Scientific Research Publishing

Journal of Data Analysis and Information Processing, 2017, 5, 115-130
http://www.scirp.org/journal/jdaip
ISSN Online: 2327-7203
ISSN Print: 2327-7211

Time Series Forecasting with Multiple Deep Learners: Selection from a Bayesian Network

Shusuke Kobayashi, Susumu Shirayama

Graduate School of Engineering, the University of Tokyo, Tokyo, Japan
Email: shuu.kbys@gmail.com, sirayama@sys.t.u-tokyo.ac.jp

How to cite this paper: Kobayashi, S. and Shirayama, S. (2017) Time Series Forecasting with Multiple Deep Learners: Selection from a Bayesian Network. *Journal of Data Analysis and Information Processing*, **5**, 115-130.
https://doi.org/10.4236/jdaip.2017.53009

Received: July 10, 2017
Accepted: August 26, 2017
Published: August 29, 2017

Abstract

Considering the recent developments in deep learning, it has become increasingly important to verify what methods are valid for the prediction of multivariate time-series data. In this study, we propose a novel method of time-series prediction employing multiple deep learners combined with a Bayesian network where training data is divided into clusters using K-means clustering. We decided how many clusters are the best for K-means with the Bayesian information criteria. Depending on each cluster, the multiple deep learners are trained. We used three types of deep learners: deep neural network (DNN), recurrent neural network (RNN), and long short-term memory (LSTM). A naive Bayes classifier is used to determine which deep learner is in charge of predicting a particular time-series. Our proposed method will be applied to a set of financial time-series data, the Nikkei Average Stock price, to assess the accuracy of the predictions made. Compared with the conventional method of employing a single deep learner to acquire all the data, it is demonstrated by our proposed method that F-value and accuracy are improved.

Keywords

Time-Series Data, Deep Learning, Bayesian Network, Recurrent Neural Network, Long Short-Term Memory, Ensemble Learning, K-Means

1. Introduction

Deep learning has been developed to compensate for the shortcomings of previous neural networks [1] and is well known for its high performance in the fields of character and image recognition [2]. In addition, deep learning's influence is impacting various other fields [3] [4] [5], and its efficiency and accuracy have been much bolstered by recent research. However, deep learning is subject to three main drawbacks. For instance, obtaining and generating appropriate

training data is problematic, it suffers from excessively long calculation times. Moreover, parameter selection is also difficult. While researchers are progressing toward overcoming such issues, according to some reports, dealing with these problems is difficult except for subjects in which the formation of feature spaces such as images and sounds are the key to success [6].

However, while it is clear that deep learning is considered to underpin artificial intelligence and because the brain's information processing mechanism is not fully understood, it is possible to develop new learners by imitating what is known about the information processing mechanisms of the brain. One way to develop new learners is to use a Bayesian network [7]. Moreover, it is also conceivable to combine multiple deep learners to create a single new learner. For example, ensemble learning and complementary learning are representative learning methods using multiple learners [8] [9] [10]. In addition, to realize a hierarchical control mechanism, there are cases where multiple learners are used.

In this research, we develop a new learner using multiple deep learners in combination with Bayesian networks as the selection method to choose the most suitable type of learner for each set of test data.

In time-series data prediction with deep learning, overly long calculation times are required for training. Moreover, a deep learner does not converge due to the randomness of the time-series data. There is also an issue with employing a Bayesian network. In this paper, we try to reduce the computation time and improve convergence by dividing training data into specific clusters using the K-means method and creating multiple deep learners from the learning derived from the divided training data. We also simplify the problem of ambiguity by using a Bayesian network to select a suitable deep learner for the task of prediction.

To demonstrate our model, we use a real-life application: predicting the Nikkei Average Stock price by taking into consideration the influence of multiple stock markets. Specifically, we estimate the Nikkei Stock Average of the current term based on the Nikkei Stock Average of the previous term as well as overseas major stock price indicators such as NY Dow and FTSE 100. We evaluate the validity of our proposed method based on the accuracy of the estimation results.

2. Related Works

In this section, we introduce the related works of multiple learners.

In ensemble learning, outputs from each learner are integrated by weighted averaging or a voting method [8]. In complementary learning, each learner is combined with the group to compensate for each other's disadvantages. Complementary learning is a concept arising from the role sharing in the memory mechanism of the hippocampus and cortex [9]. These learning methods tend to mainly use weak learners. Conversely, to realize a hierarchical control mechanism, there are cases where multiple learners are used. When the behavior of a robot or multi-agent entity is controlled, a hierarchical control mechanism is often adopted as attention is paid to the fact that such task can be divided into subtasks. Takahashi and Asada have proposed a robot behavior-acquisition me-

thod by hierarchically constructing multiple learners of the same structure [10]. A lower-level learner is responsible for different subtasks and learns low-level actions. A higher-level learner learns higher-level actions by exploiting a lower-level learner's knowledge.

Our proposed method, which will be described later, is based on the same notion as the bagging method used in ensemble learning where training data are divided and independently learned. The difference between our proposed method and the bagging method is the division method, the integration of multiple learners (the method of selecting suitable learners for each set of test data) to improve learners' accuracies in their acquisition of the material. Therefore, similar to Takahashi and Asada [10], we do not hold to the premise that tasks can be divided into subtasks. Our learner selection method is different. However, our use of deep learning entities as learners is different to Takahashi and Asada's approach as they simply used learners, which is a Q-learning algorithm extended to a continuous state behavior space. Furthermore, prior research of learning methods has not fully established a method of dividing training data, a method of integrating multiple learners, or a method of hierarchizing learners. In addition, it has also failed to improve each learner's performance after learning.

