Studies on Time Series Forecasting by using Hybrid Deep Learning Architectures
(ハイブリッドディープラーニングによる時系列予測に関する研究)

Dissertation submitted in partial fulfillment for the
degree of Ph.D. of Engineering

YEPENG CHENG

Under the supervision of
Professor Yasuhiko Morimoto

Social Computing Laboratory,
Department of Information Engineering,
Graduate School of Engineering,
Hiroshima University, Higashi-Hiroshima, Japan

February 2021

Abstract

This dissertation discusses logistic regression analysis-based retailer competition analysis, and hybrid deep learning architectures-based multi-conditional time series forecasting, respectively. We will introduce them successively.

Customer relationship analysis is vital for retail stores, especially for supermarkets. POS systems make it possible to record the daily purchasing behaviors of customers as an ID-POS database, which can be used to analyze customer behaviors of a supermarket. The customer value is an indicator based on ID-POS database for detecting the customer loyalty of a store. In general, there are many supermarkets in a city, and other nearby competitor supermarkets significantly affect the customer value of customers of a supermarket. However, it is impossible to get detailed ID-POS databases of competitor supermarkets. This study firstly focused on the customer value and distance between a customer's home and supermarkets in a city, and then constructed the models based on logistic regression analysis to analyze correlations between distance and purchasing behaviors only from a POS database of a supermarket chain. During the modeling process, there are three primary problems existed. Intuitively, the incomparable problem of customer values, the multicollinearity problem among customer value and distance data, and the number of valid partial regression coefficients. The improved customer value, Huff's gravity model, and inverse attractiveness frequency are considered to solve these problems. This thesis presents three types of models based on these three methods for loyal customer classification and competitors' influence analysis. In numerical experiments, all types of models are useful for loyal customer classification. The type of model, including all three methods, is the most superior one for evaluating the influence of the other nearby supermarkets on customers' purchasing of a supermarket chain from the viewpoint of valid partial regression coefficients and accuracy.

Traditional time series forecasting techniques can not extract good enough sequence data features, and their accuracies are limited. The deep learning structure SeriesNet is an advanced method, which adopts hybrid neural networks, including dilated causal convolutional neural network (DC-CNN) and Long-short term memory recurrent neural network (LSTM-RNN), to learn multi-range and multi-level features from multi-conditional time series with higher accu-

racy. However, they didn't consider the attention mechanisms to learn temporal features. Besides, the conditioning method for CNN and RNN is not specific, and the number of parameters in each layer is tremendous. This thesis proposes the conditioning method for two types of neural networks, and respectively uses the gated recurrent unit network (GRU) and the dilated depthwise separable temporal convolutional networks (DDSTCNs) instead of LSTM and DC-CNN for reducing the parameters. Furthermore, this thesis presents the lightweight RNN-based hidden state attention module (HSAM) combined with the proposed CNN-based convolutional block attention module (CBAM) for time series forecasting. Experimental results show our proposed model, attention-based SeriesNet (A-SeriesNet), is superior to other models from the viewpoint of forecasting accuracy and computation efficiency.

The A-SeriesNet combined augmented attention residual learning module-based convolutional neural network (augmented ARLM-CNN) subnetwork with hidden state attention module-based recurrent neural network (HSAM-RNN) subnetwork for conditional time series prediction with high accuracy. The augmented ARLM-CNN subnetwork has defects in extracting latent features of the multi-condition series. The forecasting accuracy will decrease when the feature dimension of the multi-condition series becomes high. The same problem also occurs in the HSAM-RNN subnetwork of A-SeriesNet. The dual-stage attention recurrent neural network (DA-RNN) proved that the attention-based encoder-decoder framework is an effective model for dealing with the above problem. This thesis applies the DA-RNN to the HSAM-RNN subnetwork of A-SeriesNet and presents the triple-stage attention-based recurrent neural network (TA-RNN) subnetworks. Furthermore, this thesis considers a CNN-based encoder-decoder structure named dual attention residual learning module-based convolutional neural network (DARLM-CNN) subnetwork to improve the augmented ARLM-CNN subnetwork of A-SeriesNet. Finally, this thesis presents the triple-stage attention-based SeriesNet (TA-SeriesNet), which uses a new concatenation method instead of the element-wise multiplication of A-SeriesNet to parallel connect the proposed subnetworks and reduce the dependence of forecasting results on a certain subnetwork. The experimental results show our TA-SeriesNet is superior to other state-of-art deep learning models from the

viewpoint of forecasting accuracy evaluation metrics.

Acknowledgements

First and foremost, I would like to extend my sincere gratitude to Professor Yasuhiko Morimoto, the supervisor of my study, for his valuable guidance, kind advice in every stage of the writing of this thesis. Without his enlightening instruction and impressive patience, I could not have completed my thesis. Also, my thanks go to Professor Hiroyuku Okamura, for his invaluable comments, useful suggestions and warm encouragement. Finally, it is my special pleasure to acknowledge the hospitality and encouragement of the past and present members of the social computing Laboratory, Department of Information Engineering, Hiroshima University. I also feel gratitude for my family's support and my girlfriend's accompany and encouragement for my Ph.D. graduation.

Contents

Abstract **iii**

Acknowledgements **vii**

1 Introduction **1**

2 Enterprise Competition Analysis **5**

2.1 Related work 5

2.1.1 Retailer Competition Analysis 5

2.1.2 Customer Analysis 6

2.1.3 Regression Analysis 7

2.2 Loyal Customer Analysis Model 8

2.2.1 RFM Analysis 8

2.2.2 Huff's Gravity Model 9

2.2.3 Inverse Attractiveness Frequency 10

2.2.4 Decyl Analysis 13

2.3 Analytical methods 15

2.3.1 Logistic Regression Analysis 15

2.3.2 Evaluation Criteria 16

2.4 Experiment 16

2.4.1 Experimental Data 16

2.4.2 Experimental Procedure 17

2.4.3 Experiment of Supermarket Chain 18

2.4.4 Experiment of Individual Supermarkets 23

3 Multi-conditional time series forecasting **27**

3.1 Related Work 27

3.2 Definition of multi-conditional time series 29
3.3 Attention-Based SeriesNet 30
3.3.1 Conditioning 30
3.3.2 Dilated Depthwise Separable Temporal Convolutional Networks 32
3.3.3 Convolutional Block Attention Module 33
3.3.4 Hidden State Attention Module 35
3.4 Triple-stage attention-based SeriesNet 38
3.4.1 Structure of TA-SeriesNet 38
3.4.2 Structure of DARLM-CNN subnetwork 40
3.4.3 Structure of TA-RNN subnetwork 44
3.4.4 Concatenation of subnetworks 50
3.5 Experiments of A-SeriesNet 51
3.6 Experiments of TA-SeriesNet 59
3.6.1 Training procedure and evaluation metrics 59
3.6.2 Model parameter adjustment 59
3.6.3 Experimental results 64

4 Conclusion **71**

Bibliography **73**

Publication List of the Author **79**

Chapter 1

Introduction

In today's supermarket business, the ID-POS database enables supermarkets to analyze customer behavior and adopt more targeted and personalized marketing strategies such as customer relationship management (CRM) [1], to improve the competitiveness of supermarkets. The ID-POS database digitally records customer ID, customer information, sales records., etc. Therefore, customer behavior is measurable by counting their daily shopping records as customer values. Generally speaking, customer value analysis, which is also known as RFM analysis [1]- [3], mainly depends on three parametric indicators, customer shopping recency, frequency, and monetary. They can reflect the customer loyalty of a store. The models consist of RFM indicators with other statistical parameters that are trainable by clustering analysis [4] and other machine learning methods to investigate the customer shopping preference. Tanaka et al. [5], proposed a model, including RFM indicators with the proportion of purchased products of each customer in a supermarket chain. They define the loyal customer by Decyl analysis [5], and then use logistic regression analysis [6]- [9] to find loyal customers and detect the loyal customers' preferences for each product simultaneously. Logistic regression analysis is widely used in parametric impact analysis. The coefficients of logistic regression mathematically considered as the parameters in the Odds ratio [10]. The Odds ratio can reflect the influence of variable parameters on a particular parameter. As a result, they built a loyal customer analysis model with high classification accuracy. There is a lot of the other customer's information in the ID-POS database, such as the customer's address. Therefore, Tanaka's method is also useful to detect different aspects

of the customer's behavior. For example, the distance between a customer's home and all supermarkets in a city is computable. The influence of nearby competitors is discoverable by analyzing the relationship between distance and the customer's shopping amount of the target supermarket. The customers who live close to competitors are more likely to be influenced by them, resulting in decreased shopping amounts in the target supermarket. However, logistic regression cannot train the raw distance data without preprocessing directly since the multicollinearity problem [11] may occur between the distance data and RFM indicators. Therefore, it is essential to find a method that can transform the distance data into probability similar to Tanaka's work. This thesis first considers three types of models for loyal customer classification and retailer competition anaysis.

