creative commons

COMMONS DEED

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

이학석사학위논문

Monitoring conditional volatility change of GARCH time series based on Neural Network

GARCH 시계열에서 신경망 모델을 이용한 조건부 변동성의 변화 탐지 모니터링에 관한 연구

2023 년 2 월

서울대학교 대학원

통계학과

엄 규 원

Monitoring conditional volatility change of GARCH time series based on Neural Network

GARCH 시계열에서 신경망 모델을 이용한 조건부 변동성의 변화 탐지 모니터링에 관한 연구

지도교수 이 상 열

이 논문을 이학석사 학위논문으로 제출함

2022 년 10 월

서울대학교 대학원

통계학과

엄 규 원

엄규원의 이학석사 학위논문을 인준함

2022 년 12 월

위 원 장	이재용	(인)
부위원장	이상열	(인)
위 원	PARK JUN YONG	(인)

Abstract

In this paper, we discuss a method for detecting and monitoring the point where conditional volatility changes in the ARMA-GARCH time series using a sequential neural network model. In a stationary time series, we predict the future values and volatilities with the values of the previous point in time using a sequential neural network model such as LSTM or GRU. After creating test statistics with the predicted values obtained from the previous process, we perform a monitoring process based on the CUSUM test to detect points that proceed differently from the previous ones, especially the point where conditional volatility increases rapidly. In the process, we find appropriate hyperparameters through a grid search, then apply the monitoring process in simulation data and stock price data S&P 500 and the KOSPI index, and analyze the results finally.

Keywords: Monitoring, GRU, ARMA-GARCH, Neural Network, Conditional Volatility, financial market

Student Number: 2021-25859

Contents

Abstract **i**

Chapter 1 Introduction **1**

Chapter 2 Model Description **3**

2.1 Neural Network 3

2.1.1 RNN(Recurrent Neural Network) 3

2.1.2 LSTM(Long Short-Term Memory) 4

2.1.3 GRU(Gated Recurrent Unit) 4

2.2 Monitoring on heteroscedastic location-scale time series 5

Chapter 3 Simulation Study **8**

3.1 Selecting optimal parameters 9

3.2 Monitoring results in simulation 10

Chapter 4 Real Data Analysis **17**

Chapter 5 Conclusions **22**

Bibliography **23**

국문초록 **26**

List of Tables

Table 3.1 Set of tuning parameter for grid search 10
Table 3.2 Optimal parameters 11

List of Figures

Figure 3.1 Test set : y_t vs $\hat{g}_t(\hat{\mu}_l)$ 11

Figure 3.2 Test set : h_t vs $\hat{h}_t(\hat{\theta}_l)$ 11

Figure 3.3 (α, β, ω) : $(0.2, 0.2, 0.2) \rightarrow (0.2, 0.7, 0.2)$ 13

Figure 3.4 (α, β, ω) : $(0.2, 0.2, 0.2) \rightarrow (0.7, 0.2, 0.2)$ 14

Figure 3.5 (α, β, ω) : $(0.2, 0.2, 0.2) \rightarrow (0.2, 0.2, 0.9)$ 14

Figure 3.6 (α, β, ω) : $(0.2, 0.7, 0.2) \rightarrow (0.2, 0.2, 0.2)$ 15

Figure 3.7 (α, β, ω) : $(0.7, 0.2, 0.2) \rightarrow (0.2, 0.2, 0.2)$ 15

Figure 3.8 (α, β, ω) : $(0.2, 0.2, 0.9) \rightarrow (0.2, 0.2, 0.2)$ 16

Figure 3.9 (α, β, ω) : $(0.2, 0.2, 0.2) \rightarrow (0.2, 0.7, 0.5)$ 16

Figure 4.1 S&P500 index 19

Figure 4.2 KOSPI index 19

Figure 4.3 Monitoring result in KOSPI 20

Figure 4.4 Monitoring result in S&P500 20

Figure 4.5 Monitoring result in KOSPI 21

Figure 4.6 Monitoring result in S&P500 21

Chapter 1

Introduction

The problem of detecting parameter changes in statistical models has been of interest to many researchers. It originally started with quality control, but as we often experience the situation in which the structure of statistical models changes due to external factors such as social issues and policy, the problem of detecting a model change has been explored in financial time series. On this background, Lee et al. (2003) conducted a study on how to test changes in parameters and autocovariance functions in time series models. Lee and Song (2008), Oh and Lee (2018), and Song and Kang (2018) tested changes in parameters in the ARMA-GARCH model.

However, these studies have limitations in detecting changes over a fixed period in already observed data. In the financial market, it is important to detect and respond to instability by monitoring rapidly the fluctuations in stock prices or exchange rates. Leisch et al. (2000), Horváth et al. (2004), and Zeileis et al. (2005) conducted a study to monitor a change in parameters in regression models, Berkes et al. (2004a) expanded to monitor changes in GARCH models,

and Na et al. (2011) proposed a procedure to monitor changes in parameters and autocovariance functions of time series models. In addition, Gombay and Serban (2009) conducted monitoring studies in AR models, and Dienes and Aue (2014) and Aue et al. (2015) conducted the task in ARMA models. The method of finding parameter changes in observed data using the cumulative sum (CUSUM) based on residuals showed excellent performance in GARCH models (Kim et al., 2000; Lee et al., 2004). Oh and Lee (2018) and Song and Kang (2018) proposed a method using score vectors in the ARMA-GARCH model, particularly, a two-dimensional test statistic in the former's study. Oh and Lee (2019) devised new statistics that can detect changes in both scale and location parameters using the CUSUM of the squared residuals. The method based on the modified residuals is much simpler to calculate than the method based on the score vector, and the performance is better when the error term deviates significantly from the normal distribution in a rather complex model with many parameters.