3. Proposed Method

As we mentioned, because the information processing mechanism of the brain is not fully understood, it is possible to develop new learners by imitating the information processing mechanism of the brain. In this research, we hypothesize that the brain forms multiple learners in the initial stage of learning and improves the performance of each learner in subsequent learning while selecting a suitable learner.

To design learners based on this hypothesis, it is necessary to find ways of constructing multiple learners, selecting a suitable learner, and improving the accuracy of each learner by using feedback from a particular selected learner. Hence, we assume that multiple learners have the same structure. The learners are constructed by the clustering of input data. Selection of a suitable learner is conducted with a naive Bayes classifier that forms the simplest Bayesian network. Furthermore, after fixing learners, we construct a Bayesian network and predict outcomes without changing the Bayesian network's construction. However, it is preferable to improve each learner's performance and the Bayesian network by using feedback gained from the selected learners. This will form one of our future research topics.

In the next section, we propose a method of constructing a single, unified learner by using multiple deep learners. Moreover, in Section 3.2, we propose a method of selecting a suitable learner with a naive Bayes classifier.

3.1. Learning with Multiple Deep Learners

In the analysis of time-series data with a deep learner, the prediction accuracy is

uneven because the loss function of certain time-series data does not converge. It is commonly assumed that the learning of weight parameters does not work due to the non-stationary nature of the data. This problem often occurs when multiple time-series data are used as training data. In addition, the long computational times that are required is also an issue.

To solve these problems, we think it is effective to apply clustering methods, such as K-means, SOM, and SVM, to training data; creating clusters; and constructing learners for each cluster. This is because training data divided into some clusters and multiple learners constructed for each cluster enables us to extract better patterns and improve convergence of the loss function compared to constructing a single classifier from all the training data. This method also enables the reduction of the computational time required. Moreover, classifiers for selecting a suitable learner are constructed from clustering the results of training data. This classifier achieves the task of associating test data to a suitable learner.

Figure 1 shows the framework of learning with multiple deep learners. We divided the training data into k classes (C_1, ···, C_k) and constructed k deep learners for each class. Figure 2 shows the framework of this prediction along with the test data. To determine which deep learner is in charge of prediction, we constructed a classifier for test data based on the clustering results of the training data. In this paper, we use K-means as a clustering method. Training data was divided into clusters with K-means and for each cluster, k learners were constructed. However, it is necessary to determine the number of clusters in ad-

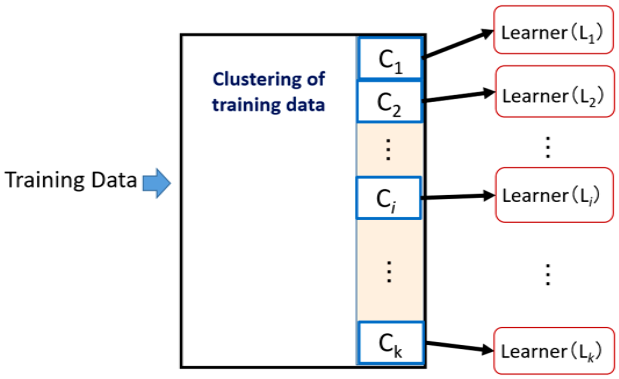

Figure 1. Multiple deep learner's structure.

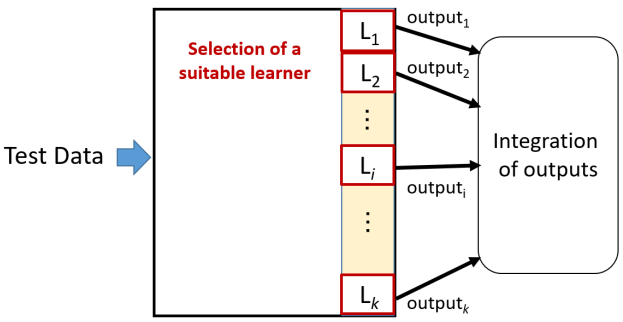

Figure 2. Prediction with multiple learners.

vance when we employ K-means. We decided the optimal number of clusters using an X-means algorithm, which calculates the optimal number of clusters best for K-means with the Bayesian information criterion. The X-means algorithm was presented in Pelleg and Moore's work [11].

Next, we use three types of deep learners, namely, deep neural network (DNN), recurrent neural network (RNN), and long short-term memory (LSTM). They are all well-established deep learning methods. To identify which deep learner was most suitable for each test data, we also used a naive Bayes classifier (the simplest type of Bayesian network). A naive Bayes classifier was constructed from the clustering results of the training data. **Figure 3** shows the framework of how the naive Bayes classifier was created. In the next section, we present a model of a naive Bayes classifier and the learning algorithm applied.

3.2. Selecting a Suitable Deep Learner

In this paper, we use a naive Bayes classifier to select a suitable deep learner for each set of test data. This method solves the classification problem using Bayes' theorem. The method hypothesizes conditional independence between feature values and is the simplest type of Bayesian network.

Let $\boldsymbol{x} = (x_1, x_2, \cdots, x_D)^{\mathrm{T}}$ be input data and $y \in Y$ the output class. A naive Bayes classifier has a graphical structure presented in **Figure 4**.