In big data analysis, time series forecasting is an essential branch developed in recent years. Traditional methods have some limitations for time series forecasting since the time series possess characteristics such as non-linearity, non-stationarity and unknown dependencies. Deep learning is an advanced approach to overcome these problems. It depends on non-linear modules to learn the fully features from the input data. Shen et al. [27] proposed a deep learning structure named SeriesNet, which combined the dilated causal convolutional neural networks (DC-CNN) [37] and the long-short term memory (LSTM) [30]. They evaluated that their model has higher forecasting accuracy and greater stableness. LSTM and DC-CNN are widely applied to time series forecasting with excellent performance. However, DC-CNN and LSTM include a large number of parameters, resulting in tremendous computation cost. Gated recurrent unit network (GRU) [33] and LSTM have a comparable performance on time series forecasting, but parameter quantity significantly reduced. So does the dilated depthwise separable temporal convolutional networks (DDSTCNs) [36] compared with DC-CNN. The SeriesNet can directly input raw time series sequences by conditioning the target time series on the additional time series. But the specific conditioning method is not clarified in their work. In addition, they did not consider the attention mechanisms in SeriesNet. Recently, most researches focus on the recurrent neural network (RNN) based attention to improve the deep learning structure. However, the heavyweight attention mechanism within massive train-

ing parameters will influence the computation efficiency. The convolutional block attention module (CBAM) [35] is a lightweight attention structure, but has only been successfully applied to image recognition so far. This thesis presents the attention-based SeriesNet (A-SeriesNet) to solve the above problems. Fig. 3.2 demonstrates the overall structure of A-SeriesNet. The A-SeriesNet has two subnetworks. The CNN-based [29] subnetwork used augmented attention residual learning module (augmented ARLM) [32] for conditioning the multi-condition series (Condition) on target time series (Input). The conditioning method of the RNN-based [31] subnetwork fed the multi-condition series into the first gated recurrent unit (GRU) [33] layer's initial hidden state via a flatten operation and a full-connection layer (FC). We adopted batch normalization (BN) [34] and convolutional block attention module-based (CBAM) [35] dilated depthwise separable temporal convolutional networks (DDSTCNs) [36, 45] instead of dilated causal convolutional neural network (DC-CNN) [37] in the residual learning module of SeriesNet [27] for parameter reduction and precision improvement. The CBAM [35] is a lightweight attention mechanism for image recognition and time series prediction, which focuses on global max pooling and global average pooling of a CNN [29] layer. The DDSTCNs [45] simplifies training parameters of DC-CNN [37] via depthwise convolution and pointwise convolution. Besides, we propose a variant of CBAM [35] named hidden state attention module (HSAM), which focuses on global max pooling and global average pooling of hidden states between two RNN [31] layers. Although the A-SeriesNet has excellent forecasting performance, the element-wise multiplication of two subnetworks structure restricts the forecasting accuracy. When there is a large forecasting deviation among two subnetworks, the overall accuracy is sensitive to either of them and not appropriate to the parallel connection of more than two subnetworks.

On the other hand, The A-SeriesNet didn't learn the multi-condition series' potential features before conditioning them on target time series in both subnetworks of A-SeriesNet. The prediction accuracy will decline when the feature dimension (series number) of multi-condition series increased. Yao et al. [38] proposed the dual-stage attention-based recurrent neural network (DA-RNN) [38] to deal with the above problem. The DA-RNN [38] encodes multi-condition se-

ries as feature context vectors via an RNN [31] encoder with input attention and decodes the context vectors conditioned on the target series via an RNN [31] decoder with temporal attention. The DA-RNN [38] proved the encoder-decoder framework is an excellent solution to the above problem. However, the encoder-decoder structure has a performance deterioration problem for a long time sequence [39]. The HSAM concentrates on hidden states between two RNN [31] layers, which is also appropriate for further improving the DA-RNN [38]. According to the disadvantages of A-SeriesNet, We improve A-SeriesNet and present the triple-stage attention-based SeriesNet (TA-SeriesNet) in this dissertation.

The organization of this dissertation is as follows. The logistic regression analysis-based retailer competition analysis is shown in Section 2. Section 3 introduces the details of hybrid deep learning architectures-based multi-conditional time series forecasting including A-SeriesNet, TA-SeriesNet and the experiments of them. Section 4 gives the conclusions and future work.

Chapter 2

Enterprise Competition Analysis

2.1 Related work

2.1.1 Retailer Competition Analysis

In economic, Reilly [14] applied the law of gravity in physics to analyze the retail industry, which indicated that consumers are willing to drive a further distance to larger retail stores for shopping. However, this law only considers the macro aspect and lacks the investigation of the micro aspect of consumer decision-making in actual shopping activities. Reilly's law assumes that consumers will choose a fixed retail store for shopping. In fact, consumers expect to go shopping at two retail stores in close geographical locations simultaneously. Huff's gravity model [13] made a breakthrough against this theoretical flaw later.

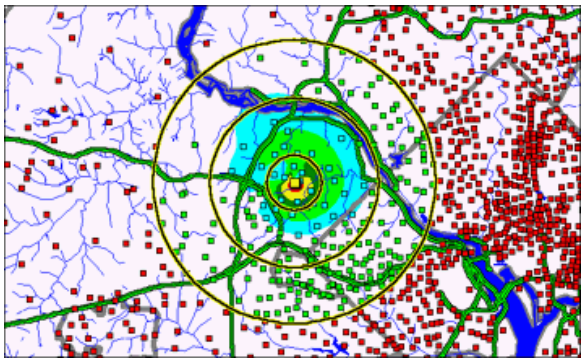


Figure 2.1: The customer location and gravity based patronage probability model

Huff's gravity model uses probability to describe the spatial relations between stores and consumers in a district. The attraction of a store to a given consumer is related to its size and geographical distance between them. The proportion of its attraction to all stores' total attractions in a region is the probability that a given consumer will purchase at this store. Retailers use this theory extensively for new site selection. However, the accuracy of the shopping store's preference for consumers is not precise enough. Nakanishi [15] et al., considered the other factors except for retail store area and distance factors to improve Huff's model, which is called the multiplicative competitive interaction (MCI) model. It still can't make a breakthrough. Fig. 2.1 [16] shows the theoretical store trade area. The blue, green, yellow and red progression represents zones of increasing patronage probability. Different circles denote the different circular trade area of the retailer store.

2.1.2 Customer Analysis

In customer analysis, RFM analysis [1], [2], [3] is to build a model that differentiates important customers from large transaction data. Chen and colleagues [17] propose an extended model of RFM analysis for the challenge prediction problem of customers in the logistics industry. Later, the research that combines machine learning methods has also been reported. Tanaka and colleagues [5] considered the RFM and logistic regression analysis to detect the loyal customer's preference for various supermarket products. They set the month elapsed from a customer's last shopping record to the data statistic day, the frequency of a customer comes to store, and the purchase amount of a customer has spent in a time interval for R, F and M values, respectively. Although in Tanaka's work, they obtained relatively precise results, the partial regression coefficients of RFM indicators are uncomparable since they have different magnitude. The principal component analysis (PCA) [18] and normalization analysis [19] can also play a role in data magnitude reduction. However, they have the defects of poor interpretability. Ref. [4], [12] proposed a method that can uniform the magnitude of RFM values where they group the RFM values and give a score of each group, respectively.

Decyl analysis [5] is another analysis method that calculates the purchase

ratio and the sales composition ratio of each rank by dividing the consumption of all customers into ten equal parts based on purchase history data. By purchase ratio and composition ratio, it is possible to know a loyal customer group with a high contribution to sales. The purpose of Decyl analysis is to grasp a loyal customer group and concentrate on it to implement efficient marketing.

2.1.3 Regression Analysis

Regression analysis is a statistical technique for estimating the relationship between dependent and a set of independent explanatory variables. Polynomial regression [20] is commonly used to analyze the curvilinear data when the power of an independent variable is more than one. It plays a crucial role in regression analysis because any function can be approximated piecewise by a polynomial. Zenker [21] et al., proposed a method including polynomial regression and response surface methodology for place marketing.

Logistic regression [6], [7], [8], [9] is also a regression analysis method for the dichotomous problem. It quantifies the correlation between a series of independent variables and a dependent variable as a logit odds ratio. The logit odds ratio is the natural logarithm of an odds ratio that represents the influence of the fluctuation of a given variable on the dependent variable. Yeung and Yee [22] predicted consumer purchase propensity by logistic regression analysis. They demonstrate how logistic regression can be used to predict consumer behavior where the explanatory variables are dichotomous and interact with each other. Constantin [23] used a logistic regression model in supporting decisions of establishing marketing strategies for accommodation analysis. Tanaka [5] et al., built a loyal customer analysis model consist of original RFM values and the proportion of item purchasing of a customer. They define the loyal customer of a supermarket chain by Decyl analysis and tag them as target variables. After that, they use logistic regression analysis to find loyal customers and detect the item preference of them effectively.

2.2 Loyal Customer Analysis Model

2.2.1 RFM Analysis

RFM analysis [1], [2], [3] contains three indicators, how recently a customer has purchased (Recency), how often they purchase (Frequency), and how much they spend (Monetary). To solve the problem that different magnitude RFM indicators are uncomparable in Tanaka's work [5] and make the analysis results more accurate. This study considers converting the original RFM values into the form of customer RFM scores based on Ref. [4], [12]. The analytic models set R value as days elapsed from last sales record to data statistics day, F value as the frequency of customer come to store and M value as the average of one time purchase amount in a time interval from first shopping day to data statistics day. R value is ordered by ascending and F, M value is ordered by descending. Each of them is divided into five groups according to top rank 20%, 20% to 40%, 40% to 60%, 60% to 80%, 80% to 100%, respectively. Each group of R, F and M value is scored from level 1 to 5 based on their group rank. If a customer owns three high RFM scores such as $(5, 5, 5)$ or $(5, 4, 4)$, this customer has a high loyalty in a store. Suppose a store has 100 customers, each with a different RFM value. The examples of the R, F and M score are shown in table 2.1, 2.2 and 2.3. The units of RFM values are days, times and amounts of money, respectively. Table 2.4 shows the concrete structure of experimental data for the first proposal of this thesis, the RFM type model.