Therefore, in this study, the CUSUM method based on the modified residuals proposed by Oh and Lee (2019) is applied to the monitoring procedure for the location-scale heteroscedastic models, where the residuals are obtained based on a neural network (NN) model. The NN model is used to check whether the monitoring procedure is well implemented through hyperparameter tuning. Throughout an empirical simulation study, we design an appropriate NN model, adjust hyperparameters, and check whether monitoring works well in the actual daily stock prices data such as S&P 500 and KOSPI. Especially, we detect the points where stock prices change rapidly due to economic recession. This paper consists of four chapters. Chapter 2 introduces monitoring procedures in sequential neural network models, and chapters 3 and 4 conduct a simulation study and real data analysis. Chapter 5 provides conclusions.

Chapter 2

Model Description

2.1 Neural Network

2.1.1 RNN(Recurrent Neural Network)

The single RNN model is presented as follows:

$$h_t = g(Wx_t + Uh_{t-1} + b) \tag{2.1}$$

where x_t is the m - dimensional input vector at time t, h_t is the n-dimensional hidden state, g is the activation function, such as the logistic function, the hyperbolic tangent function, or the Rectified Linear Unit (ReLU) and W, U and b are the appropriately sized parameters (two weights and a bias). Specifically, in this case, W is an $n \times m$ matrix, U is an $n \times n$ matrix, and b is an $n \times 1$ vector. It is well known that RNN is difficult to capture long-term dependencies because the stochastic gradients tend to either vanish or explode with long sequences.

2.1.2 LSTM(Long Short-Term Memory)

The LSTM model uses the computation of equation (2.1) as an intermediate candidate for the internal memory cell state $\tilde{c}_t$ and adds it in a weighted sum to the previous value of the internal memory state c_{t-1} to produce the current value of the memory cell c_t. This can be expressed in the following equations:

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{2.2}$$

$$\tilde{c}_t = g(W_c x_t + U_c h_{t-1} + b_c) \tag{2.3}$$

$$h_t = o_t \odot g(c_t) \tag{2.4}$$

The g is a nonlinear activation function typically used as the hyperbolic tangent function or Rectified Linear Unit (ReLU). The weighted sum is implemented in equation (2.2) through element-wise Hadamard multiplication denoted by $\odot$ to gating signals. The gating signals are the input, forget, and output signals denoted i_t, f_t, and o_t, respectively.

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{2.5}$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{2.6}$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{2.7}$$

2.1.3 GRU(Gated Recurrent Unit)

The GRU reduces the gating signals to two from the LSTM model. The two gates are named an update gate z_t and a reset gate r_t.

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

$$\tilde{h}_t = g(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h)$$

with two gates presented as:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r)$$

In the case of LSTM, there were three gates: a forget gate, an input gate, and an output gate, but GRU uses only two gates: a reset gate and an update gate. In addition, the cell state and hidden state are combined to express one hidden state. GRU has a lower number of parameters than LSTM, which makes training time faster than LSTM with no disadvantage in prediction performance.

2.2 Monitoring on heteroscedastic location-scale time series

Next, take a look monitoring process idea. In this study, we use the method of Lee and Kim (2022). Consider the time series model as follows:

$$y_t = g_t(\mu_0) + {h_t}^{\frac{1}{2}}(\theta_0)\eta_t \tag{2.8}$$

where $g_t(\mu_0) = g_t(y_{t-1}, y_{t-2}, ...; \mu)$ and $h_t(\theta_0) = h_t(y_{t-1}, y_{t-2}, ...; \theta)$ are the conditional mean and variance. The $\{\eta_t\}$ is an iid error process with zero mean and finite variance. Also, $\{\eta_t\}$ is independent with y_t. A typical example is the ARMA(m,n)-GARCH(p,q) model. It is defined as follows:

$$\begin{cases} y_t = \mu + \sum_{i=1}^{m} \phi_i y_{t-i} + \sum_{j=1}^{n} \theta_j \epsilon_{t-j} + \epsilon_t, \ \epsilon_t = {h_t}^{1/2}\eta_t \\ h_t = \omega + \sum_{i=1}^{p} \alpha_i \epsilon_{t-j}^2 + \sum_{j=1}^{q} \beta_j h_{t-i}, \end{cases} \tag{2.9}$$

where $\{\eta_t\}$ is an iid error process with mean zero and finite variance. Next, to detect a change of model parameters $\Theta = (\mu, \theta)^T$ using y_t, we set up null and alternative hypothesis:

$$H_0 : \Theta_t = \Theta_0, \quad t = 1, \ldots, n \quad vs.$$

$$H_1 : \begin{cases} \Theta_t = \Theta_0, & t = 1, \ldots, k \\ \Theta_t = \Theta_1, & t = k+1, \ldots, n \end{cases} \quad \text{for some } k = 1, \ldots, n-1 \tag{2.10}$$

with $\Theta_1 \neq \Theta_0$. To conduct a test (2.3), we consider bivariate time series referring to Oh and Lee(2019):

$$\xi_t(\theta) = (g_t(\mu)\eta_t(\theta), {\eta_t}^2(\theta) - 1)^T \tag{2.11}$$

with $\eta_t(\theta) = (y_t - g_t(\mu))/{h_t}^{1/2}(\theta)$ is residual. As $\eta_t(\theta)$ is not observable, we take

$$\hat{\eta_t}(\theta) = (y_t - \hat{g_t}(\mu))/\hat{h_t}^{1/2}(\theta)$$

where $\hat{g_t}(\mu)$ and $\hat{h_t}(\theta)$ is predicted via NN model. Under proper conditions, we get the basic process:

$$\hat{W_k} = \hat{K_n}^{-1/2} \sum_{t=1}^{k} \hat{\xi_t}$$

with

$$\hat{\xi_t}(\theta) = (\hat{g_t}(\hat{\mu_l})\hat{\eta_t}(\hat{\theta_l}), \hat{\eta_t}^2(\hat{\theta_l}) - 1)^T, \hat{K_n} = \frac{1}{l}\sum_{t=1}^{l} \hat{\xi_t}' \hat{\xi_t}'^T,$$

where $\hat{\xi_t}'$ is obtained from training set ${y_1}', \ldots, {y_l}'$. Then, we construct the monitoring process using $\hat{T_n}(k) = \max\left\{\hat{T_{n1}}(k), \hat{T_{n2}}(k)\right\}$ with

$$\begin{cases} T_{n1}(k) = \frac{1}{\sqrt{n}} \parallel \max_{m \leq k} \hat{W_m} - \hat{W_k} \parallel_{\max} \\ T_{n2}(k) = \frac{1}{\sqrt{n}} \parallel \min_{m \leq k} \hat{W_m} - \hat{W_k} \parallel_{\max} \end{cases}$$

where for any vector process $\mathbf{x}_m = (x_{m1}, \ldots, x_{mp})^T \in \mathbf{R}^2$, $\max_{m \le k} \mathbf{x}_m (\min_{m \le k} \mathbf{x}_m)$ denotes the vector whose ith entry is equal to $\max_{m \le k} \mathbf{x}_{mi} (\min_{m \le k} \mathbf{x}_{mi})$ for $i = 1, \ldots, p$, and for $\mathbf{y} = (y_1, \ldots, y_p)^T \in \mathbf{R}^p$, $\| \mathbf{y} \|_{\max} = \max_{1 \le k \le p} |y_k|$. Then, by Donker's invariance principle and the continuous mapping theorem, the following holds:

$$\hat{T_n} := \max_{1 \le k \le n} \hat{T_n}(k) = \max_{1 \le k \le n} (\hat{T_{n1}}(k) \vee \hat{T_{n2}}(k)) \xrightarrow{d} {T_2}^* := {T_{21}}^* \vee {T_{22}}^* \tag{2.12}$$

with

$$\begin{cases} {T_{21}}^* = \sup_{0 \le s \le 1} | \sup_{0 \le u \le s} B_2(u) - B_2(s)| \\ {T_{22}}^* = \sup_{0 \le s \le 1} | \inf_{0 \le u \le s} B_2(u) - B_2(s)|, \end{cases}$$

where B_2 denotes a two-dimensional standard Brownian motion. It means that H_0 is rejected if $T_n > C(\alpha)$, where $C(\alpha)$ denotes the control limit for significance level $0 < \alpha < 1$. In addition, if $\hat{T_n}(k)$ crosses the control limit $C(\alpha)$ at some point $k = 1, \ldots, n$, an anomaly is detected. Since $\hat{T_n}$ is a combination of $\hat{T_{n1}}$ and $\hat{T_{n2}}$, it can detect not only the increasing process but also decreasing process. However, it is important to find a point where the volatility increases rapidly in the financial market, so the primal purpose is to find a point where the conditional variance increases in a simulation study and real data.

Chapter 3

Simulation Study

In this section, we check the performance of the monitoring process using NN. To speed up the model training, we use the GRU model. Using the GRU model, we try to fit the simulation-generated stationary time series model and find the hyperparameters that enable rational prediction via grid search. Since the time series model is motivated from ARMA(1,1)-GARCH(1,1), which is a relatively simple model, the NN prediction model structure is also designed appropriately simple. The parameters to build a model are as follows.

- input shape(= steps): size or shape of input data. In this study, it is equivalent to time step j in $y_{t-1}, y_{t-2}, \ldots, y_{t-j}$ to predict $\hat{y}_t$. For simplicity, let it be steps.
- units: the dimension of the hidden layer. If the number of units is too small(or too large), it is likely to happen underfitting(or overfitting).
- epoch: one epoch means that training is completed once for the entire data set. If the epoch number is too small, underfitting can happen. Conversely,

if it is too large, the risk of overfitting increases, and the training time would be longer.