Furthermore, conditional probability $p(y|\boldsymbol{x})$ is defined as Expression (1) from the Bayes' theorem:

$$p(y|\boldsymbol{x}) = \frac{p(y)\,p(\boldsymbol{x}|y)}{p(\boldsymbol{x})} \tag{1}$$

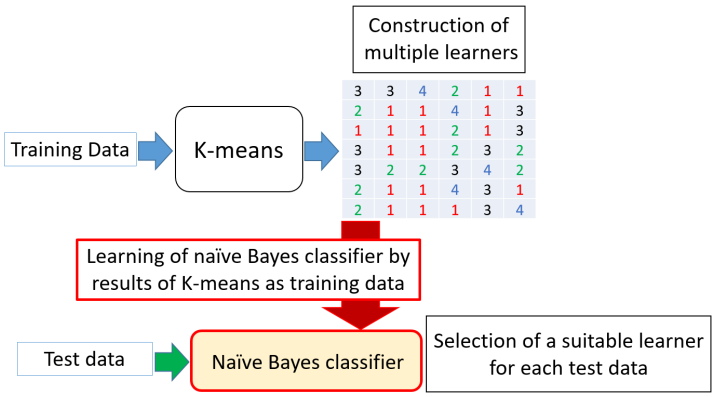

Figure 3. Learning of naive Bayes classifier.

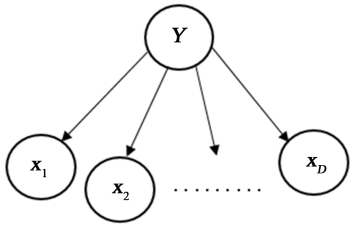

Figure 4. Structure of a naive Bayes classifier.

In the case of prediction, the predicted class $\hat{y}$ is defined as the class of which posterior probability $p(y|\boldsymbol{x})$ is the largest of all classes. Expression (2) presents the model of a naive Bayes classifier:

$$\begin{aligned} p(y|\boldsymbol{x}) &\propto p(y)\,p(\boldsymbol{x}|y) \\ &= p(y)\,p(x_1, x_2, \cdots, x_D|y) \\ &= p(y)\prod_{d=1}^{D} p(x_d|y) \end{aligned} \tag{2}$$

Let $X = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n, \cdots, \boldsymbol{x}_N\}$ be training data and $Y = \{y_1, \cdots, y_n, \cdots, y_N\}$ the correct class. $\boldsymbol{x}_n$ is D-dimensional vector. The correct class Y ($Y = \{1, 2, \cdots, k\}$) presents which k learner is associated to $\boldsymbol{x}_n$.

We hypothesize that each training dataset is generated independently. Let $\boldsymbol{\theta}$ be a parameter of probability distribution. The likelihood function $L(X, Y; \boldsymbol{\theta})$ is defined as Expression (3):

$$L(X, Y; \boldsymbol{\theta}) = \prod_{n=1}^{N} p(y_n)\,p(\boldsymbol{x}_n|y_n) = \prod_{n=1}^{N}\left[\, p(y_n)\prod_{d=1}^{D} p(x_{nd}|y_n)\right] \tag{3}$$

Moreover, x_{nd} is a vector component of $\boldsymbol{x}_n$. We can calculate the log likelihood function $\log L(X, Y; \boldsymbol{\theta})$ using the logarithm in Expression (3). The learning of a naive Bayes classifier results in the problem that we search for parameters that fit data the best. In other words, it decides the parameters that maximize the logarithmic likelihood.

Let us assume that $p(x_{nd}|y_n)$ follows a normal distribution and $p(y_n)$ follows a uniform distribution. Expression (4) presents a log likelihood function:

$$\log L(X, Y; \boldsymbol{\theta}) = \sum_{n=1}^{N}\left[\log p(y_n) + \sum_{d=1}^{D}\left[-\frac{1}{2}\log 2\pi - \log \sigma_{y_n d} - \frac{\left(x_{nd} - \mu_{y_n d}\right)^2}{2\sigma_{y_n d}^2}\right]\right] \tag{4}$$

The number of a cluster is defined as l ($l \in \{1, 2, \cdots, k\}$), and d represents the number of a dimension. Let $\delta(\cdot)$ be a delta function. We apply the maximum likelihood method to Expression (4), solved for parameters and get Expressions (5) and (6). Expressions (5) and (6) are the average and standard deviation of $p(x_{nd}|y_n)$:

$$\mu_{ld} = \frac{\sum_{n=1}^{N} x_{nd} \cdot \delta(y_n = l)}{\sum_{n=1}^{N} \delta(y_n = l)} \tag{5}$$

$$\sigma_{ld} = \frac{\sum_{n=1}^{N} \left(x_{nd} - \mu_{ld}\right)^2 \cdot \delta(y_n = l)}{\sum_{n=1}^{N} \delta(y_n = l)} \tag{6}$$

Therefore, in the learning of a naive Bayes classifier, Expressions (5) and (6) are derived from training data X and correct data Y. In selecting a suitable deep learner for test data, we use a naive Bayes classifier that is already trained. The predicted class y is the class of the largest probability for test data, and it is

determined by Expression (7). A naive Bayes classifier associates each test data to k deep learners.

$$\hat{y} = \arg\max_{y} \left[p(y|\boldsymbol{t}) \right] = \arg\max_{y} \left[p(y) \prod_{d=1}^{D} p(t_d|y) \right] \tag{7}$$

4. Experiment

As a case study, we predicted the future return of Nikkei Stock Average by applying six economic time-series datasets to our proposed method. From the previous day's data, we predicted whether the return of the next day's Nikkei Stock Average would be larger than the average return of Nikkei Stock Average or not.

4.1. How to Prepare the Data

We predicted the financial time-series using the method proposed in the previous section. The time-series data used in this case study were the closing prices of the daily data of the Nikkei Stock Average, New-York DOW, NASDAQ, S& P500, FTSE100 and DAX from January 1, 2000, to December 31, 2014. The New-York DOW, NASDAQ, and S & P500 are U.S. stock indicators. FTSE100 is a U.K. stock indicator and DAX is a German stock indicator. These data were sourced from Yahoo Finance [12] and the Federal Reserve Bank of St. Lois [13].