Table 2.1: The example of R score

Customer	R	R rank	Percentage	R group	R score
1	65	48	48%	40%-60%	3
2	354	87	87%	80%-100%	1
3	30	28	28%	20%-40%	4

Table 2.2: The example of F score

Customer	F	F rank	Percentage	F group	F score
1	28	80	80%	60%-80%	2
2	23	82	82%	80%-100%	1
3	96	20	20%	0-20%	5

Table 2.3: The example of M score

Customer	M	M rank	Percentage	M group	M score
1	1926	49	49%	40%-60%	3
2	150	95	95%	80%-100%	1
3	2111	44	44%	40%-60%	3

2.2.2 Huff's Gravity Model

The ID-POS database contains the customer information and two year customer shopping record of target supermarket chain A including supermarkets A1 and A2 in the experimental city. This thesis converts the customer address to longitude and latitude and uses Euclidean distance [24] to compute the distance to each supermarket in the experimental city. There are seven supermarket chains in that city, including one target supermarket chain A and six competitive supermarket chains B, C, F, H, J and N. The chain name will combine with a number to denote each individual supermarket of supermarket chains. The horizontal axis of Fig. 2.7 and 2.8 shows all supermarkets in the experimental city. A new method based on Huff's gravity model [13] is considered to convert the distance factors into uniform attractiveness probability.

$$hf_{ij} = \frac{\frac{s_j}{d_{ij}^{\alpha}}}{\sum_{j=1}^{n} \frac{s_j}{d_{ij}^{\alpha}}}, \quad \sum_{j=1}^{n} hf_{ij} = 1 \tag{2.1}$$

In a district, the attraction of a retail store to a given customer is the ratio of its size denoted as s to distance between them denoted as d. Therefore, hf_{ij} indicates the attractiveness probability of customer i will go shopping at store j and α denotes the distance decline coefficient. If the size of every store

Figure 2.2: The interaction between circular trade area of stores in a district

is fixed, distance can convert into a uniform attractiveness probability. The attractiveness probability is a negative correlation to the distance. The customer obtains a significant influence when he lives close to a supermarket, vice versa. The multicollinearity problem [2] will avoid since the sum of the attractiveness probability of a customer is one, irrespective of the high or low of the RFM score. Table 2.5 shows the structure of experimental data for the second proposal, the RFM+ type model including RFM score and hf score.

2.2.3 Inverse Attractiveness Frequency

The third proposal is based on Tanaka's work [5]. They proposed inverse shop frequency to reduce the customer's item preference between different individual supermarkets of a supermarket chain, and obtained precise results for loyal customer classification. This thesis focuses on the influence of competitive supermarkets with different radius of their circular trade area impact on the customers of affiliated target supermarkets. Fig. 2.2 [25] shows an example of the interaction between the trade area of stores. It reflects the competition between the stores in a district. For different stores, the customer's shopping preference and shopping frequency are affected by distance, store area., etc. Therefore, how to mathematically measure the influence of competitors is an important issue. The polynomial regression analysis [20] is used to detect the impact of competitors firstly. This thesis conducts two experiments of polynomial regression analysis on the customer shopping data and distance data of A1 and A2. Fig. 2.3 shows the tendency of distance and each customer monthly purchased

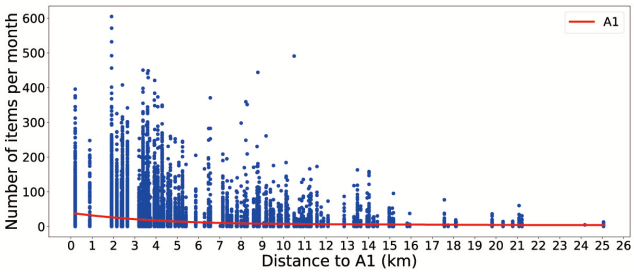


Figure 2.3: The correlation between distance and sales for A1

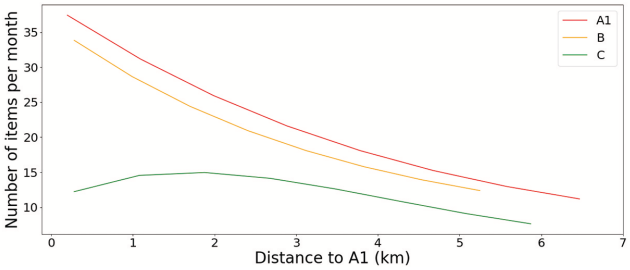


Figure 2.4: The comparison of A1 with 2 competitors

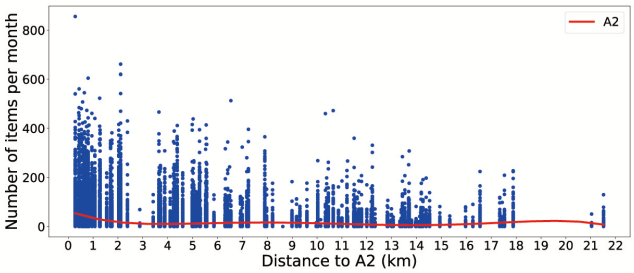


Figure 2.5: The correlation between distance and sales for A2

item number of A1. The closer the distance to A1, the higher the shopping quota is. Fig. 2.4 shows the comparison of A1 with two competitors. This thesis chooses the distance that lives around the competitors less than 3km trade area and closer than A1. The customers of A1 close to the competitors are affected since B and C are lower than A1. So do the tendencies in Fig. 2.5 and 2.6 for A2.

According to Fig. 2.3 and 2.5, this thesis defines the customers that purchased item number per month is higher than the polynomial regression curve

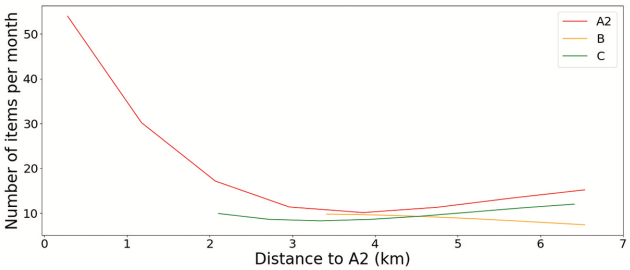


Figure 2.6: The comparison of A2 with 2 competitors

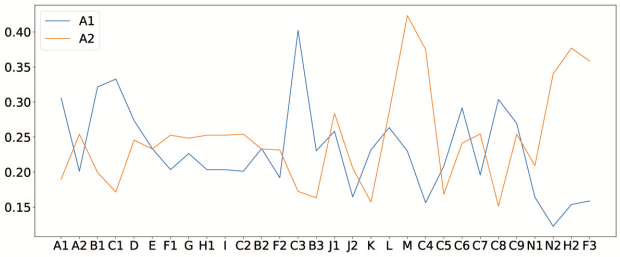


Figure 2.7: The proportion of dominated customers of A1 and A2

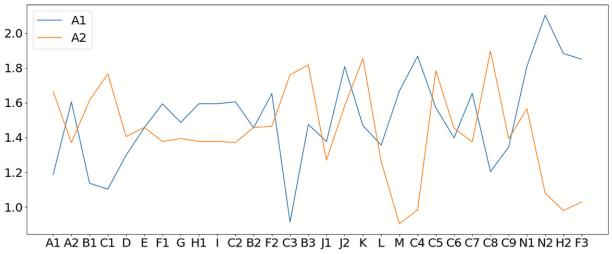


Figure 2.8: The iaf score of A1 and A2

as the dominated customers by target supermarkets. The more significant the proportion of these customers surrounding the competitive supermarkets, the powerful the target supermarkets have the influences in this area. Fig. 2.7 shows the proportion of dominated customers of A1 and A2 surrounding a 3km radius of the trade area of all supermarkets in the experimental city. The difference between A1 and A2 is from 10% to 20%. The inverse attractiveness frequency score of each supermarket is formulated to reduce this difference for feature

quantity expression as below.

$$iaf_{i,s} = log\frac{c_i}{d_{i,s}} \tag{2.2}$$

The c_i denotes the total number of customers in the specific radius of the trade area of the ith competitive supermarket. The $d_{i,s}$ indicates the number of dominated customers by target supermarket s surrounding the specific radius of the trade area of the ith competitive supermarket. The iaf vector is defined by each customer of the target supermarket chain A. The high proportion in Fig. 2.7 will have a low iaf value, which means that the target supermarket has a powerful impact on customers surrounding competitive supermarkets. Oppositely, a weak impact has a high value. In this thesis, the customers purchased at affiliated target supermarkets may be far from some competitors since this study detects almost all supermarkets in a city. The value of c_i or $d_{i,s}$ perhaps zero. The Laplacian smoothing is considered to avoid this situation.

$$iaf_{i,s} = log\frac{c_i + n\lambda}{d_{i,s} + \lambda} \tag{2.3}$$

The n denotes the total number of supermarkets in a city. The λ is a smoothing coefficient. The hf score matrix for all customers in target supermarkets A1 and A2 are computable by Eq. 2.1, and then multiply by the corresponding iaf score vector of A1 and A2 to obtain a hf-iaf score matrix, respectively. In consideration of the heterogeneity of the influence tendency of A1 and A2 shown in Fig. 2.7, the hf-iaf score is used to acquire the feature quantity expression of the customers for the third proposal. Fig. 2.8 demonstrates the iaf score of A1 and A2, where the radius of the trade area of all supermarkets is 3km. The trend is reversed from Fig 2.7. Table 2.6 shows the structure of experimental data for the RFM++ type model including RFM score and hf-iaf score.