- batch size: the size of the data sample assigned for each batch, which is a dataset divided for smooth learning. 32, 64, and 128 are used generally.
- estimators: maximum number of base models for boosting. If the NN structure is simple, the trained model may be biased and result in unstable prediction. To prevent this situation, a boosting technique that generates and combines multiple base models with the same structure can be used.
- scaler: feature scaling function. In general, when two or more features are input, there is a risk of biased prediction if the units or distributions of each feature value are different. In addition, the NN may not be well trained when the input value is not balanced, or the input value is too small or large. For this reason, scaling input value is required before model training or prediction. The scaling function includes MinMax, Robust, Standard, etc.
- loss: loss function that evaluates the performance of the model. The smaller the loss, the model performance is better, and the loss function can be Mean Square Error(MSE), Mean Absolute Error(MAE), and Root Mean Square Error(RMSE).
- optimizer: method of updating a model from loss obtained through loss function. It can be SGD, Adam, and RMSprop.

3.1 Selecting optimal parameters

In this section, we find the optimal parameters of the conditional mean and variance model via the grid search method. First, we generate 2000 simulation

data (y_t) following the ARMA(1,1)-GARCH(1,1) model. Among them, the previous 1500 and the later 500 data have different parameters and movements on the graph (the most noticeable difference is the variance of y_t). Since it is impossible to find the predictions of conditional mean and conditional variance at the same time with the NN model designed in the study, the prediction values of conditional mean are first derived with one NN(mean NN model). And the residuals of the true and predictive values generate the time series of conditional variance. Then the predicted value of conditional variance is obtained with another NN(variance NN model). The hyperparameters obtained from the NN model and the comparison of model predictions and actual values are as follows.

Tuning Parameter	mean model	variance model
steps	(1, 2, 4, 6, 10)	(1, 2, 4, 6, 10)
scaler	(MinMax, Standard, Robust)	(MinMax, Standard, Robust)
epochs	(5, 10, 20, 40, 60)	(5, 10, 20, 40, 60)
batch size	(16, 32, 64, 128)	(16, 32, 64, 128)
estimators	(2,5,10,50,100,200)	(2,5,10,50,100,200)
units	(5, 10, 20, 50, 100, 200, 300)	(5, 10, 20, 50, 100, 200, 300)

Table 3.1 Set of tuning parameter for grid search

3.2 Monitoring results in simulation

In this section, we calculate the statistic $\hat{T_n}(k)$ via NN predictions $\hat{g}_t$, $\hat{h_t}$ and residuals $\hat{\eta}_t$ obtained from the simulation data and check the monitoring process

tuning parameter	mean model	variance model
steps	4	2
scaler	Robust	MinMax
epochs	40	40
batch size	32	32
estimators	2	2
units	200	200

Table 3.2 Optimal parameters

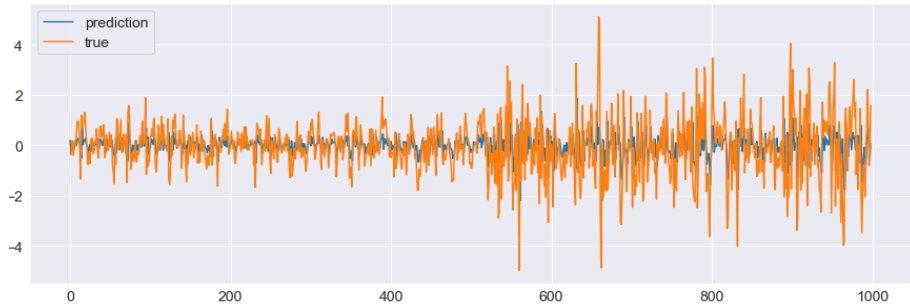

Figure 3.1 Test set : y_t vs $\hat{g}_t(\hat{\mu}_l)$

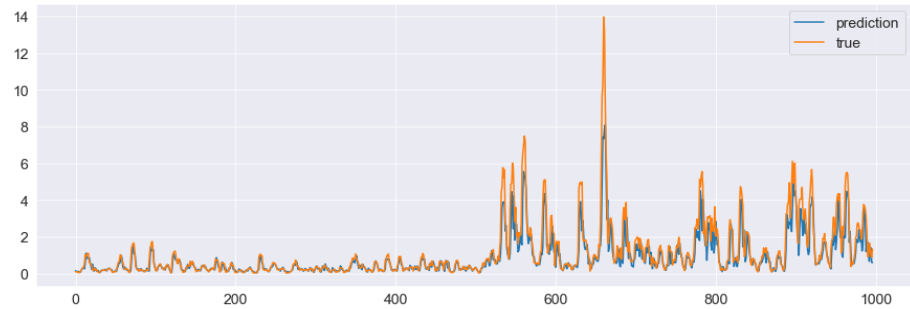

Figure 3.2 Test set : h_t vs $\hat{h}_t(\hat{\theta}_l)$

based on them. The specific calculation process of $\hat{T}_n(k)$ is referred to in the previous chapter, 2.2. In addition, the critical value of the significance level at $\alpha = 0.05$ in the test (2.3) uses the value c $= 2.689$, obtained by Monte Carlo simulations. Therefore, if the value of $\hat{T}_n(k)$ exceeds 2.689, we conclude that there is a change in the structure of the time series at $t = k$, and if the value of $\hat{T}_n := \max_{1 \le k \le n} \hat{T}_n(k)$ is less than c $= 2.689$, we conclude that there is no change. Since the primal purpose of this study is to detect changes in conditional volatility, we set a change point in the test data with the parameter shift of the GARCH model. For the change point, we select $t = [n/2]$, which is the middle point of the test set. In addition, the size of the train set and the size of the test set are given as $l = 1000$ and $n = 1000$.

