $T(0,1,v)$. Consequently, this gives rise to two distinct models, referred to as DeepAR-GMM-GARCH and DeepAR-GMM-GARCH-t.

Assuming that the distribution of $p(ln(h_t)|\theta(z_t))$ follows a Gaussian distribution, model (3.2) will be reduced to a more simple version:

$$
\begin{aligned}
&z_t = g(z_{t-1}, r_{t-1}^2, x_{t-1}, \Theta_1),\\
&\mu_t = log(1 + exp(w_\mu^T z_t + b_\mu)),\\
&\sigma_t = log(1 + exp(w_\sigma^T z_t + b_\sigma)),\\
&P(ln(h_t)|z_t, \Theta_2) \sim N(\mu_t, \sigma_t^2),\\
&r_t = \varepsilon_t \sqrt{h_t}.
\end{aligned}
\tag{3.3}
$$

For similarity, we call the above as DeepAR-GARCH model.

3.2. Inference and learning

3.2.1. Algorithm

For a given time series, our goal is to estimate the parameters Θ_1 of the LSTM cells and the parameters Θ_2 in function θ which applies an affine transformation followed by a softplus activation. We employ a quasi-maximum likelihood estimation method with the likelihood function: $\Theta = argmax \sum_i logp(\widetilde{h_i}|\Theta_1, \Theta_2)$. Inferring from this likelihood function necessitates taking into account the latent variable z_t.

The flowchart for the training algorithm of the model is shown below. First, we utilize the BIC criterion to identify the number of classifications, K, for all samples. Each data point is assigned a label from 1 to K, and each cluster k has its mean vector and covariance matrix. Based on these findings, we establish the initial π_k as the proportion of data points labelled as k and set the initial mean vector μ_k and covariance matrix $\sum_k$ to the mean vector and covariance matrix within cluster k. As a result, we obtain the parameter values ($\theta = \widetilde{\pi}_{k,0}, \widetilde{\mu}_{k,0}, \widetilde{\sigma}^2_{k,0}$) from the initial cluster and use them to pretrain the DeepAR-GMM-GARCH model. This approach allows our model to converge quickly. Next, we partition the training sample data into multiple batches, select one sample from a batch and use the sample ($r^2_{t0-m}, \ldots, r^2_{t0-1}$) as the input for the DeepAR-GMM-GARCH model. The model calculates a set of $\widetilde{\pi}_{k,t}, \widetilde{\mu}_{k,t}, \widetilde{\sigma}^2_{k,t}$, after which we sample from this Gaussian mixture model, compute the loss, and update the parameters through gradient descent. Since direct differentiation of the sampling is infeasible, we apply the reparameterization trick to adjust the model's parameters. We continue this training process until the end of the training cycle. Last, we input the training set sample into our trained model for prediction evaluation. The model sequentially calculates the parameters for both the latent variable and the mixed Gaussian model and then proceeds with sampling. Ultimately, we provide the prediction results derived from the sampling outcomes.

The training algorithm is shown in the Algorithm 1.

Algorithm 1 Training Procedure for DeepAR-GMM-GARCH Mixture Model

for each batch **do**
  for each $t \in [t0 - m, t0 - 1]$ **do**
    if t is $t0 - m$ **then**
      $z_{t-1} = 0$
    else {t is not $t0 - m$}
      $z_t = g(z_{i-1}, r^2_{i-1}, x_{t-1}, \Theta_1)$
    end if
    for each $k \in [1, K]$ **do**
      $\widetilde{\mu}_{k,t} = log(1 + exp(w^T_{k,\mu} z_t + b_{k,\mu}))$
      $\widetilde{\sigma}^2_{k,t} = log(1 + exp(w^T_{k,\sigma} z_t + b_{k,\sigma}))$
      $\widetilde{\pi}_{k,t} = log(1 + exp(w^T_{k,\pi} z_t + b_{k,\pi}))$
    end for
    sample $ln(\widetilde{h_t}) \sim GMM(\widetilde{\pi}_{k,t}, \widetilde{\mu}_{k,t}, \widetilde{\sigma}^2_{k,t})$
  end for
  compute Loss, model parameters Θ_1, Θ_2 adjust using gradient descent method.
end for

3.2.2. Loss function

During the training process, the definition of the loss function determines the prediction quality of the model. We use the the average negative loglikelihood function as the loss function $Loss = L_h$. In the GARCH model, we usually assume that ε_t obeys a Gaussian distribution or a Student distribution, two loss functions are as follows:
(1) When $\varepsilon_t \sim N(0, 1)$, the loss function is:

$$L_h = -\frac{1}{N}\sum_{t=1}^{N}[\log(\widetilde{h}_t - \frac{r_t^2}{2\widetilde{h}_t})]. \tag{3.4}$$