2.2.4 Decyl Analysis

In economics, there is a theory which is known as 80/20 rule that 20 percent of customers account for 80 percent of sales. Decyl analysis is derived from this theory. This thesis refers to Tanaka's [5] research and uses Decyl analysis to define loyal customers of supermarket chain A. Decyl analysis arranges customers in descending order of customer's consumption and then divides them into ten equal groups in terms of headcount, as shown in Fig 2.9. The top three

groups generated 80.01% sales in the first year. Therefore, loyal customers of supermarket chain A are defined as the top three groups. These three groups are considered as the target variables of logistic regression analysis to build the model for loyal customer classification in the first year. Decyl analysis also conducts on the customer shopping data in the second year for testing the built models.

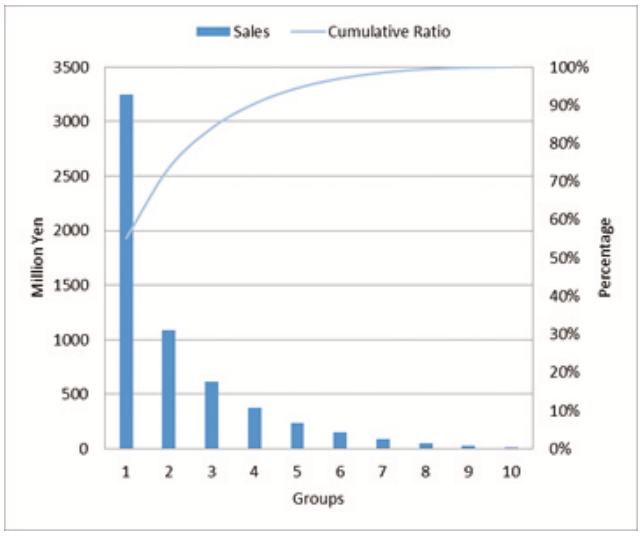


Figure 2.9: Pareto chart of customer Decyl analysis

Table 2.4: The structure of experimental data for RFM type model

Customer	R score	F score	M score
1	5	5	1
2	5	3	5
⋮			
n	4	3	2

Table 2.5: The structure of experimental data for RFM+ type model

Customer	R score	F score	M score	hf score vector
1	5	5	1	...
2	5	3	5	...
		⋮		
n	4	3	2	...

Table 2.6: The structure of experimental data for RFM++ type model

Customer	R score	F score	M score	hf-iaf score vector
1	5	5	1	...
2	5	3	5	...
		⋮		
n	4	3	2	...

2.3 Analytical methods

2.3.1 Logistic Regression Analysis

The formulation of logistic regression [6], [7], [8], [9] is defined as follows, where p_c is a probability that customer c is a loyal customer, ω denotes partial regression coefficients, x indicates explanatory variables, and d represents bias.

$$p_c = \frac{1}{1 + e^{(d+\omega_1 x_{1,c}+\omega_2 x_{2,c}+...+\omega_k x_{k,c})}} \tag{2.4}$$

This thesis uses logistic regression analysis for detecting the influence degree of the competitive supermarket on loyal customers of target supermarket chain A. The top three customer groups of Decyl analysis are tagged as target variables for the loyal customer classification. For the explanatory variables, this study adopts three RFM score indicators and the transformed distance factors by the proposed methods of this thesis.

2.3.2 Evaluation Criteria

This thesis uses the Accuracy, the Precision, the Recall and the F1-score as evaluation criteria. The formulas are defined as follows:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.5}$$

$$Precision = \frac{TP}{TP + FP} \tag{2.6}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.7}$$

$$F1\text{-}score = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{2.8}$$

Let TP be true positives that samples correctly classified as positive, FN be false negatives that samples incorrectly classified as negative, FP be false positives that samples incorrectly classified as positive, and TN be true negatives that samples correctly classified as negative. For supermarket competition analysis, this study detects the value of all partial regression coefficients and rejects all cases where the statistical significance level (p value) [19] is greater than 5%.

2.4 Experiment

2.4.1 Experimental Data

This study uses 2 year ID-POS data of a supermarket chain A including A1 and A2 in Higashihiroshima city, Hiroshima, Japan. There are 40,977,672 sales records in the ID-POS data where the number of IDs and categorized products is 176076 and 2251, respectively. In addition, there are 30 supermarkets including two target supermarkets in that city. Table 2.7 shows the detail of experimental data. The 2 year ID-POS data is segmented into the first year and the second year. The first year data is divided into train data and validation data to build the models. The second year data (test data) is a measurement for testing the constructed models.

Table 2.7: The experimental data of target supermarket chain A

	First year (2016.04-2017.03)		Second year (2017.04-2018.03)
	Train data	Validation data	Test data
A	82029	27344	111772
A1	32690	10897	44020
A2	49339	16447	67752

2.4.2 Experimental Procedure

The experiments are executed on Windows 8 with 2.50GHz Intel Core i7 and 8GB memory and conducted on the python environment. For all cases, the store area is fixed as $1000m^2$ and the distance decline coefficient is fixed as 2. The smoothing coefficient is fixed as 1.

The 2 year ID-POS data of the target supermarket chain A is divided into the first year as current customer information and the second year as future customer information. RFM score indicators will combine with 30 converted distance indicators of competitors to build feature quantities as shown in table 2.4, 2.5, 2.6. Decyl analysis will also conduct on 2 year ID-POS data to define the loyal customers, respectively. This research uses the customer ID, 33 feature quantities as explanatory variables and loyal customers as target variables to build experimental data in logistic regression analysis. The first year experimental data is divided into 2 pieces, 75% for training data and 25% for validation data. The oversampling and undersampling problems [26] are judged that it is unnecessary in this experiment. The first year experimental data is utilized to construct the models. The constructed model will implement on the second year data (test data) to classify the loyal customers. There are 2 stages of the experiments in this research. The first stage is an experiment of loyal customer classification on the entire supermarket chain A. The second stage is an experiment of loyal customer classification on individual supermarkets of the target supermarket chain A. Both of the stages contain three types of model analysis, the RFM, RFM+ and RFM++ type model. The evaluation of the model is carried out from the viewpoints of accuracy, precision, recall rate, classification accuracy (F1-score), and feature understanding of loyal customers.

2.4.3 Experiment of Supermarket Chain

Table 2.8 presents the classification results of the proposed models for target supermarket chain A. In the experiments, this thesis generates five models. 'RFM-A' model only includes 3 RFM score indicators. 'RFM+A' model contains 3 RFM score indicators and 30 hf score indicators. This research chooses 3km, 4km and 5km radius of the trade area of each supermarket in the experimental city to generate the three 'RFM++A' models for supermarket chain A. The RFM++ type model is superior to the other two type models for loyal customer classification from the viewpoint of four evaluation criteria.

Table 2.8: The accuracy analysis for chain A

	Accuracy	***Precision***	***Recall***	***F1-score***
RFM-A	91.89%	79.82%	97.47%	0.878
RFM+A	92.05%	80.28%	97.67%	0.881
RFM++3km-A	92.06%	80.31%	97.73%	0.882
RFM++4km-A	92.10%	80.46%	97.73%	0.883
RFM++5km-A	92.23%	80.78%	97.82%	0.885

Table 2.9: The RFM model for chain A

Variables	***Coefficients***	***P values***
Intercept	-20.01	0.0%
R score	0.26	0.0%
F score	3.47	0.0%
M score	1.96	0.0%

Table 2.10: The RFM+ model for chain A

Variables	***Coefficients***	***P values***
Intercept	-22.27	0.0%
R score	0.25	0.0%
F score	3.77	0.0%
M score	2.06	0.0%
A2	0.15	0.0%
A1	0.07	0.0%
C2	0.06	0.1%
C7	0.03	1.5%
C6	0.03	3.2%
G	-0.03	0.0%
C9	-0.03	0.1%
C5	-0.04	0.0%
C3	-0.04	0.1%
B2	-0.04	0.0%
J2	-0.05	0.0%
E	-0.06	0.0%
C8	-0.06	0.0%
K	-0.06	0.0%
D	-0.44	0.0%

Table 2.11: The RFM++3km model for chain A

Variables	***Coefficients***	***P values***
Intercept	-21.90	0.0%
R score	0.28	0.0%
F score	3.74	0.0%
M score	2.12	0.0%
A2	0.85	0.1%
C2	0.37	0.0%
C7	0.22	0.0%
A1	0.19	0.0%
B1	0.13	0.0%
C6	0.10	0.0%
I	-0.03	0.0%
J1	-0.03	0.0%
N1	-0.05	0.0%
M	-0.11	0.0%
C1	-0.18	0.0%
F3	-0.19	0.0%
F1	-0.23	0.0%
E	-0.38	1.8%
C5	-0.39	0.1%
C9	-0.40	1.6%
B3	-0.41	0.0%
H1	-0.45	0.5%
C4	-0.46	0.0%
C3	-0.49	0.4%
G	-0.49	0.0%
B2	-0.50	0.0%
L	-0.51	0.0%
C8	-0.54	0.0%
H2	-0.58	0.2%
J2	-0.62	0.0%
N2	-0.88	3.6%
K	-1.13	0.0%
F2	-1.76	2.6%
D	-5.39	0.0%