- Case 1 : (α, β, ω) : $(0.2, 0.2, 0.2) \to (0.2, 0.7, 0.2)$
- Case 2 : (α, β, ω) : $(0.2, 0.2, 0.2) \to (0.7, 0.2, 0.2)$
- Case 3 : (α, β, ω) : $(0.2, 0.2, 0.2) \to (0.2, 0.2, 0.9)$
- Case 4 : (α, β, ω) : $(0.2, 0.7, 0.2) \to (0.2, 0.2, 0.2)$
- Case 5 : (α, β, ω) : $(0.7, 0.2, 0.2) \to (0.2, 0.2, 0.2)$
- Case 6 : (α, β, ω) : $(0.2, 0.2, 0.9) \to (0.2, 0.2, 0.2)$
- Case 7 : (α, β, ω) : $(0.2, 0.2, 0.2) \to (0.2, 0.7, 0.5)$

The graphs show the results of monitoring in each case. The orange and blue graphs represent actual values and NN predictions, respectively. And the black dotted line in the center, i.e., $t = [n/2]$, is the parameter change point and the green line is the point where $\hat{T}_n(k)$ starts to exceed the critical value, that is, the point where the parameter of the model changes at the significance level of

0.05. According to the results, the performance is good when the conditional volatility increases in the data (cases 1, 2, 3, 7), but the performance is poor when the conditional volatility decreases (cases 4, 5, 6). Perhaps in the sections where the conditional volatility is high, the training error value also has a high variance, which results in poor monitoring performance in the cases where the volatility decreases. If we increase the length of the training set or increase the learning rate to reduce errors, we can detect changes in parameters with better performance. In addition, changing the structure of models or tuning hyperparameters can sensitively respond to test statistics even if the volatility decreases, but it may cause type 1 errors, so we should try it carefully. Of course, in the actual stock price data, the above-mentioned problem is trivial. But in situations where it is important to detect variance shrink, we may take the risk of type 2 errors.