(2) When $\varepsilon_t \sim t(0, 1, v)$, the loss function is:

$$L_h = -\frac{1}{N}\sum_{t=1}^{N}[\log(\widetilde{h}_t) + \frac{1}{2}(v+1)\log(1 + \frac{r_t^2}{\widetilde{h}_t(v-2)})]. \tag{3.5}$$

To calculate the above loss functions, we need to get samples for $\widetilde{h}_t$ based on algorithm1 given in Section 3.2.1. In practice, if ε_t follows other distributions, based on the idea of QMLE, we still can use the loss function given in (3.4) see Liu and So, 2020.

4. Simulation

Experiments are carried out on volatility inference using simulated time series. These series exhibit flexibility, with both volatility and mixing coefficients changing over time, as detailed below:

$$\begin{aligned}
&r_t = \varepsilon_t\sqrt{h_t}, \varepsilon_t \sim N(0,1),\\
&p\left(h_t|F_{t-1}\right) = \eta_{1,t}\phi\left(\mu_t, \sigma_{1,t}^2\right) + \eta_{2,t}\phi\left(\mu_t, \sigma_{2,t}^2\right),\\
&\mu_t = a_0 + a_1 h_{t-1},\\
&\sigma_{1,t}^2 = \alpha_{01} + \alpha_{11} r_{t-1}^2 + \beta_1\sigma_{1,t-1}^2,\\
&\sigma_{2,t}^2 = \alpha_{02} + \alpha_{12} r_{t-1}^2 + \beta_2\sigma_{2,t-1}^2,\\
&\pi_{1,t} = c_0 + c_1 h_{t-1},\\
&\eta_{1,t} = \exp\left(\pi_{1,t}\right) / \left(1 + \exp\left(\pi_{1,t}\right)\right),
\end{aligned} \tag{4.1}$$

where F_{t-1} denotes the information set through time $t-1$ and ϕ is the Gaussian density function. $\eta_{1,t}$ and $\eta_{2,t}$ are mixing coefficients of two gaussian distribution and satisfy: $\eta_{2,t} = (1 - \eta_{1,t})$. When generating the simulation data, we set: $\alpha_{01} = 0.01$, $\alpha_{11} = 0.1$, $\beta_1 = 0.15$, $\alpha_{02} = 0.04$, $\alpha_{12} = 0.15$, $\beta_2 = 0.82$, $c_0 = 0.02, c_1 = 0.90, a_0 = 0.02, a_1 = 0.6$. The time series has initial values:$r_0 = 0.1, \sigma_0^2 = 0$, $h_0 = 0$. The sample sizes of T = 500, 1000 and 1500 are considered, and the replication time is 1000.

For the series simulated from (4.1), we apply three models to forecast their volatility, namely, the GARCH model with $\epsilon_t \sim N(0, 1)$(GARCH-n), the GARCH model with$\epsilon_t \sim T(0, 1, v)$(GARCH-t) and the DeepAR-GMM-GARCH model. Using MCMC sampling method, the degree of freedom for the Student-t distribution is determined to be 6. For the DeepAR-GMM-GARCH model, we set a recurrent neural network with three LSTM layers and 24 hidden nodes. For the input nodes m in Figure 2, we

use the Grid Search Algorithm to find their optimal values. We use BIC rule to choose K based on the Mclust package [35]. Our model's hyperparameters are trained by the software Optuna, a commonly applied automatic hyperparameters optimization software.

Table 1 informs the three volatility forecasting models' in-sample errors (RMSE and MAE). The GARCH-n and GARCH-t show similar performance, and our DeepAR-GMM-GARCH model is outstanding; All the models get decreased RMSE as sample size increase. These results imply the proposed estimation can be asymptotically convergent. Table 2 reports the average out-of-sample errors of the three volatility forecasting models. Similar to the in-sample results, our DeepAR-GMM-GARCH model is superior to the GARCH-n and GARCH-t models.

Table 1. The average in-sample errors of the GARCH-n, the GARCH-t and the DeepAR-GMM-GARCH models.

sample size	Modle	RMSE	MAE
$T = 500$	GARCH-n	0.1364	0.0628
	GARCH-t	0.1211	0.0591
	DeepAR-GMM-GARCH	0.1068	0.0548
$T = 1000$	GARCH-n	0.0621	0.0437
	GARCH-t	0.0578	0.0419
	DeepAR-GMM-GARCH	0.0398	0.0331
$T = 1500$	GARCH-n	0.0604	0.0428
	GARCH-t	0.0652	0.0401
	DeepAR-GMM-GARCH	0.0300	0.0325

Note: Number of replications = 1000.