Table 2.12: The RFM++4km model for chain A

Variables	***Coefficients***	***P values***
Intercept	-23.51	0.0%
R score	0.28	0.0%
F score	3.81	0.0%
M score	2.14	0.0%
F2	1.63	0.0%
B1	1.54	0.0%
I	1.47	0.0%
A2	1.34	0.0%
A1	1.31	0.0%
C2	1.21	0.0%
N2	1.10	1.1%
C6	1.09	0.0%
F3	0.94	0.0%
C7	0.93	0.0%
M	0.91	0.1%
J1	0.84	0.0%
L	0.81	1.5%
F1	0.70	0.0%
C9	0.59	0.1%
H1	0.56	0.1%
H2	0.45	2.6%
G	0.45	0.4%
B3	0.45	0.0%
C5	0.41	0.4%
B2	0.40	0.3%
C4	0.37	0.0%
C3	0.25	0.0%
E	0.23	0.0%
J2	0.20	0.0%
C8	0.17	0.0%
N1	-0.13	0.0%
K	-0.30	0.0%
C1	-0.63	0.0%
D	-4.49	0.0%

Table 2.13: The RFM++5km model for chain A

Variables	***Coefficients***	***P values***
Intercept	-23.79	0.0%
R score	0.27	0.0%
F score	3.77	0.0%
M score	2.11	0.0%
B1	1.84	0.0%
A2	1.83	0.0%
A1	1.76	0.0%
M	1.72	0.0%
C2	1.67	0.0%
C6	1.48	0.0%
I	1.45	0.0%
C7	1.43	0.0%
J1	1.26	0.0%
C4	1.25	0.0%
L	1.19	0.0%
N2	1.06	1.0%
F1	1.03	0.0%
H1	0.98	0.0%
H2	0.97	0.0%
C9	0.96	0.0%
F3	0.91	0.0%
F2	0.86	0.0%
G	0.83	0.0%
N1	0.80	0.0%
B2	0.77	0.0%
C5	0.76	0.0%
J2	0.68	0.0%
E	0.66	0.1%
B3	0.59	2.7%
C8	0.54	0.1%
C3	0.53	0.9%
K	0.26	0.0%
C1	-0.31	0.0%
D	-2.52	0.6%

Table 2.9 shows the partial regression coefficients of the RFM type model for target supermarket chain A. The loyal customers have the attribution of high RFM scores since three partial regression coefficients are positive. This is consistent with intuitive understanding. By unifying the magnitude of the three indicators, the F value seems more critical for loyal customers because its value is maximal. However, it is difficult to thoroughly analyze the influence of supermarket competition by these three indicators.

Table 2.10 presents the competition analysis by the partial regression coefficients of the RFM+ type model. The coefficients are ordered by descending except for RFM scores and intercept. All competitors give loyal customers of the target supermarket chain A negative influence except for supermarket C2, C6, C7. The most of statistical significance level is less than 5%. This thesis omits the cases that the statistical significance level higher than 5%. The values of target supermarkets A1 and A2 are positive, which is consistent with the intuitive idea since the closer to them, the more likely it becomes a loyal customer. The loyal customers are most active affected by supermarket D.

Table 2.11, 2.12 and 2.13 demonstrate the partial regression coefficients of three RFM++ type models. The competitors have a powerful impact on customers of supermarket chain A who live in their 3km radius of the trade area because most of the coefficients of the 'RFM++3km-A' model are negative. When the radius of the trade area increased to 4km for all supermarkets, the influence of competitors become worse because most coefficients of the 'RFM++4km-A' model are positive. This tendency is also found in the case of the 'RFM++5km-A' model. The values of coefficients become positive and greater than the 'RFM++4km-A' model. From these results, the analysis diversity of the RFM++ type model is superior to the other two type models. In addition, the RFM++ type model can grasp the impact of all nearby competitors since the statistical significance level is all less than 5%.

2.4.4 Experiment of Individual Supermarkets

Table 2.14 and 2.17 show the comparison of three type models for A1 and A2. Similar to the experiments of supermarket chain A, this thesis employs the first year customer shopping data of A1 and A2 to build the RFM and RFM+

type models, and then classify the loyal customer in the second year. For the RFM++ type model, the constructed 3 models of target supermarket chain A as shown in table 2.11, 2.12 and 2.13 are used to classify the loyal customers of A1 and A2 in the second year. The results demonstrate the RFM++ type model is the most superior one in the loyal customer classification of individual supermarkets.

Table 2.15 and 2.18 present the RFM type model for A1 and A2, respectively. Similar to the cases in supermarket chain A, the RFM scores in both cases are all positive. The RFM+ type model for A1 and A2 are shown in table 2.16 and 2.19. It is interesting that even among A1 and A2 have competition with each other. In the case of the RFM+ type model of A1, A2 has a negative value. So does the case in the RFM+ type model of A2. The statistical significance level is confirmed that most of the cases are less than 5%.

Table 2.14: The accuracy analysis for A1

	Accuracy	***Precision***	***Recall***	***F1-score***
RFM-A1	92.33%	79.71%	95.34%	0.868
RFM+A1	92.63%	80.51%	95.38%	0.873
RFM++3km-A	92.71%	80.81%	95.84%	0.877
RFM++4km-A	92.73%	80.81%	95.87%	0.877
RFM++5km-A	92.76%	80.94%	95.92%	0.878

Table 2.15: The RFM model for A1

Variables	***Coefficients***	***P values***
Intercept	-26.76	0.0%
R score	0.38	0.0%
F score	4.64	0.0%
M score	1.99	0.0%

Table 2.16: The RFM+ model for A1

Variables	***Coefficients***	***P values***
Intercept	-25.89	0.0%
R score	0.39	0.0%
F score	4.51	0.0%
M score	2.09	0.0%
B1	0.26	0.0%
A1	0.05	0.0%
G	-0.03	1.1%
B2	-0.04	0.0%
C9	-0.04	0.1%
C8	-0.05	0.0%
E	-0.05	0.8%
C5	-0.05	0.0%
A2	-0.06	0.0%
K	-0.06	1.0%
J2	-0.06	0.0%
C1	-0.07	0.0%
D	-0.55	0.3%

Table 2.17: The accuracy analysis for A2

	Accuracy	***Precision***	***Recall***	***F1-score***
RFM-A2	91.48%	80.33%	94.80%	0.870
RFM+A2	91.58%	80.72%	96.53%	0.879
RFM++3km-A	92.85%	83.43%	97.12%	0.898
RFM++4km-A	92.98%	83.90%	96.80%	0.899
RFM++5km-A	93.05%	83.99%	96.74%	0.899

Table 2.18: The RFM model for A2

Variables	***Coefficients***	***P values***
Intercept	-23.15	0.0%
R score	0.23	0.0%
F score	3.85	0.0%
M score	2.40	0.0%

Table 2.19: The RFM+ model for A2

Variables	***Coefficients***	***P values***
Intercept	-22.76	0.0%
R score	0.23	0.0%
F score	3.85	0.0%
M score	2.48	0.0%
A2	0.17	0.0%
B1	0.16	0.0%
C2	0.09	0.0%
B3	0.05	1.2%
F1	0.04	0.4%
B2	-0.02	0.0%
G	-0.03	0.1%
K	-0.04	3.8%
J2	-0.04	0.0%
M	-0.04	0.2%
C3	-0.05	0.8%
C8	-0.05	0.0%
A1	-0.09	0.0%
C1	-0.15	0.4%
D	-0.63	0.0%

Chapter 3

Multi-conditional time series forecasting

3.1 Related Work

The autoregressive integrated moving average (ARIMA) [40] model was a milestone in the development of modern time series forecasting. The ARIMA [40] model can transform a non-stationary sequence into stationary via a differencing operation. However, the differencing process limits the performance of ARIMA [40] model, which generally amplifies high-frequency noise in time series. The support vector machine (SVM) [41] is another promising model applied to classification tasks. The support vector regression (SVR) [42] is a derivation method of SVM [41] for time series forecasting, which maps the time series into a high-dimension feature space via a non-linear mapping and performs a linear regression in this space. The SVR [42] only considers the time series globally and lacks the flexibility to capture the local trend.

The artificial neural network (ANN) [43] is a further forecasting model that imitates a biological neural network structure, which updates the internal system of artificial neurons to generate an approximate model via learning the non-linear characteristics of external information. Since the ANN [43] is not appropriate for a sequence with dependencies between variables, the recurrent neural network (RNN) [31] and improved RNN [31] named long-short term memory (LSTM) [30] appeared successively. The LSTM [30] adopts a three gates memory unit to avoid the gradient disappearance problem [44] of RNN [31] and can remember short-term memory and maintain a part of long-term memory for

a long time sequence. The variants of LSTM [30], named gated recurrent unit (GRU) [33], integrates the cell state and the hidden state of LSTM [30] into a whole, and reduces the number of gate units to improve computational efficiency while ensuring the same performance as LSTM [30].

The convolutional neural networks (CNN) [29] is another branch of ANN [43], originally applied for image recognition. The time series prediction is a variant of image recognition from a 3D to a 2D problem. The width and channels of an image are the time steps (width) and feature dimensions (channels) of a time series. The CNN [29] introduced a movable filter like a human visual reception field with fewer weight parameters than ANN [43] to observe the entire time series, which slides on a time series from left to right or the opposite direction. Since time series has contextual correlation, this mechanism outperforms ANN [43] in computational efficiency and spatial relationship extraction. The dilated causal convolutional neural networks (DC-CNN) [37] improved CNN [29] via an input skipping method to increase the receptive field without tuning filter size. The dilated depthwise separable temporal convolutional networks (DDSTCNs) [45], known from Google's Xception architecture [45] for image classification, further improved DC-CNN [37] via a depthwise separable convolution [45] and a pointwise convolution to separate input channels and merge output channels, respectively.