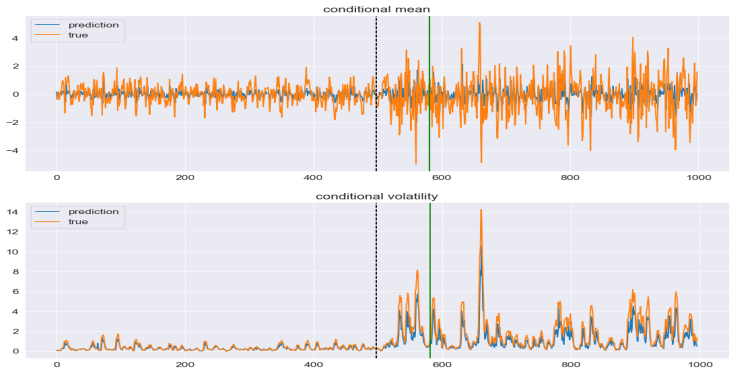

Figure 3.3 (α, β, ω) $: (0.2, 0.2, 0.2) \rightarrow (0.2, 0.7, 0.2)$

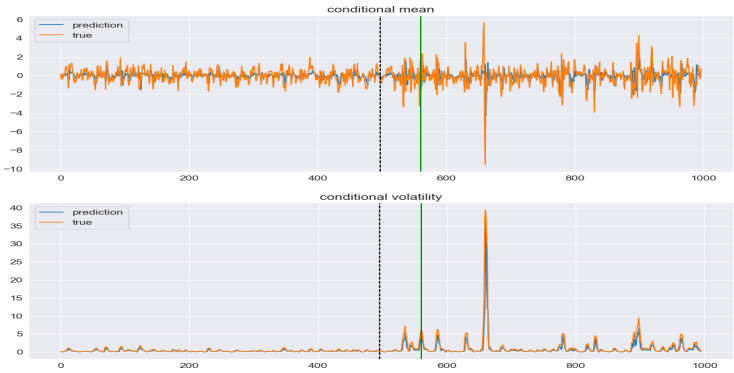

Figure 3.4 (α, β, ω) $: (0.2, 0.2, 0.2) \rightarrow (0.7, 0.2, 0.2)$

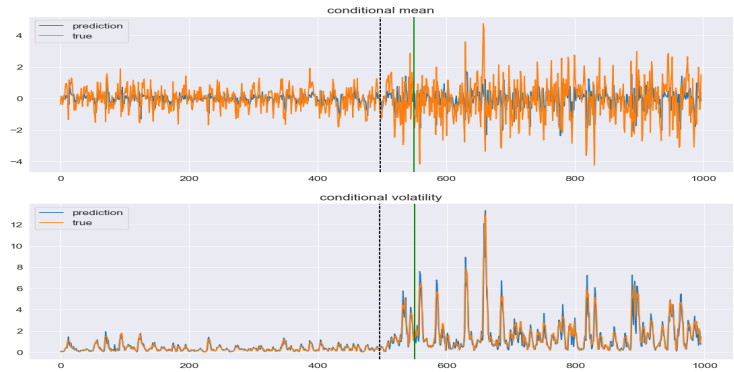

Figure 3.5 (α, β, ω) $: (0.2, 0.2, 0.2) \rightarrow (0.2, 0.2, 0.9)$

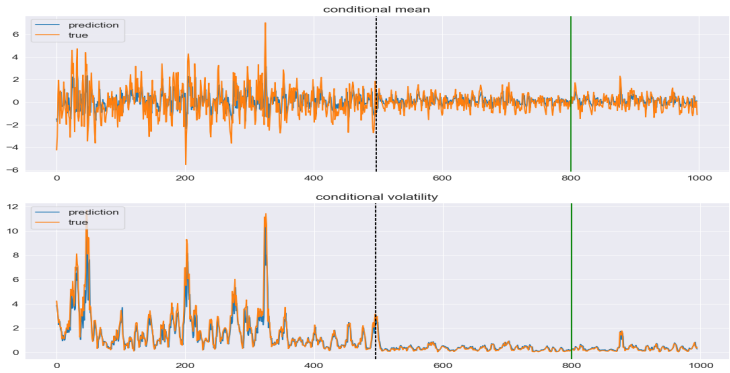

Figure 3.6 (α, β, ω) $: (0.2, 0.7, 0.2) \rightarrow (0.2, 0.2, 0.2)$

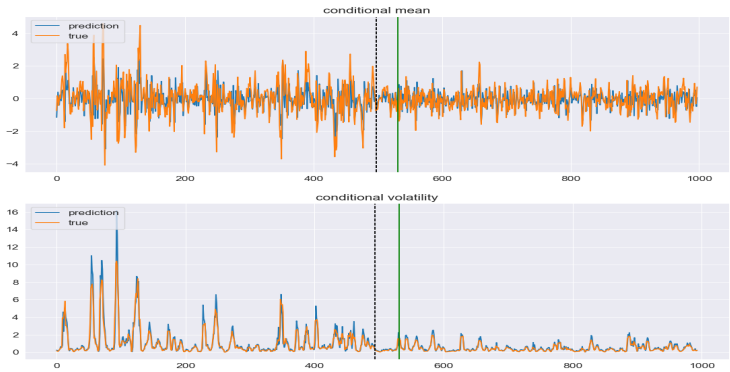

Figure 3.7 (α, β, ω) $: (0.7, 0.2, 0.2) \rightarrow (0.2, 0.2, 0.2)$

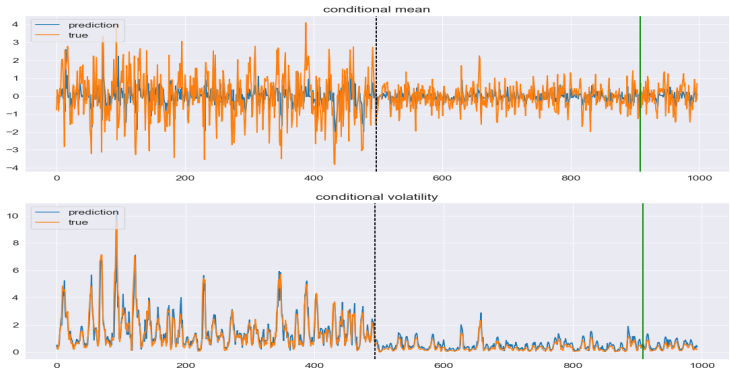

Figure 3.8 (α, β, ω) $: (0.2, 0.2, 0.9) \rightarrow (0.2, 0.2, 0.2)$

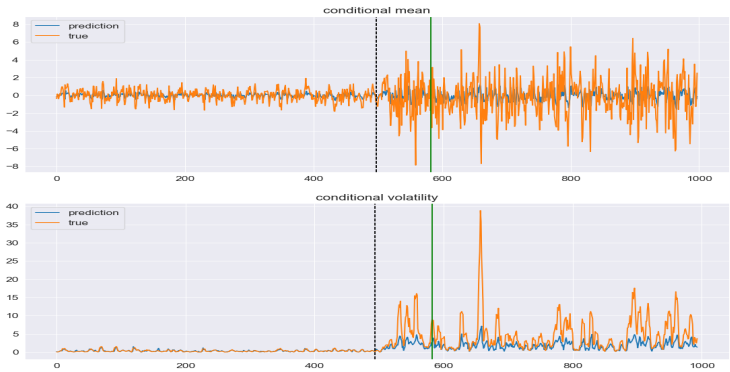

Figure 3.9 (α, β, ω) $: (0.2, 0.2, 0.2) \rightarrow (0.2, 0.7, 0.5)$

Chapter 4

Real Data Analysis

In this section, we aim to detect a point where stock price volatility changes rapidly in the stock price data of S&P500 and KOSPI using the NN monitoring method. We can obtain them from "Investing.com", and daily closing prices for about 20 years are used for analysis. Since stock price data does not satisfy stationarity, the rate of return (or log return) is derived through differencing closing prices as follows:

$$y_t = 100 \times (\log z_t - \log z_{t-1}) \quad (t \geq 1), \tag{4.1}$$

where z_t denotes closing price, and accordingly, y_t denotes log return. Looking at the graph of y_t, we can see that the volatility was very high in 2008, and it increased for a while in 2020. In 2008, there was a global financial crisis that started in the United States, and in early 2020, there was a global economic downturn caused by COVID-19, so we can assume that this graph shape represents these events. To ensure that the monitoring process works well around early 2020, where y_t had high volatility. To do this, we start training the NN

model from the time when the volatility of y_t stabilizes after 2012. The values up to 2018 are used for training the model. In monitoring, the time is considered as a change point, where the value of $\hat{T}_n(k)$ is greater than 2.689. When the monitoring is applied with this rule, the following result is obtained. In the graphs below, the vertical line means the change point estimated via $\hat{T}_n(k)$ with $\alpha = 0.05$. In addition, the estimated change points are "2020-03-10" in KOSPI and "2020-02-27" in S&P500, respectively.

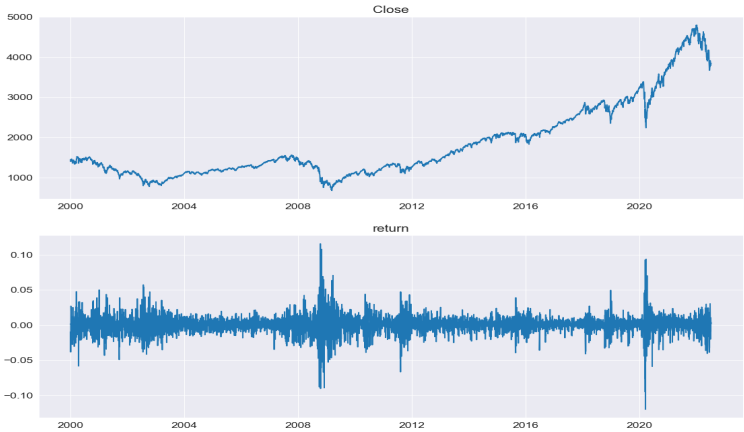

Figure 4.1 S&P500 index

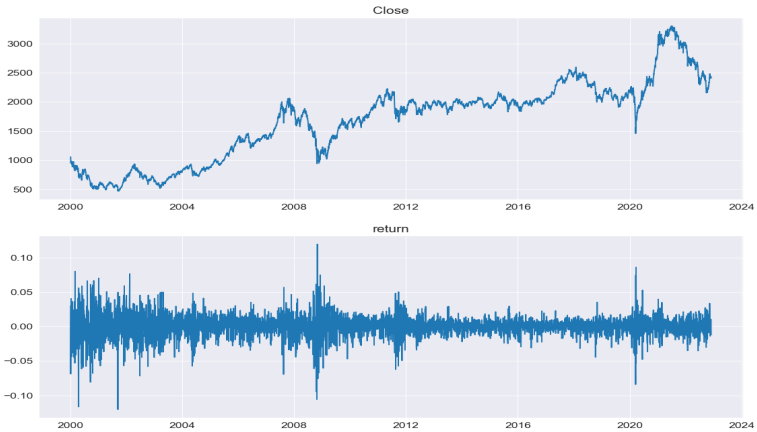

Figure 4.2 KOSPI index

Figure 4.3 Monitoring result in KOSPI

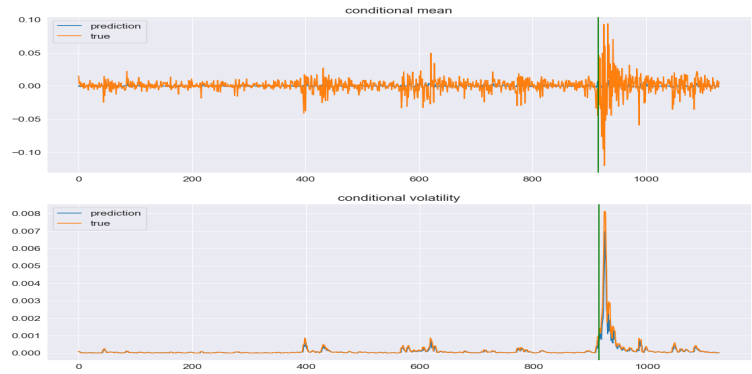

Figure 4.4 Monitoring result in S&P500

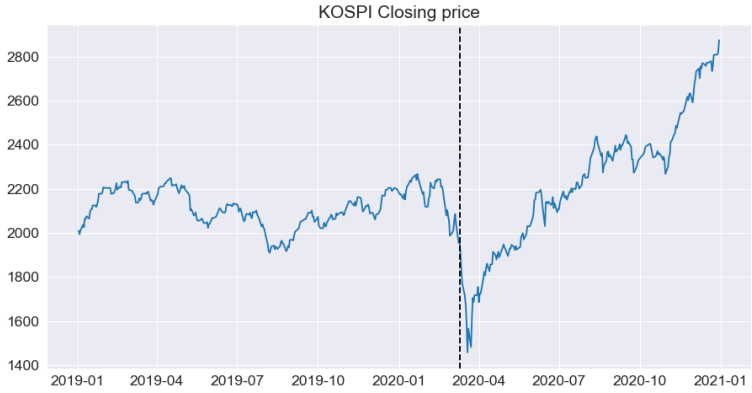

Figure 4.5 Monitoring result in KOSPI

Figure 4.6 Monitoring result in S&P500

Chapter 5

Conclusions

In the previous section, we tested the monitoring in GARCH-type time series using the NN model. The advantage of monitoring based on the NN model is that it can be used even if the data structure is not linear ARMA-GARCH and unknown. Our results overall proved the merits of NN method in monitoring. However, NN model has a complex structure that cannot be explained by existing statistical knowledge, so it does not present well the model explaining a given data. Also, the NN model has many hyperparameters to be adjusted, making it hard to find the optimal parameters, which need adjusting for each given data set. As only the NN method was considered here, conducting a comparison study with other machine learning methods will be our future project.

Bibliography

Lee, S.; Ha, J.; Na, O.; Na, S. (2003), "The Cusum Test for Parameter Change in Time Series Models", Scandinavian Journal of Statistics, 30, 781-796.

Lee, S.; Song, J. (2008), "Test for parameter change in ARMA models with GARCH innovations", Statistics Probability Letters, 78, 1990-1998.

Lee, S.; Oh, H. (2018), "On score vector- and residual-based CUSUM tests in ARMA-GARCH models", Statistical Methods and Applications., 27, 385-406.