Table 2. The average out-of-sample errors of the GARCH-n, the GARCH-t and the DeepAR-GMM-GARCH models.

sample size	Modle	RMSE	MAE
$T = 500$	GARCH-n	0.3564	0.3028
	GARCH-t	0.3271	0.3091
	DeepAR-GMM-GARCH	0.2761	0.2311
$T = 1000$	GARCH-n	0.2619	0.2117
	GARCH-t	0.2318	0.2033
	DeepAR-GMM-GARCH	0.2091	0.1834
$T = 1500$	GARCH-n	0.2241	0.2179
	GARCH-t	0.2213	0.1971
	DeepAR-GMM-GARCH	0.1911	0.1722

Note: Number of replications = 1000.

5. Empirical anaysis

Comprehensive stock index represents the average of the economic performance of the whole financial market. In this section, we study the China Shanghai Shenzhen index(CSI 300 index) daily OHLC

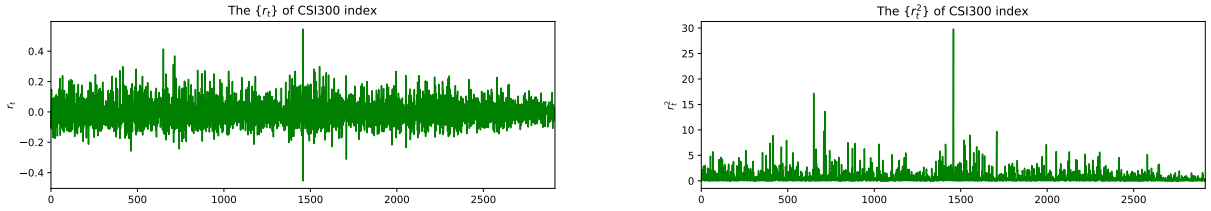
data. The OHLC data contains daily high, low, open and close prices. Scholars have pointed out that combining the open, high, low and close prices can obtain more effective volatility estimates. Hence, We also intruduce OHLC data to our mixture model.

5.1. CSI 300 data set

The data of the CSI 300 index studied in this paper are from January 4, 2010, to December 30, 2021, with a total of 2916 trading days. Let r_t be the returns of the corresponding series, which are calculated using the closing price C_t series of the CSI300 index.

$$r_t = 100 \log \frac{C_{t+1}}{C_t}. \tag{5.1}$$

The time series of r_t and r_t^2 are ploted in Figure 3. We can find that r_t^2 displays a significant volatility clustering characteristic, and the amplitude of volatility is gradually decreasing.



(a) r_t (b) r_t^2

Figure 3. The time series of r_t and r_t^2 for the CSI 300 index, as shown in (5.1), spans from January 4, 2010, to December 30, 2021.

The collected data is divided into training data and test data. Table 3 describes the statistical characteristics of training and test data, respectively. The mean of r_t is relatively small, only 0.007465. The standard deviation is 2.251006, which displays that the degree of variation is large. The skewness is less than 0, and the kurtosis is greater than 3, indicating that the sequence is of left deviation and heavy tail. The test data has the same characteristics as the training data, such as data dispersion, large volatility, left deviation and higher protrusion than the normal distribution. This may imply that the normal distribution may not be suitable for our data, and other heavy-tailed distributions, such as t distribution or mixed normal distribution, could be more suitable.

Table 3. Descriptive statistics for the training and test sets of CSI 300 returns.

Data Set	Period	Mean	Std.	Skew.	Kurt.
training data	04/01/2010 to 29/12/2017	0.007465	2.251006	−0.755580	5.136707
test data	02/01/2018 to 30/12/2021	0.019125	1.717302	−0.427819	3.248347

Besides the close price(C_t), we also introduce the high price(H_t), the open price(O_t) and the low price(L_t). Define $u_t = (H_t - O_t)^2$, $d_t = (L_t - O_t)^2$, $c_t = (C_t - O_t)^2$.