However, some dates do not show all 6 stock prices because the dates of holidays in each country are different. In such cases, we assumed that markets that had no data due to holidays remained unchanged and adopted the previous day's stock prices. We defined data from 2000 to 2013 as training data and data from 2014 as test data. Because time-series data typically has strong non-stationary tendencies, it is difficult to deal with them in their raw format. Thus, we transformed stock price data to returns.

Let time-series data be $\boldsymbol{p}(t)(0 \le t \le T)$. According to Expression (8), we transformed the stock price vector $\boldsymbol{p}(t)$ and $\boldsymbol{p}(t-1)$ to return $\boldsymbol{r}(t)$:

$$\boldsymbol{r}(t) = \{\log \boldsymbol{p}(t) - \log \boldsymbol{p}(t-1)\} \times 100 \tag{8}$$

Return $\boldsymbol{r}(t)(1 \le t \le T)$ was derived by shifting the date one by one day.

We conducted the Dickey-Fuller test to check stationarity of return $\boldsymbol{r}(t)$.The null hypothesis for this test is that there is a unit root. The alternative hypothesis is that the time-series is stationary. As a result of this test, the null hypothesis was rejected at a significance level of 5%. We assumed stationarity of return $\boldsymbol{r}(t)$ and predicted deviation from the average return of Nikkei. We defined the average as simple average because it is constant over time from the definition of stationarity.

4.2. Result

We now present the experimental results of the prediction of financial time-series data. From today's data $\boldsymbol{r}(t)$, we predicted whether the next day's return $\boldsymbol{r}(t+1)$ of Nikkei Stock Average would be larger than the average return of

Nikkei or not. The deep learners used in this experiment were DNN, RNN, and LSTM. The input layer of DNN had 6 units. The two hidden layers had 6 units. The output layer had 2 units. Weights of DNN were learned over 300 iterations. Conversely, the input layer of RNN and LSTM had 6 units. The two hidden layers had 6 units. The output layer had 2 units. The weights of RNN and LSTM were learned over 100 iterations. First, we show results of the conventional prediction method. In this experiment, we used only one deep learner. All training data were used to train this single deep learner. Five experiments were conducted for each learner.

As Table 1, Table 3 and Table 5 show, the F-values of DNN, RNN, and LSTM were 61.40%, 71.55%, and 69.73%, respectively. The accuracy of DNN, RNN, and LSTM were 65.77%, 71.69%, and 53.69%, respectively. The computational time of DNN, RNN, and LSTM were 3.235×10^2 [s], 3.796×10^2 [s], and 1.531×10^2 [s], respectively.

In Figure 5, we show how the loss function of training data and test data changed. Let $\boldsymbol{t} = (t_1, t_2, \cdots, t_S)^{\mathrm{T}}$ be label data and $\boldsymbol{y} = (y_1, y_2, \cdots, y_S)^{\mathrm{T}}$ outputs of a neural network where S denotes the number of the output neurons. N represents the amount of data. The softmax error E is defined as Expression (9).

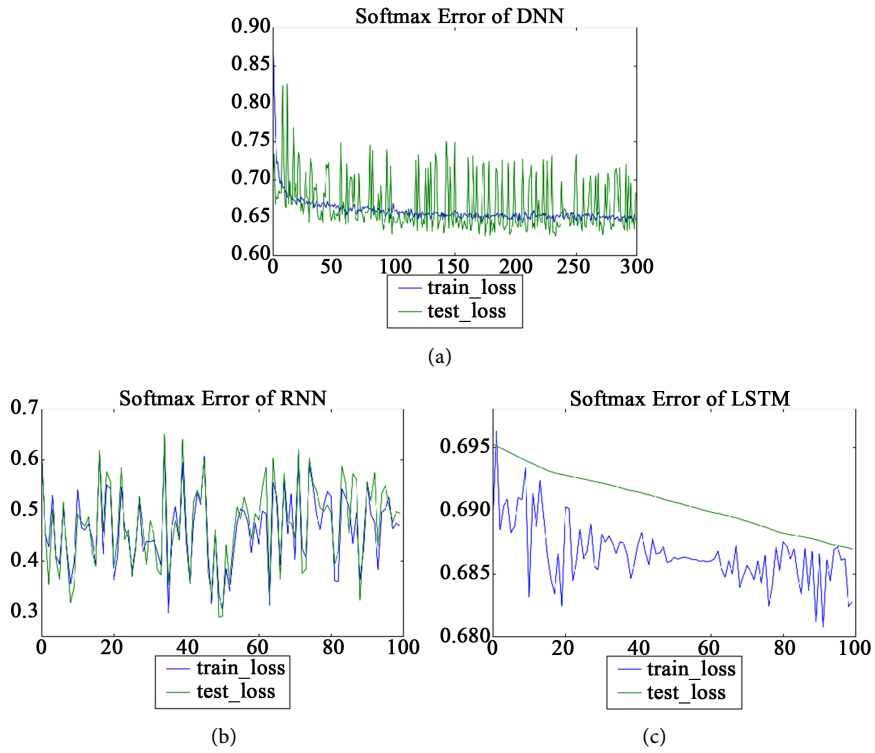

Figure 5. Cross entropy. (a) DNN; (b) RNN; (c) LSTM.

Table 1. F-value and accuracy of test data (DNN).