Subsequently, the hybrid neural network framework that aggregates multiple neural networks, such as [46, 47, 49], has developed significantly. Shen et al. [27] considered the SeriesNet for time series forecasting as illustrated in Fig. 3.1, where the DC-CNN-based [37] residual learning module [28] and LSTM [30] subnetworks have the ability for time series feature extraction.

The deep learning structures focus on the inputs from a global perspective and ignore its local trends until the emergence of the attention mechanism changed this situation. The squeeze-and-excitation networks (SeNet) [50] and the convolutional block attention module (CBAM) [35] adopt in Google ResNet [51] represent the lightweight CNN-based [29] attention mechanisms for image recognition. SeNet [50] only uses global average pooling attention in the residual learning module, but CBAM-based [35] ResNet [51] improved SeNet [50] from global average pooling and global max pooling perspective. Nauta et al. [36]

proved attention mechanism is successful in combining with the DDSTCNs [45] for time series forecasting. Unlike single attention-based structures, the encoder-decoder framework enables the multiple attention mechanisms to appear in two modules simultaneously. The previous attention has a particular influence on the latter. Yao et al. [38] verified the dual-stage attention-based encoder-decoder structure is appropriate for time series forecasting and superior to single attention-based networks.

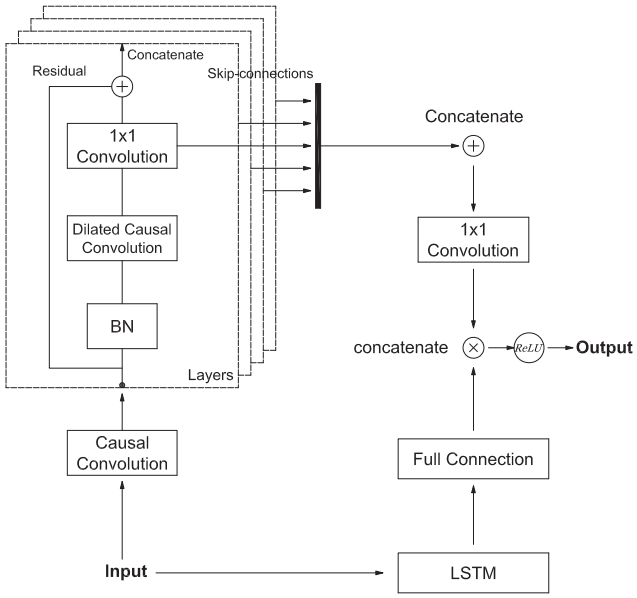


Figure 3.1: The structure of the SeriesNet.

3.2 Definition of multi-conditional time series

Given a one-dimensional target time series with T time steps (Input) $\mathbf{y} = \{y_1, y_2, \ldots, y_T\} \in \mathbb{R}^{1\times T}$, the next value y_t conditional on the sequence's history, $y_1, ..., y_{t-1}$ is predictable by maximizing the likelihood function as below:

$$p(\mathbf{y}) = \prod_{t=1}^{T} p(y_t | y_1, y_2, \ldots, y_{t-1}), \tag{3.1}$$

There exists multi-condition series (Condition) $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\} = \{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^n\} \in \mathbb{R}^{n\times T}$, where n is the feature dimension. The given target time series $\mathbf{y} = \{y_1, y_2, \ldots, y_T\} \in \mathbb{R}^{1\times T}$ conditional on these additional time series is

mathematically defined by:

$$p(\mathbf{y}|\mathbf{x}) = \prod_{t=1}^{T} p(y_t|y_1, y_2, \ldots, y_{t-1}, \mathbf{x}). \tag{3.2}$$

3.3 Attention-Based SeriesNet

This thesis improves Shen's work [27] by using two different attention mechanisms on two sub-networks of SeriesNet, respectively. The first subnet utilizes CBAM-based DDSTCNs to instead of DC-CNN [37] to learn short interval features. The stacked deep residual connection blocks [51] with different dilated rates can learn long interval features with different reception fields. The batch normalization (BN) [34] is added to solve the gradient vanishing problem. For the second subnet, HSAM-based GRU is applied instead of LSTM for learning the holistic features followed by a full connection (FC) layer to set the output dimensionality. Finally, the outputs of two sub-networks will be element-wise multiplied together for time series forecasting. The attention-based SeriesNet can directly conduct on the raw time series by conditioning methods.

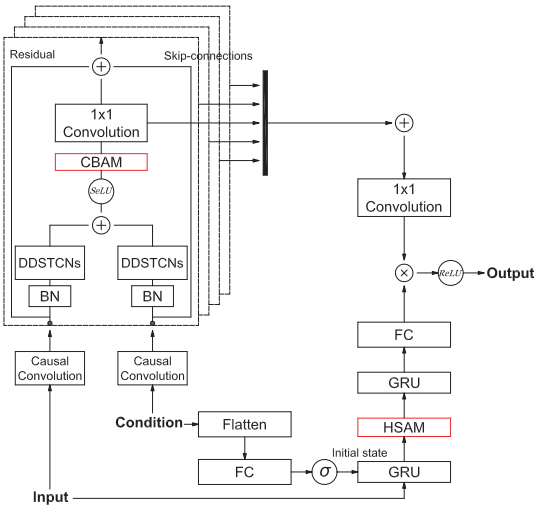


Figure 3.2: The structure of the attention-based SeriesNet.

3.3.1 Conditioning

The conditioning method for CNN is similar to Borovykh's work [32] except for the activation function. This thesis adopts the scaled exponential linear

unit (SeLU) [53] instead of the rectified linear unit (ReLU) [54] since the self-normalizing properties of the SeLU has more robust representations of the time series. As shown in Figure 3.3, the input and condition are conditioned in the first residual layer (L), followed by the CBAM [35] and the 1×1 convolution, and summed with the parametrized skip connections. The result from this layer is the input in the subsequent convolution layer with a residual connection, which is repeated to obtain the output from layer L and forwarded to a 1×1 convolution to generate the final CNN output.

This thesis presents the conditioning method for RNN based on Philipperemy's [48] work as demonstrated in Figure 3.2. The given multi-conditions $\mathbf{y} \in \mathbb{R}^{i \times T}$ is considered as the initial state of the first RNN layer by transforming its shape into $\mathbf{y} \in \mathbb{R}^{p \times m}$, where m is the unit number of the first RNN layer and p's value is 1 or 2 for GRU and LSTM, respectively. Since LSTM owns hidden state and cell state, GRU only has hidden state. In case of GRU, the flatten operation is implemented on $\mathbf{y} \in \mathbb{R}^{i \times T}$ to convert its shape into $\mathbf{y} \in \mathbb{R}^{1 \times v}$, where v is the product of i and T. The FC layer with a sigmoid activation function is followed with the flatten operation to obtain the target shape $\mathbf{y} \in \mathbb{R}^{1 \times m}$. For LSTM, this thesis first adopts flatten operation followed by a FC layer with a sigmoid activation function to transform the shape of $\mathbf{y} \in \mathbb{R}^{i \times T}$ into $\mathbf{y} \in \mathbb{R}^{1 \times 2m}$, and then reshapes it into $\mathbf{y} \in \mathbb{R}^{2 \times m}$. Each row of $\mathbf{y} \in \mathbb{R}^{2 \times m}$ is considered as the initial hidden state and initial cell state,respectively. This approach naturally solves the shape problem of multi-conditions, and also avoids polluting the inputs with additional information.

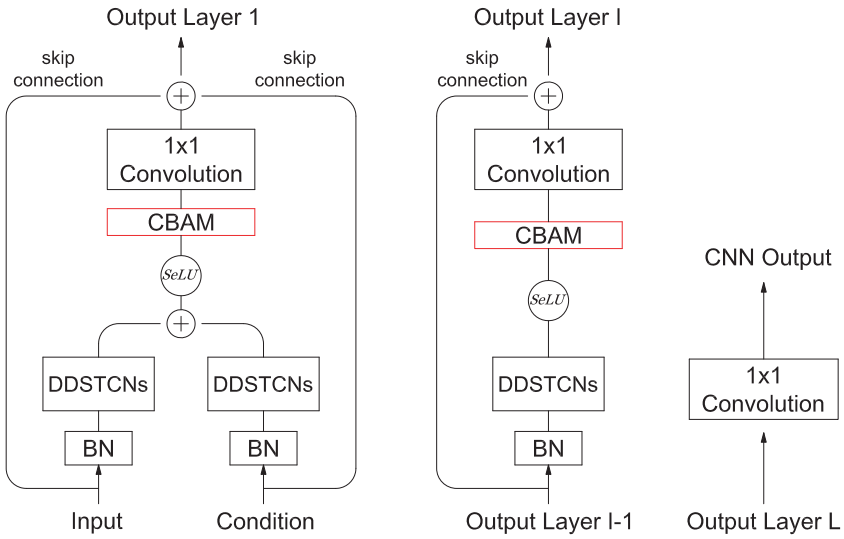


Figure 3.3: The structure of the conditional CNN sub-networks.