The correlation matrix below (5.2) shows the correlation coefficients between u_t, d_t, c_t and r_t^2. It can be found that there is large correlation coefficient between the pair (u_t,r_t^2), (d_t,r_t^2) and (c_t,r_t^2). From a common sense, large values for u_t, d_t and c_t, usually means large volatility(r_t^2). However, classical

volatility models do not take such extreme values into account. Consequently, it is reasonable to use neural network together with u_t, d_t and c_t to forecast volatility, because neural network can introduce additional covariates and capture the complex relation between different covariates.

$$\begin{bmatrix} & r_t^2 & u_t & d_t & c_t \\ r_t^2 & 1.000 & 0.394 & 0.475 & 0.716 \\ u_t & 0.394 & 1.000 & 0.012 & 0.556 \\ d_t & 0.475 & 0.012 & 1.000 & 0.690 \\ c_t & 0.716 & 0.556 & 0.690 & 1.000 \end{bmatrix} \tag{5.2}$$

5.2. Error measures

This paper uses four evaluation indicators to measure the predictive performance of the model, they are: NMAE, HR, linear correlation coefficient and rank correlation coefficient, which is defined as follows:

$$\text{NMAE} = \frac{\sum_{t=1}^{N} \left| r_{t+1}^2 - \widetilde{h}_{t+1} \right|}{\sum_{t=1}^{N} \left| r_{t+1}^2 - r_t^2 \right|}, \tag{5.3}$$

$$\begin{aligned} \text{HR} &= \frac{1}{N} \sum_{t=1}^{N} \theta_t, \\ \theta_t &= \begin{cases} 1 & : \quad \left(\widetilde{h}_{t+1} - r_t^2\right)\left(r_{t+1}^2 - r_t^2\right) \geq 0 \\ 0 & : \quad \text{else} \end{cases}, \end{aligned} \tag{5.4}$$

where N represents the number of predicted samples. Both NMAE and HR values range between 0 and 1. The smaller the values of these two indicators, the better the model's performance.

Scholars usually use high-frequency data volatility estimates as a proxy for actual volatility to evaluate forecasting models. We also use realized volatility($\sigma_{RV,t}^2$) as a proxy for actual volatility, calculated by summing up the squares of intra-day returns every 5 minutes.

$$\sigma_{RV,t}^2 = \sum_{i=1}^{48} [\log r_{t,i} - \log r_{t,i-1}]^2. \tag{5.5}$$

We focus on the out-of-sample predictive performance of the models, the correlation between realized volatilities $\sigma_{RV,t+1}^2$ and predicted volatilities $\widetilde{h}_{t+1}$ is measured only on the test set. We calculated Pearson's coefficient

$$r = \frac{\sum_{i=1}^{N} \left(\sigma_{RV,t+1}^2 - \sigma_{RV}^2\right)\left(\widetilde{h}_{t+1} - \widetilde{h}\right)}{\sqrt{\sum_{i=1}^{N} \left(\sigma_{RV,t+1}^2 - \sigma_{RV}^2\right)^2} \sqrt{\sum_{i=1}^{N} \left(\widetilde{h}_{t+1} - \widetilde{h}\right)^2}}, \tag{5.6}$$

where σ_{RV}^2 and $\widetilde{h}$ denote the respective mean values, and Spearman's rank order correlation coefficient r_s. r_s is also calculated using Eq (5.6). However, the actual volatilities are replaced by their ranks. Spearman's rank order correlation coefficient is considered more robust than Pearson's coefficient. r and r_s are both between -1 and 1. A value of $r(r_s)$ around 0 means that the realized volatilities and predicted volatilities are uncorrelated.

5.3. In-sample results and out-sample results

5.3.1. In-sample results

Simulation experiments demonstrate that our proposed model exhibits greater prediction accuracy than the GARCH model. In this section, to highlight the advantages of our model, we compare it to the classic GARCH model, ANN-GARCH (an existing neural network GARCH model) and the DeepAR-GARCH model using empirical data.

Among the four models, the GARCH and ANN-GARCH models predict conditional volatility, whereas the DeepAR-GARCH and DeepAR-GMM-GARCH models provide probabilistic forecasts for conditional volatility. To facilitate comparison, we will calculate the mean and quantiles of the probability density function for conditional volatility derived from the DeepAR-GARCH and DeepAR-GMM-GARCH models.

The estimated parameters of the GARCH models(GARCH-n and GARCH-t) are summarized in Tables 4 and 5. For the GARCH-t model, the degrees of freedom parameter v is estimated at around 6. The GARCH-n and GARCH-t models are all nearly estimated with higher values of β_1 and lower values of α_1. The sum of α_1 and β_1 is almost 1, Which implies the sequence may be non-stationary. Therefore, our model without the stationary constraint is more suitable. The ANN-GARCH model employs a three-layer ANN structure, featuring two input nodes, 24 nodes for the hidden layer and a single output node. Likewise, the DeepAR-GARCH and DeepAR-GMM-GARCH models also have a three-layer design, consisting of 14 input nodes, 24 nodes for the hidden layer and an output layer with two output nodes and five output nodes.

Table 4. Parameter estimation of GARCH-n model.

Data Set	a_0	α_1	β_1
$CSI300$	$1.8365e-04$	0.1000	0.8800

Table 5. Parameter estimation of GARCH-t model.