	F-value	accuracy
Conventional method	0.6140 ± 0.00293	0.6577 ± 0.0148
Proposed method	0.5854 ± 0.01131	0.6877 ± 0.00411

$$E = -\sum_{n=1}^{N}\sum_{i=1}^{S} t_i^{(n)} \log y_i^{(n)} \tag{9}$$

where the superscript (n) denotes the n-th number of the training data.

Figure 5(a) presents results when all data were applied to only one DNN. Figure 5(b) presents the results in the case of applying all data to only one RNN. In the same way, Figure 5(c) presents how loss functions change when we apply all data to a single LSTM.

Next, we present the results of our proposed method. In this experiment, we constructed multiple deep learners in accordance with our method. The construction of the multiple learners and the production of the predictions are as follows.

After we applied X-means to training data and determined the optimum division of number K, we constructed k clusters with K-means and k deep learners. We applied test data to a naive Bayes classifier learned by clustering the results of training data. With this naive Bayes classifier, we associated each test dataset to a suitable deep learner and predicted whether the following day's retun of Nikkei Stock Average would be above the average or not. Five experiments were also conducted in order to measure F-value, accuracy, and computational time.

F-value of multiple DNN, RNN, and LSTM were 58.54%, 72.40%, and 81.42%, respectively. The accuracy of multiple DNN, RNN, and LSTM were 68.77%, 72.62%, and 69.08%, respectively. In addition, the computational time of multiple DNN, RNN, and LSTM were 2.064×10^2 [s], 2.077×10^3 [s], and 1.533×10^3 [s], respectively.

The results of each experiment are summarized in Tables 1-6. The top row of each Table presents the results in the case of applying the conventional method, whereas the lower row presents the results when our method was used.

Table 2. Computational time of test data (DNN).

	Computational time [s]
Conventional method	$3.235 \times 10^2 \pm 2.578$
Proposed method	$2.064 \times 10^2 \pm 6.642$

Table 3. F-value and accuracy of test data (RNN).

	F-value	accuracy
Conventional method	0.7155 ± 0.02712	0.7169 ± 0.02657
Proposed method	0.7240 ± 0.01284	0.7262 ± 0.01389

Table 4. Computational time of test data (RNN).

	Computational time [s]
Conventional method	$3.796 \times 10^3 \pm 1.259 \times 10^1$
Propose method	$2.077 \times 10^3 \pm 3.591 \times 10^1$

Table 5. F-value and accuracy of test data (LSTM).

	F-value	accuracy
Conventional method	0.6973 ± 0.02192	0.5369 ± 0.02462
Proposed method	0.8142 ± 0.000615	0.6908 ± 0.002107

Table 6. Computational time of test data (LSTM).

	Computational time [s]
Conventional method	$1.532 \times 10^3 \pm 6.041$
Propose method	$1.533 \times 10^3 \pm 6.797$

Moreover, we show the change in error functions when our method was applied. The optimum division number derived from X-means varies depending on how the initial clusters in the algorithm of X-means were decided although the behavior of the error functions showed similarity.

As an example, we present graphs illustrating how loss functions changes. With the X-means algorithm, the optimum division number N was determined and training data was divided into N classes from C_1 to C_N. The number of each cluster for three deep learners is as follows. Table 7 represents how much data is comprised in each cluster in the case of using multiple DNN. Table 8 shows the data in each cluster in the case of using multiple RNN. Table 9 shows how much data each cluster has in the case of using multiple LSTM.

Figure 6 illustrates when DNN was used as deep learners. Figure 7 shows the change of the loss function when RNN was used as deep learners. Similarly, Figure 8 shows graphs of when LSTM was used as a deep learner.

5. Discussion

In our research, we hypothesized that the brain forms multiple learners at the initial stage of learning and improves the performance of each learner while selecting the most suitable learner in subsequent learning tasks. In this paper, we proposed a method of constructing multiple learners and a method of selecting a suitable learner for each dataset.

Our proposed method is as follows:

1) The optimum division number of clustering is determined using X-means.

2) Training data is divided using K-means and multiple learners for each cluster constructed with DNN, RNN, and LSTM.

3) A naive Bayes classifier is constructed by the clustering result of training data.

4) A suitable deep learner for each test dataset is selected with the constructed naive Bayes classifier.

5) Prediction is conducted by the selected learner.

Predictive experiments on financial time-series data of six stock indicators were performed using the proposed method. Our experiments suggest that when multiple learners are used, most loss functions decrease compared with the case

Table 7. Amount of data in each cluster (DNN).

Cluster	Training	Test
C_1	37	2
C_2	2069	184
C_3	74	2
C_4	71	2
C_5	131	7
C_6	11	0
C_7	234	3
C_8	427	20
C_9	598	40

Table 8. Amount of data in each cluster (RNN).

Cluster	Training	Test
C_1	28	0
C_2	2129	188
C_3	14	0
C_4	138	8
C_5	44	0
C_6	8	0
C_7	96	2
C_8	403	18
C_9	191	4
C_{10}	568	38
C_{11}	4	2
C_{12}	5	0
C_{13}	6	0
C_{14}	3	0
C_{15}	15	0

Table 9. Amount of data in each cluster (LSTM).