3.3.2 Dilated Depthwise Separable Temporal Convolutional Networks

The DDSTCNs introduced in [36] based on the depthwise separable convolution [45], which is well known by Google's Xception architecture for image classification [45]. A depthwise separable convolution splits a kernel into two separate kernels that do two convolutions: the depthwise convolution and the pointwise convolution. The depthwise convolution separates the channels by applying a different kernel to each input channel. The pointwise convolution adopts a one times one kernel to each output channel of depthwise convolution and merges them together. This architecture is different from normal CNN that two convolutions improve computation performance than only one kernel per layer. The separate channels can correctly handle each dimension of input data impacts on output data, followed by a pointwise convolution tunes the number of output channels where the multiplications between parameters reduced significantly. Our architecture consists of k channels, one for each output from batch normalization (BN) [34] layer. An overview of this architecture is shown in left subfigure of Figure 3.4. The right subfigure of Figure 3.4 is an example of stacked temporal DC-CNN, which explains the details of the left zero padding to predict the first values. The dilation rate $1, 2, 4, ..., 2^n$ is considered in the

depthwise convolution of each DDSTCNs layer to adjust the receptive field.

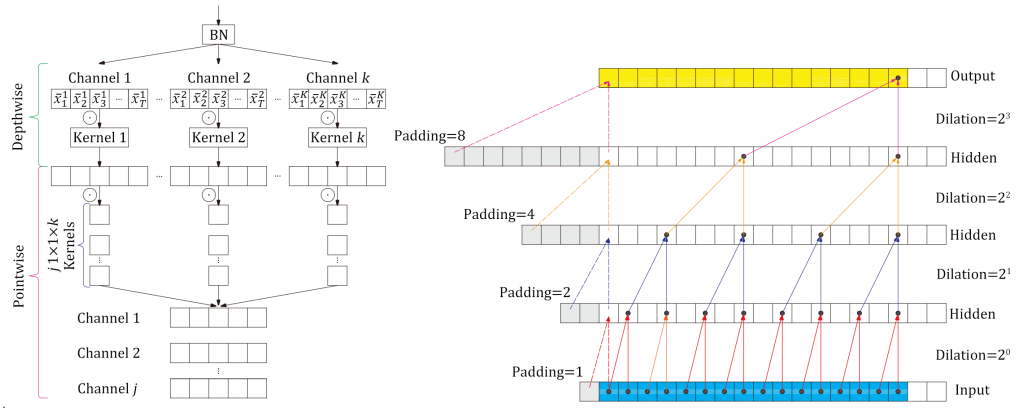


Figure 3.4: Structure of dilated depthwise separable temporal convolutional networks (left) and dilated causal convolutional neural networks (right).

3.3.3 Convolutional Block Attention Module

The CBAM [35] as shown in Fig. 3.5 adopts global average pooling and max pooling both in channel and spatial direction of a 2D image within an intermediate feature map satisfying $\mathbf{F} \in \mathbb{R}^{C\times H\times W}$, where C, H and W denotes the channel, height and width, respectively. Fig. 3.6 illustrates the details of channel attention module $\mathbf{M}_c \in \mathbb{R}^{C\times 1\times 1}$ and spatial attention module $\mathbf{M}_s \in \mathbb{R}^{1\times H\times W}$ of CBAM for 1D time series. Given an intermediate feature map $\mathbf{F} \in \mathbb{R}^{n\times T}$ as input, this thesis uses feature dimension n and time steps T of the previous layer output instead of C and W in a image. The feature (channel) attention generates time step context descriptors $\mathbf{F}^n_{avg} \in \mathbb{R}^{n\times 1}$ and $\mathbf{F}^n_{max} \in \mathbb{R}^{n\times 1}$ of a feature map by using both average and max pooling operation along the time step axis, and then fowards to a shared multi-layer perception (MLP) to produce the feature (channel) attention map $\mathbf{M}_n \in \mathbb{R}^{n\times 1}$ as:

$$\begin{aligned}\mathbf{M}_n(\mathbf{F}) &= \sigma(MLP(AvgPool(\mathbf{F}))) + MLP(MaxPool(\mathbf{F})) \\ &= \sigma(\mathbf{W}_1(\mathbf{W}_0(\mathbf{F}^n_{avg})) + \mathbf{W}_1(\mathbf{W}_0(\mathbf{F}^n_{max}))),\end{aligned} \tag{3.3}$$

where σ indicates the sigmoid activation function, the MLP weights $\mathbf{W}_0 \in \mathbb{R}^{n/r\times n}$ and $\mathbf{W}_1 \in \mathbb{R}^{n\times n/r}$ respectively followed by a ReLU and sigmoid activation function are shared for both inputs. r is the reduction ratio used to reduce the parameters in $\mathbf{W}_0$. The feature attention map $\mathbf{M}_n \in \mathbb{R}^{n\times 1}$ element-wise

multiplies the intermediate feature map $\mathbf{F} \in \mathbb{R}^{n \times T}$ to generate a new intermediate map $\mathbf{F}^{'} \in \mathbb{R}^{n \times T}$ to feed in time step (spatial) attention module:

$$\mathbf{F}^{'} = \mathbf{M}_n(\mathbf{F}) \otimes \mathbf{F}, \tag{3.4}$$

where $\otimes$ is an element-wise multiplication. The time step (spatial) attention module generates a concatenated feature descriptor $[\mathbf{F}^{'T}_{avg}; \mathbf{F}^{'T}_{max}] \in \mathbb{R}^{2 \times T}$ by applying average pooling and max pooling along the feature axis, followed by a standard convolution layer. The time step (spatial) attention map $\mathbf{M}_T \in \mathbb{R}^{1 \times T}$ is computed as:

$$\begin{aligned} \mathbf{M}_T(\mathbf{F}^{'}) &= \sigma(f^{1\times 7}([AvgPool(\mathbf{F}^{'}); MaxPool(\mathbf{F}^{'})])) \\ &= \sigma(f^{1\times 7}([\mathbf{F}^{'T}_{avg}; \mathbf{F}^{'T}_{max}])), \end{aligned} \tag{3.5}$$

where $f^{1\times 7}$ indicates a 1×7 kernel size convolution operation. At last, the element-wise multiplication between $\mathbf{M}_T \in \mathbb{R}^{1 \times T}$ and $\mathbf{F}^{'} \in \mathbb{R}^{n \times T}$ is executed to renew the intermediate feature map as:

$$\mathbf{F}^{''} = \mathbf{M}_T\left(\mathbf{F}^{'}\right) \otimes \mathbf{F}^{'}, \tag{3.6}$$

where $\mathbf{F}^{''} \in \mathbb{R}^{n \times T}$ and wil be input to next layer.

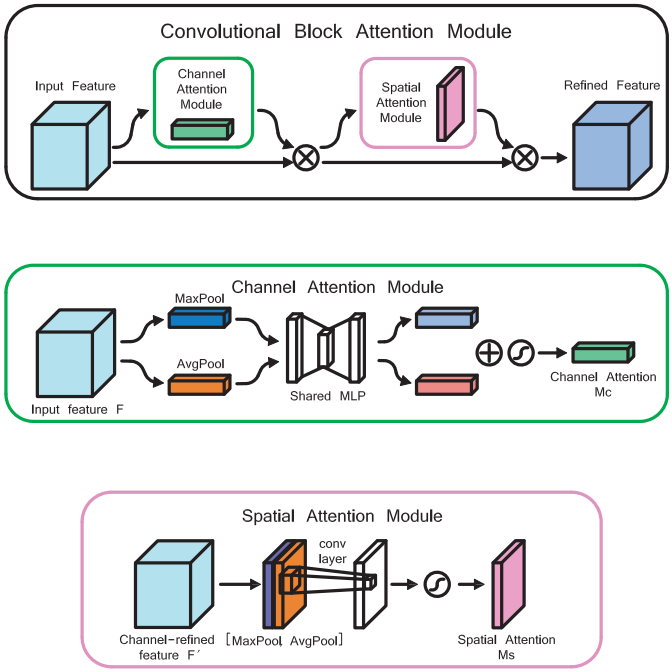


Figure 3.5: The overview of CBAM.

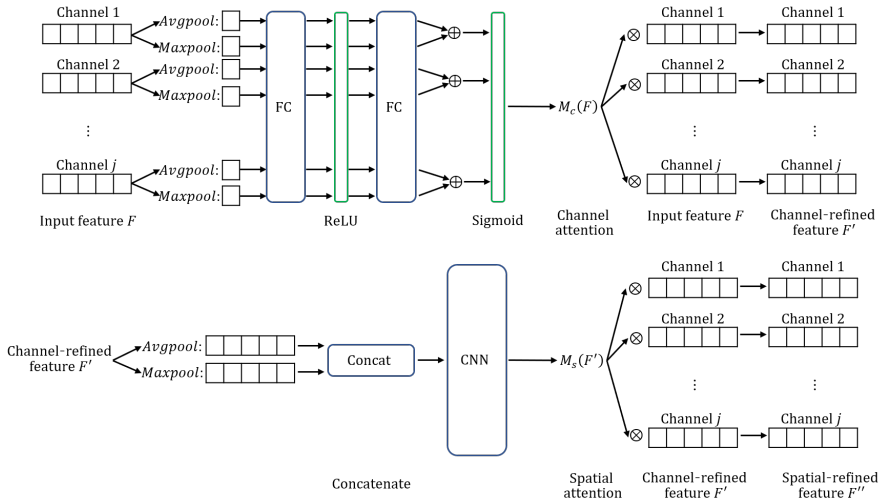


Figure 3.6: Overview of convolutional block attention module included channel attention module (top) and spatial attention module (bottom).