Song, J.; Kang, J. (2018), "Parameter change tests for ARMA–GARCH models", Computational Statistics Data Analysis, 121, 41-56.

Leisch, F.; Hornik, K.; Kuan, CM. (2000), "Monitoring structural changes with the generalized fluctuation test", Econometric Theory, 16, 835-854.

Horváth, L.; Hušková, M.; Kokoszka, P.; Steinebach J. (2004), "Monitoring changes in linear models", Journal of Statistical Planning and Inference, 126, 225-251.

Zeileis, A.; Leisch, F.; Kleiber, C.; Hornik, K. (2005), "Monitoring structural change in dynamic econometric models", Journal of Applied Econometrics, 20, 99-121.

Berkes, I.; Gombay, E.; Horváth, L.; Kokoszka P. (2004), "Sequential change-point detection in GARCH (p, q) models", Econometric Theory, 20, 1140-1167.

Na, O.; Lee, Y.; Lee, S. (2011), "Monitoring parameter change in time series models", Statistical Methods Applications volume, 20, 171-199.

Gombay, E.; Serban, D. (2009), "Monitoring parameter change in AR (p) time series models", Journal of Multivariate Analysis, 100, 715-725.

Dienes, C.; Aue, A. (2014), "On-line monitoring of pollution concentrations with autoregressive moving average time series", Journal of Time Series Analysis, 35, 239-261.

Aue, A.; Dienes, C.; Fremdt, S.; Steinebach, J. (2015), "Reaction times of monitoring schemes for ARMA time series", Bernoulli, 21, 1238-1259.

Kim, S.; Cho, S.; Lee, S. (2000), "On the Cusum test for parameter changes in garch(1,1) Models", Communications in Statistics - Theory and Methods, 29, 445-462.

Lee, S.; Tokutsu, Y.; Maekawa, K. (2004), "The cusum test for parameter change in regression models with ARCH errors", Journal of the Japan Statistical Society, 34, 173-188.

Oh, H.; Lee, S. (2019), "Modified residual CUSUM test for location-scale time series models with heteroscedasticity", Annals of Institute of Statistical Mathematics, 71, 1059-1091.

Lee, S.; Kim, C. (2022), "Monitoring parameter change for time series models with application to location-Scale heteroscedastic models", Journal of Statistical Computation and Simulation, 92, 3885-3916

이상열 (2012), 시계열 분석: 이론 및 SAS 실습, 자유아카데미, 제 1판

조신섭, 손영숙, 성병찬 (2016), SAS/ETS를 이용한 시계열분석, 율곡출판사, 제 4판

국문초록

본 연구에서는 순환 신경망 모델을 사용하여 자기회귀(AR) 이동평균(MA) - 일반화 자기회귀이분산성(GARCH) 시계열에서 조건부 변동성이 변화하는 지점을 감지하고 모니터링하는 방법에 대해 논의한다. 주어진 정상 시계열 데이터를 LSTM이나 GRU와 같은 순환 신경망 모델을 이용하여 이전 시점의 관측값으로 미래 시점의 예측값과 변동성을 구한다. 그 과정에서 얻은 예측값으로 적절한 통계량을 만든 후, 시계열의 진행이 변하는 지점, 특히 조건부 변동성이 급격히 증가하는 점을 탐지하기 위해 CUSUM 테스트 기반 모니터링을 수행한다. 예측에 필요한 적절한 하이퍼파라미터는 그리드 서치를 이용하고, 시뮬레이션 데이터 및 실제 주가 데이터 S&P 500과 KOSPI 지수에서 모니터링 과정을 적용하고 결과를 분석한다.

주요어: 모니터링, GRU, 이분산성시계열모형, 신경망 모델, 변동성, 금융시장
학번: 2021-25859