Cluster	Training	Test
C_1	2129	187
C_2	37	2
C_3	84	2
C_4	74	2
C_5	133	7
C_6	568	39
C_7	224	3
C_8	403	18

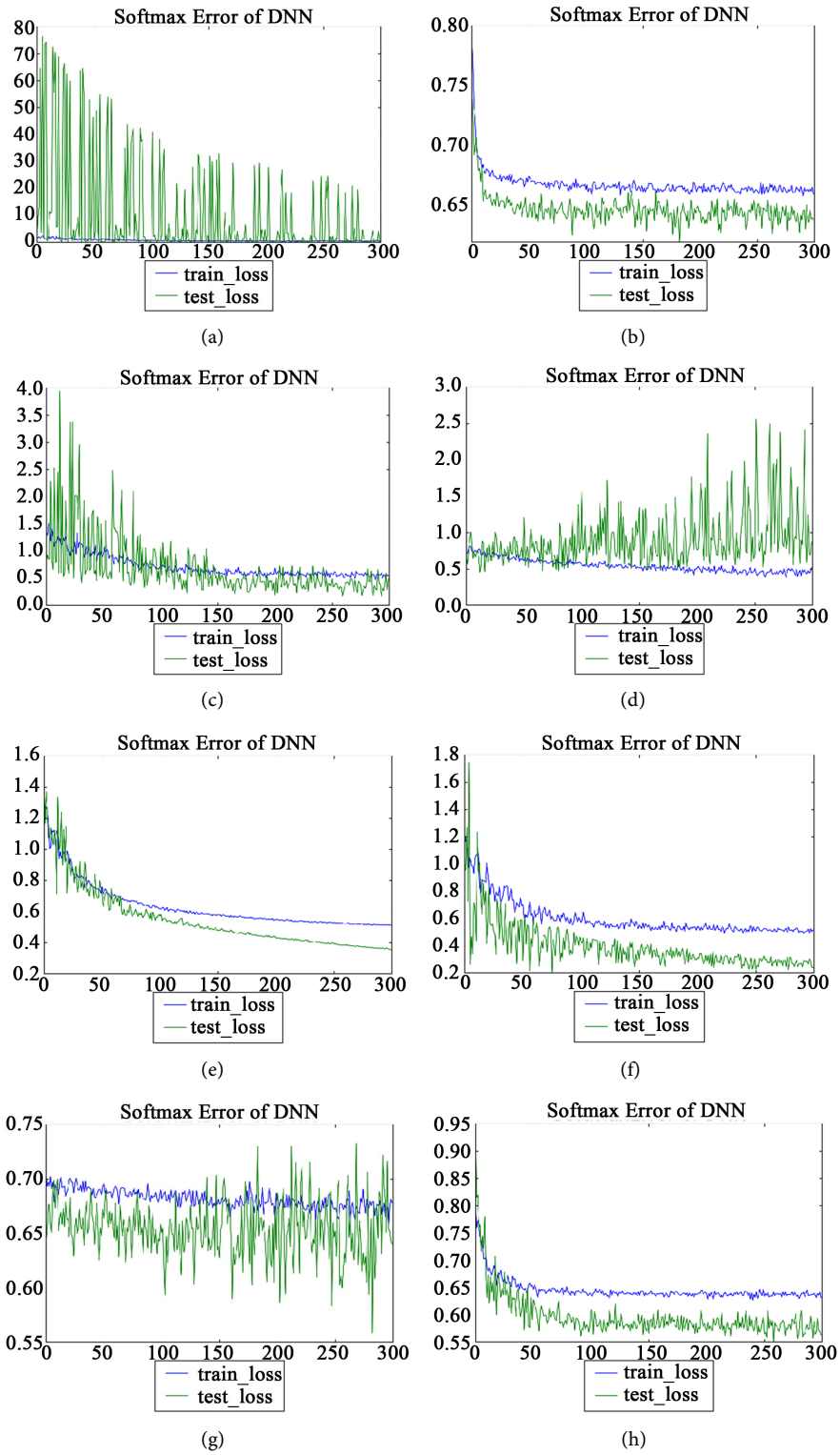

Figure 6. Cross entropy. (a) C_1; (b) C_2; (c) C_3; (d) C_4; (e) C_5; (f) C_7; (g) C_8; (h) C_9.

where all data are learned by a single learner. In the case of using multiple LSTM, F-values are improved greatly compared to using multiple DNN and RNN. Conversely, the accuracy in the case of using multiple LSTM was a little higher than that of multiple DNN. However, it was a little lower than the accu-