Data Set	a_0	α_1	β_1	v
$CSI300$	$1.8103e-04$	0.1000	0.8400	6.4625

Table 6. In-sample forecasting results for the GARCH-n, GARCH-t, DeepAR-GARCH and DeepAR-GMM-GARCH models.

Data Set	Model	Loss	NMAE	HR
In-sample	GARCH-n	1.401	0.763	0.704
	GARCH-t	1.331	0.761	0.637
	ANN-GARCH	1.603	0.827	0.690
	DeepAR-GARCH	1.541	**0.748**	0.717
	DeepAR-GMM-GARCH	**1.311**	0.751	**0.630**

Table 6 lists the performance of five volatility prediction models. In the in-sample study, the

DeepAR-GMM-GARCH model has the smallest HR and loss, and the DeepAR-GARCH model has the smallest NMAE. The volatility prediction performance of the DeepAR-GMM-GARCH model is better than the traditional GARCH and DeepAR models.

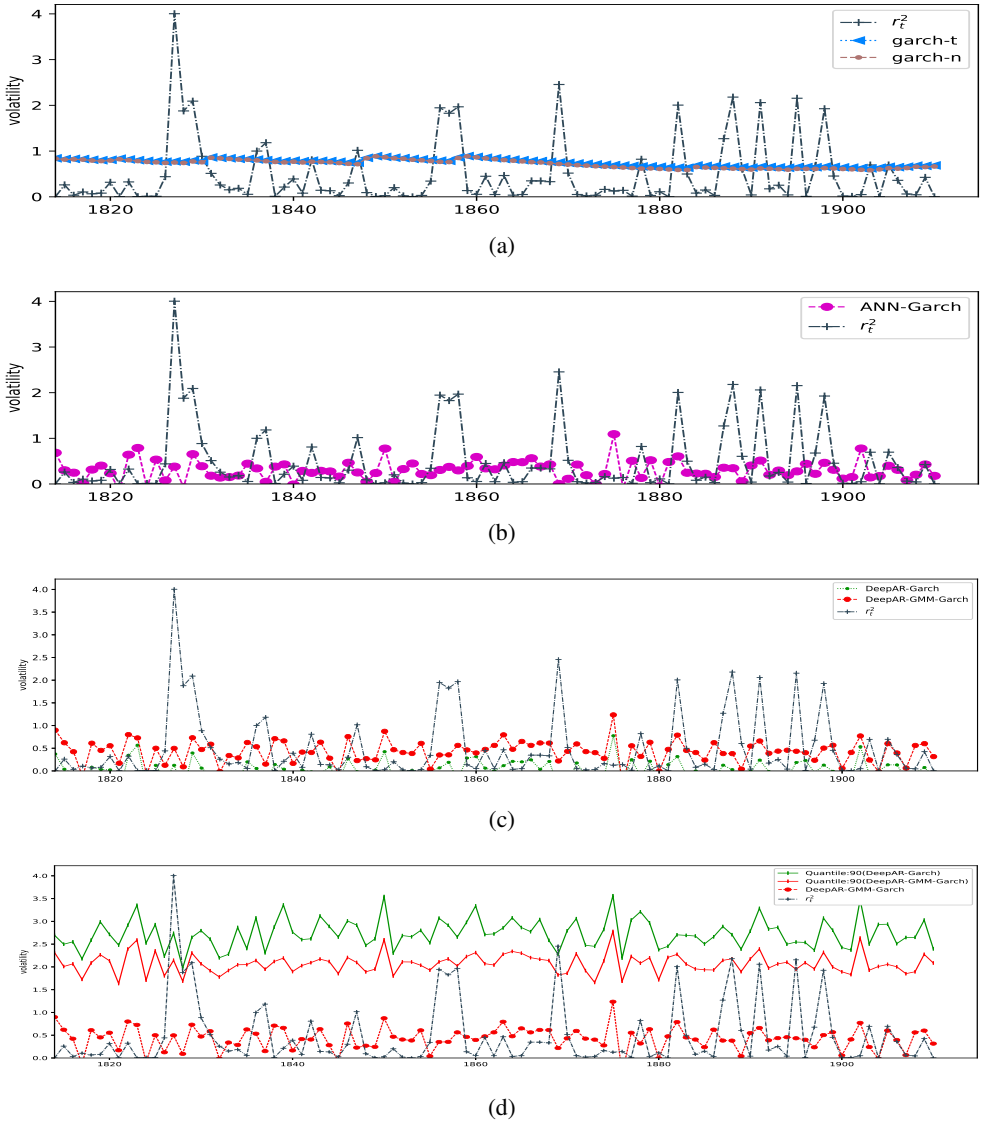


Figure 4. Subplots $(a), (b)$, (c) and (d) are the plots of the in-sample volatility forecasting results of the classic GARCH, ANN-GARCH, DeepAR-GARCH and DeepAR-GMM-GARCH models. (a) is the comparison of volatility forecasting results from the GARCH models. (b) is the comparison of volatility forecasting results between the ANN-GARCH model and r_t^2. (c) is the comparison of probabilistic forecasting models between the DeepAR-GARCH model and DeepAR-GMM-GARCH model. (d) is the comparison of 90% quantile forecasting results between the DeepAR-GARCH model and DeepAR-GMM-GARCH model.

In Figure 4, we display a portion of the forecasting results from various models and compare them

with r_t^2. As shown in (a), the forecasting results from the GARCH models differ from r_t^2. The GARCH models fail to capture significant changes in r_t^2. From (b),(c), it is evident that the neural network models capture the trend of r_t^2 and more accurately predict large fluctuations, with the DeepAR-GMM-GARCH model demonstrating the best performance. In (d), we observe that the estimated 90% quantiles from the DeepAR-GMM-GARCH model appear to be closer to the observations (r_t^2).

To sum up, from the estimation results of Table 6 and the plots in Figure 4, it is shown that introducing the extreme values(u_t,d_t,c_t) can help to improve the forecasting accuracy of the mixture volatility models. Hence the proposed approach is of particular practical value.

5.3.2. Out-sample results

In the out-of-sample analysis, as discussed in Section 5.1, the test set comprises the time series of 972 trading days subsequent to the respective training set.

Table 7. Out-of-sample forecasting results for the GARCH-n, GARCH-t, ANN-GARCH, DeepAR-GARCH and DeepAR-GMM-GARCH models.

Data Set	Model	Loss	NMAE	HR