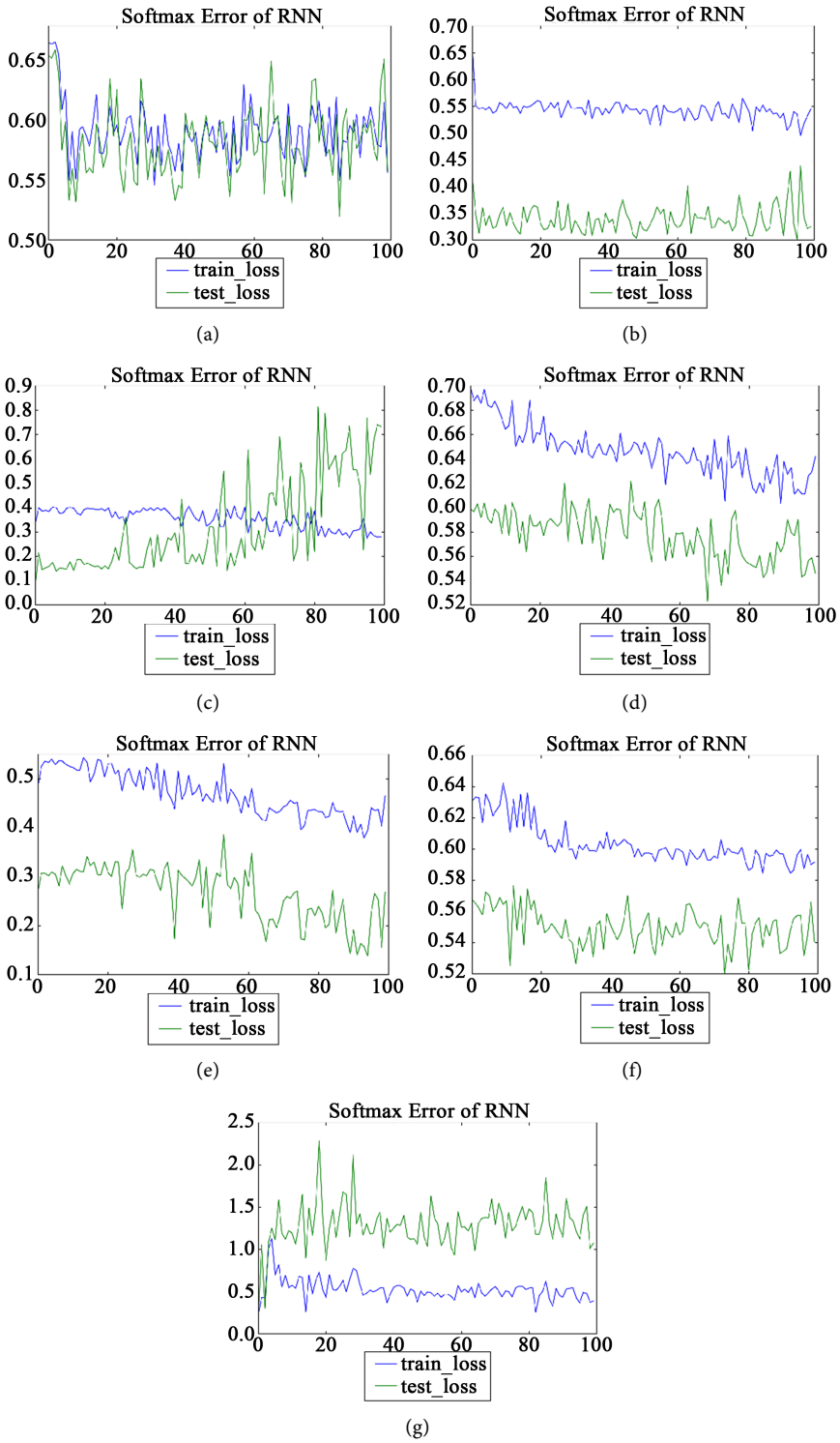

Figure 7. Cross entropy. (a) C_2; (b) C_4; (c) C_7; (d) C_8; (e) C_9; (f) C_{10}; (g) C_{11}.

racy in the case of use of multiple RNN. Furthermore, when LSTM was used as multiple learners, the computational time became shorter than when an RNN was used.

These results indicate that our proposed method enables us to deal with the non-stationary nature of time-series data and extract more accurate patterns.