Out-sample	GARCH-n	2.320	0.917	0.868
	GARCH-t	2.008	0.915	0.859
	ANN-GARCH	2.517	0.903	0.783
	DeepAR-GARCH	**1.916**	0.929	0.801
	DeepAR-GMM-GARCH	2.100	**0.790**	**0.722**

Table 7 presents the performance of the models using common error measures (loss function, NMAE and HR). The DeepAR-GARCH model attains a lower loss function value compared to the neural network models on the test set. The neural network models display lower NMAE and HR values than the GARCH model on the training set. The DeepAR-GMM-GARCH models exhibit the lowest NMAE and HR values on the test data set.

In Figure 5, we plot part of the forecasting results from the five models mentioned with out-sample data set and compare it with r_t^2. It can be seen from (a) that the GARCH models do not capture significant changes of r_t^2, the same as the in-sample results. For (b),(c), The neural network models capture most of the fluctuations of r_t^2 well, the DeepAR-GMM-GARCH model performing the best.

From (d), we could find that The estimated 90% quantiles from the DeepAR-GMM-GARCH model seem to be more closely aligned with the observations (r_t^2).

Section 5.2 mentions that the linear correlation r and the rank correlation r_s are two measures for comparing realized volatilities. The linear correlation r and the rank correlation r_s between predicted and realized volatilities of the test set are reported in Table 8. On average, the DeepAR-GMM-GARCH model shows the best performance of all models. It obtains the highest rank correlation on the test set. Rank correlation is more robust than linear correlation since it detects correlations nonparametrically.

Figure 5. Subplots $(a), (b)$, (c) and (d) are the plots of the out-sample volatility forecasting results of the classic GARCH, ANN-GARCH, DeepAR-GARCH and DeepAR-GMM-GARCH models. (a) is the comparison of volatility forecasting results from the GARCH models. (b) is the comparison of volatility forecasting results between the ANN-GARCH model and realized volatility(r_t^2). (c) is the comparison of probabilistic forecasting models between the DeepAR-GARCH model and DeepAR-GMM-GARCH model. (d) is the comparison of 90% quantile forecasting results between the DeepAR-GARCH model and DeepAR-GMM-GARCH model.

Table 8. A comparison of linear correlation (r) and rank correlation (r_s) between the realized volatility and volatility predicted by GARCH-n, GARCH-t, DeepAR-GARCH, ANN-GARCH and DeepAR-GMM-GARCH models for test data set on the CSI300 index. The best model (the highest correlation) is underlined.

Out-sample	GARCH-n	GARCH-t	ANN-GARCH	DeepAR-GARCH	DeepAR-GMM-GARCH
r	0.381	0.420	0.490	0.473	<u>0.504</u>
r_s	0.477	0.502	0.500	0.516	<u>0.527</u>

6. Conclusions

This paper studies a mixture volatility forecasting model based on the autoregressive neural work and the GARCH model to obtain more precise forecasting for the conditional volatility model. The inference, loss functions and training algorithm of the mixture model are given. The simulation results show that our model performs better with less error than the classic GARCH models. The empirical study based on the CSI300 index shows that our model can significantly improve the forecasting accuracy with extreme values compared to the usual models.

Our research findings can offer valuable insights into the prediction of volatility uncertainty. In future studies, our model can be employed for various high-frequency volatility analysis, where it is anticipated to exhibit enhanced performance.

Acknowledgments

This work is partially supported by Guangdong Basic and Applied Basic Research Foundation (2022A1515010046) and Funding by Science and Technology Projects in Guangzhou (SL2022A03J00654).

Conflict of interest

The authors declare no conflict of interest.

References

1. R. F. Engle, Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation, *Econometrica*, **50** (1982), 987–1007. https://doi.org/10.2307/1912773
2. T. Bollerslev, Generalized autoregressive conditional heteroskedasticity, *J. Econom.*, **31** (1986), 307–327. https://doi.org/10.1016/0304-4076(86)90063-1
3. D. B. Nelson, Conditional heteroskedasticity in asset returns: A new approach, *Econometrica*, **59** (1991), 347–370. https://doi.org/10.2307/2938260
4. L. R. Glosten, R. Jagannathan, D. E. Runkle, On the relation between the expected value and the volatility of the nominal excess return on stocks, *J. Financ.*, **48** (1993), 1779–1801. https://doi.org/10.1111/j.1540-6261.1993.tb05128.x

5. J. Hull, A. White, The pricing of options on assets with stochastic volatilities, *J. Financ.*, **42** (1987), 281–300. https://doi.org/10.1111/j.1540-6261.1987.tb02568.x
6. B. J. Blair, S. H. Poon, S. J. Taylor, Forecasting S&P 100 volatility:the incremental information content of implied volatilities and high-frequency index returns, in *Handbook of Quantitative Finance and Risk Management*, Springer, (2010), 1333–1344. https://doi.org/10.1007/978-0-387-77117-5
7. F. Audrino, D. Colangelo, Semi-parametric forecasts of the implied volatility surface using regression trees, *Stat. Comput.*, **20** (2010), 421–434. https://doi.org/10.1007/s11222-009-9134-y
8. C. Luong, N. Dokuchaev, Forecasting of realised volatility with the random forests algorithm, *J. Risk Financial Manag.*, **11** (2018), 61. https://doi.org/10.3390/jrfm11040061
9. S. Mittnik, N. Robinzonov, M. Spindler, Stock market volatility: identifying major drivers and the nature of their impact, *J. Bank Financ.*, **58** (2015), 1–14. https://doi.org/10.1016/j.jbankfin.2015.04.003
10. Z. Li, B. Mo, H. Nie, Time and frequency dynamic connectedness between cryptocurrencies and financial assets in China, *Int. Rev. Econ. Financ.*, **86** (2023), 46–57. http://dx.doi.org/10.1016/j.iref.2023.01.015
11. Z. Li, H. Dong, C. Floros, A. Charemis, P. Failler, Re-examining bitcoin volatility: a CAViaR-based approach, *Int. Rev. Econ. Financ.*, **58** (2022), 1320–1338. http://dx.doi.org/10.1080/1540496X.2021.1873127
12. Z. Li, C. Yang, Z. Huang, How does the fintech sector react to signals from central bank digital currencies, *Financ. Res. Lett.*, **50** (2022), 103308. http://dx.doi.org/10.1016/j.frl.2022.103308
13. Z. Li, L. Chen, H. Dong, What are bitcoin market reactions to its-related events, *Int. Rev. Econ. Financ.*, **73** (2021), 1–10. http://dx.doi.org/10.1016/j.iref.2020.12.020
14. T. Li, J. Wen, D. Zeng, K. Liu, Has enterprise digital transformation improved the efficiency of enterprise technological innovation? A case study on Chinese listed companies, *Math. Biosci. Eng.*, **19** (2022), 12632–12654. http://dx.doi.org/10.3934/mbe.2022590
15. Y. Liu, P. Failler, Z. Liu, Impact of environmental regulations on energy efficiency: a case study of China's air pollution prevention and control action plan, *Sustainability*, **14** (2022), 3168. http://dx.doi.org/10.3390/su14063168
16. Y. Liu, Z. Li, M. Xu, The influential factors of financial cycle spillover: evidence from China, *Emerg. Mark. Financ. Tr.*, **56** (2020), 1336–1350. http://dx.doi.org/10.1080/1540496X.2019.1658076
17. D. G. Kirikos, An evaluation of quantitative easing effectiveness based on out-of-sample forecasts, *Natl. Account. Rev.*, **4** (2022), 378–389. https://dx.doi.org/10.3934/NAR.2022021
18. J. Saleemi, COVID-19 and liquidity risk, exploring the relationship dynamics between liquidity cost and stock market returns, *Natl. Account. Rev.*, **3** (2021), 218–236. https://dx.doi.org/10.3934/NAR.2021011
19. S. A. Gyamerah, B. E. Owusu, E. K. Akwaa-Sekyi, Modelling the mean and volatility spillover between green bond market and renewable energy stock market, *Green Finance*, **4** (2022), 310–328. https://dx.doi.org/10.3934/GF.2022015