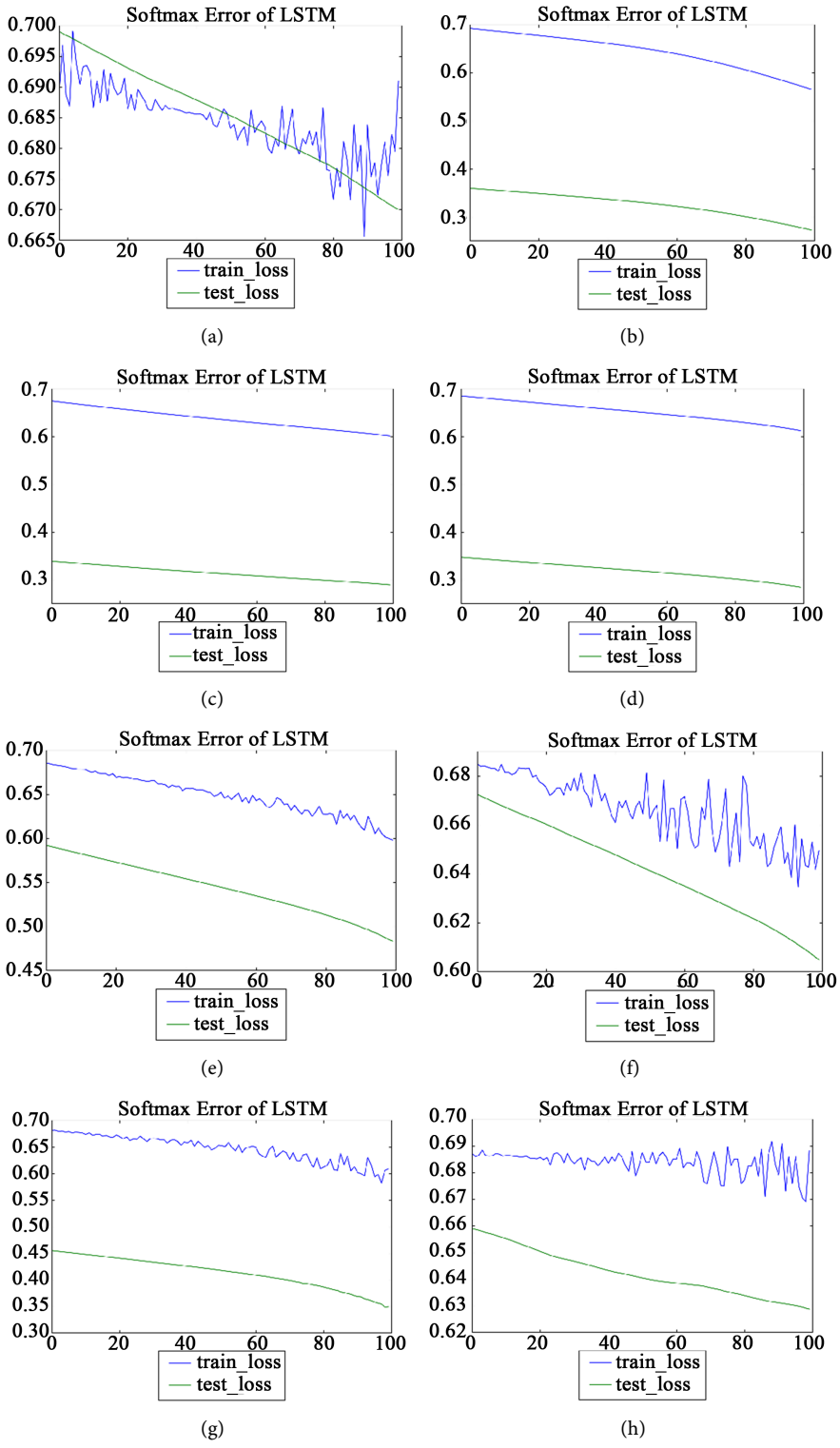

Figure 8. Cross entropy. (a) C_1; (b) C_2; (c) C_3; (d) C_4; (e) C_5; (f) C_6; (g) C_7; (h) C_8.

We suppose that LSTM is especially effective in the prediction of time-series data that have remarkable features. In our proposed method, the division of time-series data by K-means clustering corresponds to extracting the remarkable features of such data. We believe that it is possible to improve the proposed me-

thod further by determining more suitable parameters for deep learners according to each cluster.

6. Conclusions

We propose a new method of constructing multiple deep learners and determining which deep learner is in charge of the test data with a naive Bayes classifier. Experiments suggested that when multiple learners were used, the loss functions showed a decreasing trend as compared with the case where all the data were learned by a single learner. As a result, F-values and the accuracy of our method are better than those of the conventional method. Moreover, our proposed method also shortens the computational time required.

Concerning this research topic, the future issues under consideration are as follows:

First, the validity of the method of assigning test data will be considered. In this paper, we used a naive Bayes classifier to assign test data to a suitable learner. However, in terms of the prediction method, it is also possible to use the K-means method or SVM instead of the naive Bayes classifier. It is necessary to compare the experimental results of our method with research using K-means or SVM.

Second, improving each learner and the Bayesian network itself by using feedback from a selected learner is considered. In this paper, after fixing multiple learners, we constructed a Bayesian network and performed predictive experiments without changing the construction of the Bayesian network. However, considering the information processing mechanism of the human brain, it is preferable to give feedback on prediction result to learners and the Bayesian network.

Third, case studies will be conducted using the proposed method with different data. In this paper, we applied financial time-series data to our method. It is considered that depending on the data, the deep learner's method of producing optimum prediction results and the method of assigning test data to multiple learners change. We experimented after deciding the learner's method and the method of assigning test data in advance. However, a future development would be to construct a framework that could mechanically determine which model would give the best predictions based on the data provided.

References

[1] Schmidhuber, J. (2015) Deep Learning in Neural Networks: An Overview. *Neural Networks*, **61**, 85-117.

[2] Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S. and Lew, M.S. (2016) Deep Learning for Visual Understanding: A Review. *Neurocomputing*, **187**, 27-48.

[3] Chao, J., Shen, F. and Zhao, J. (2011) Forecasting Exchange Rate with Deep Belief Networks. *Proceedings of* 2011 *International Joint Conference on Neural Networks*, San Jose, CA, 31 July-5 August 2011, 1259-1266.

[4] Ribeiro, B. and Lopes, N. (2011) Deep Belief Networks for Financial Prediction. *Lecture Notes in Computer Science*, **7064**, 766-773. https://doi.org/10.1109/IJCNN.2011.6033368

[5] Bengio, Y., Courville, A. and Vincent, P. (2014) Deep Learning: Methods and Applications. *Foundations and Trends in Signal Processing*, **7**, 197-387. https://doi.org/10.1561/2000000039

[6] Samek, W., Binder, A., Montavon, G., Lapuschkin, S. and Muller, K.-R. (2016) Evaluating the Visualization of What a Deep Neural Network Has Learned. *IEEE Transactions on Neural Networks and Learning Systems*, **PP**, 1-14. https://doi.org/10.1109/TNNLS.2016.2599820

[7] Zuo, Y. and Kita, E. (2012) Stock Price Forecast Using Bayesian Network. *Expert Systems with Applications*, **372**, 6729-6737.

[8] Krawczyk, B., Minku, L.L., Gama, J., Stefanowski, J. and Woozniak, M. (2017) Ensemble Learning for Data Stream Analysis: A Survey. *Information Fusion*, **37**, 132-156.

[9] Schapiro, A.C., Turk-Browne, N.B., Botvinick, M.M. and Norman, K.A. (2017) Complementary Learning Systems within the Hippocampus: A Neural Network Modelling Approach to Reconciling Episodic Memory with Statistical Learning. *Philosophical Transaction of the Royal Society B*, **372**, 20160049. https://doi.org/10.1098/rstb.2016.0049

[10] Takahashi, Y., Takeda, M. and Asada, M. (1999) Continuous Valued Q-Learning for Vision-Guided Behavior Acquisition. *Proceedings of* 1999 *IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Taiwan, August 15-18 1999, 255-260.

[11] Pelleg, D. and Moore, A. (2000) X-Means: Extending K-Means with Efficient Estimation of the Number of Clusters. *Proceedings of 17th International Conference on Machine Learning*, Stanford, CA, 29 June-2 July 2000, 727-734.

[12] Yahoo! Finance. http://finance.yahoo.com/

[13] Economic Research Federal Reserve Bank of St. Louis. http://research.stlouisfed.org/