20. H. Siddiqi, Financial market disruption and investor awareness: the case of implied volatility skew, *Quant. Finance Econ.*, **6** (2022), 505–517. https://dx.doi.org/10.3934/QFE.2022021

21. L. Li, X. Zhang, Y. Li, C. Deng, Daily GARCH model estimation using high frequency data, *J. Guangxi Norm. Univ., Nat. Sci.*, **39** (2021), 1181–1191.

22. S. A. Hamid, Z. Iqbal, Using neural networks for forecasting volatility of S&P 500 index futures prices, *J. Bus. Res.*, **57** (2004), 1116–1125. https://doi.org/10.1016/S0148-2963(03)00043-2

23. I. E. Livieris, E. Pintelas, P. Pintelas, A CNN-LSTM model for gold price time-series forecasting, *Neural Comput. Appl.*, **32** (2020), 17351–17360. https://doi.org/10.1007/s00521-020-04867-x

24. C. L. Dunis, X. Huang, Forecasting and trading currency volatility: an application of recurrent neural regression and model combination, *J. Forecast.*, **21** (2002), 317–354. https://doi.org/10.1002/for.833

25. R. G. Donaldson, M. Kamstra, An artificial neural network-GARCH model for international stock return volatility, *J. Empir. Financ.*, **4** (1997), 17–46. https://doi.org/10.1016/S0927-5398(96)00011-4

26. T. H. Roh, Forecasting the volatility of stock price index, *Expert Syst. Appl.*, **33** (2007), 916–922. https://doi.org/10.1016/j.eswa.2006.08.001

27. M. Bildirici, Ö. Ö. Ersin, Improving forecasts of GARCH family models with the artificial neural networks:An application to the daily returns in Istanbul Stock Exchange, *Expert Syst. Appl.*, **36** (2009), 7355–7362. https://doi.org/10.1016/j.eswa.2008.09.051

28. E. Hajizadeh, A. Seifi, M. H. F. Zarandi, I. B. Turksen, A hybrid modeling approach for forecasting the volatility of S&P 500 index return, *Expert Syst. Appl.*, **39** (2012), 431–436. https://doi.org/10.1016/j.eswa.2011.07.033

29. W. Kristjanpoller, M. C. Minutolo, Gold price volatility: A forecasting approach using the Artificial Neural Network-GARCH model, *Expert Syst. Appl.*, **42** (2015), 7245–7251. https://doi.org/10.1016/j.eswa.2015.04.058

30. N. Nikolaev, P. Tino, E. Smirnov, Time-dependent series variance learning with recurrent mixture density networks, *Neurocomputing*, **122** (2013), 501–512. https://doi.org/10.1016/j.neucom.2013.05.014

31. H. Y. Kim, C. H. Won, Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models, *Expert Syst. Appl.*, **103** (2018), 25–37. https://doi.org/10.1016/j.eswa.2018.03.002

32. W. K. Liu, M. K. P. So, A GARCH model with artificial neural networks, *Information*, **11** (2020), 489. https://doi.org/10.3390/info11100489

33. D. Salinas, V. Flunkert, J. Gasthaus, T. Januschowski, DeepAR: Probabilistic forecasting with autoregressive recurrent networks, *Int. J. Forecast.*, **36** (2020), 1181–1191. https://doi.org/10.1016/j.ijforecast.2019.07.001

34. P. Glasserman, D. Pirjol, W-shaped implied volatility curves and the Gaussian mixture model, *Quant. Financ.*, **36** (2021), 1–21. https://doi.org/10.1080/14697688.2023.2165448

35. L. Scrucca, M. Fop, T. B. Murphy, A. E. Raftery, mclust 5: clustering, classification and density estimation using Gaussian finite mixture models, *R J.*, **8** (2016), 289–317. https://doi.org/10.32614/RJ-2016-021

AIMS Press