3.3.4 Hidden State Attention Module

This thesis presents the RNN-based HSAM by integrating the two modules of CBAM together. The HSAM is implemented between every two GRU layers as illustrated in Figure 3.7. The GRU unit merges the memory cell state and hidden state of LSTM unit into one hidden state, and reduces the three sigmoid gates of LSTM unit to two gates: reset gate $\mathbf{r}_t$ and update gate $\mathbf{z}_t$ to simplify the structure. Feeding the given one-dimensional time series with T time steps $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\} \in \mathbb{R}^{1\times T}$ in a GRU layer, the update formulas of the GRU unit are summarized as:

$$\mathbf{z}_t = \sigma(\mathbf{W}_z[\mathbf{h}_{t-1}; \mathbf{x}_t]), \tag{3.7}$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_r[\mathbf{h}_{t-1}; \mathbf{x}_t]), \tag{3.8}$$

$$\tilde{\mathbf{h}}_t = tanh(\mathbf{W}[\mathbf{r}_t \otimes \mathbf{h}_{t-1}; \mathbf{x}_t]), \tag{3.9}$$

$$\mathbf{h}_t = (\mathbf{1} - \mathbf{z}_t) \otimes \mathbf{h}_{t-1} + \mathbf{z}_t \otimes \tilde{\mathbf{h}}_t, \tag{3.10}$$

where $\mathbf{h}_t \in \mathbb{R}^{m\times 1}$ is the hidden state with size m and $\otimes$ is an element-wise multiplication. $[\mathbf{h}_{t-1}; \mathbf{x}_t] \in \mathbb{R}^{(m+n)\times 1}$ is a concatenation of the previous hidden state $\mathbf{h}_{t-1}$ and the current input $\mathbf{x}_t$. $\mathbf{W}_z$, $\mathbf{W}_r$, $\mathbf{W} \in \mathbb{R}^{m\times(m+n)}$ are weight parameters to learn. The multi GRU layers utilize per time step hidden state of previous GRU layer as an input forwarding to the corresponding state of the

next GRU layer. The input at each time step (feature axis) has great influence on the related hidden state output of the next GRU layer. Therefore, this thesis aims to extract the average pooling and max pooling only along the hidden state feature axis of the previous GRU layer. There is an intermediate feature map $\mathbf{h} \in \mathbb{R}^{m \times T}$ represents all hidden states of previous GRU layer. The hidden state attention produces feature context descriptors $\mathbf{h}_{avg}^{T} \in \mathbb{R}^{1 \times T}$ and $\mathbf{h}_{max}^{T} \in \mathbb{R}^{1 \times T}$ through average pooling and max pooling along feature axis, and feeds them into a shared MLP layer. The outputs of the shared MLP layer are concatenated together as $[\mathbf{W}_1(\mathbf{W}_0(\mathbf{h}_{avg}^{T})); \mathbf{W}_1(\mathbf{W}_0(\mathbf{h}_{max}^{T}))] \in \mathbb{R}^{2 \times T}$ followed by a standard convolution layer to obtain the hidden state map $\mathbf{H}_T \in \mathbb{R}^{1 \times T}$ as below:

$$\begin{aligned} \mathbf{H}_T(\mathbf{h}) &= \sigma(f^{1\times 7}([MLP(AvgPool(\mathbf{h}))); MLP(MaxPool(\mathbf{h})])) \\ &= \sigma(f^{1\times 7}([\mathbf{W}_1(\mathbf{W}_0(\mathbf{h}_{avg}^{T})); \mathbf{W}_1(\mathbf{W}_0(\mathbf{h}_{max}^{T}))])), \end{aligned} \tag{3.11}$$

where the MLP weights $\mathbf{W_0} \in \mathbb{R}^{m/r \times 1}$ and $\mathbf{W_1} \in \mathbb{R}^{1 \times m/r}$ with reduction ratio r are also followed by a ReLU and sigmoid activation function, respectively. Finally, the hidden state map $\mathbf{H}_T$ element-wise multiplies the intermediate feature map $\mathbf{h}$ to produce a renewed intermediate feature map $\mathbf{h}^{'} \in \mathbb{R}^{m \times T}$ feeding in next GRU layer:

$$\mathbf{h}^{'} = \mathbf{H}_T(\mathbf{h}) \otimes \mathbf{h}. \tag{3.12}$$

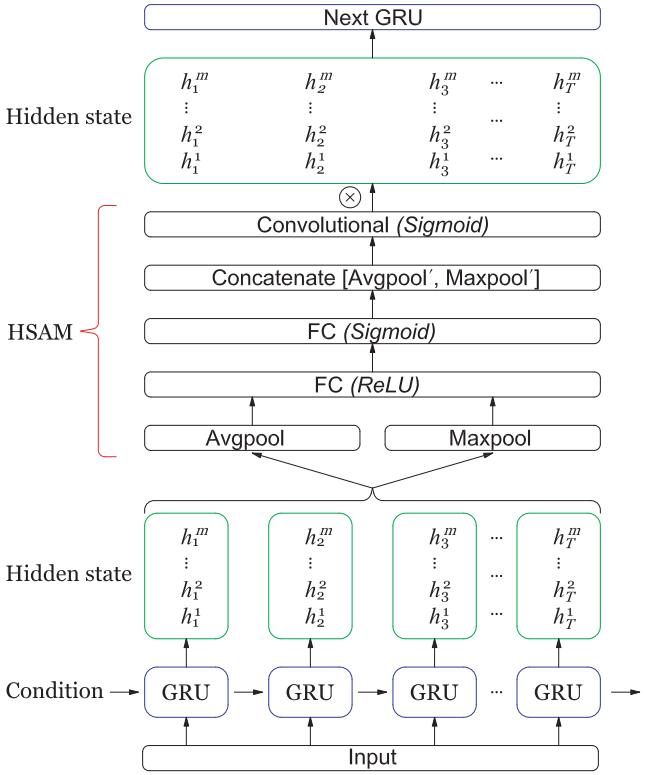


Figure 3.7: The overview of HSAM.

3.4 Triple-stage attention-based SeriesNet

This section mainly introduces the overall improvement of TA-SeriesNet compared with A-SeriesNet and clarifies the two types of subnetworks and their concatenation method of TA-SeriesNet in detail.

3.4.1 Structure of TA-SeriesNet

This thesis presents the triple-stage attention-based SeriesNet (TA-SeriesNet) as shown in Fig. 3.8 to learn this likelihood function, which includes the triple-stage attention-based long-short term memory (TA-LSTM), the triple-stage attention-based gated recurrent unit (TA-GRU) and the dual attention residual learning module-based covolutional neural network (DARLM-CNN). Since the TA-SeriesNet is reformed from A-SeriesNet, we briefly clarify their difference and improvement first according to Fig. 3.2 and Fig. 3.8.

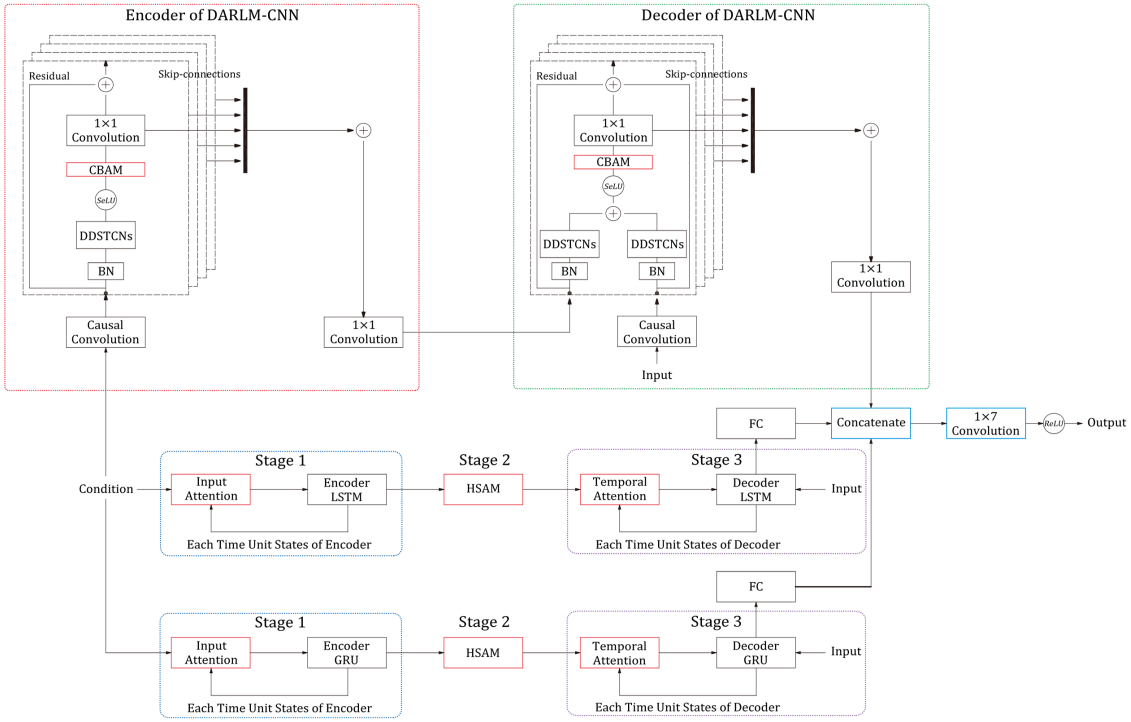


Figure 3.8: Overview of triple-stage attention-based SeriesNet included three encoder-decoder subnetworks.

3.4.1.1 Disadvantages of A-SeriesNet

The A-SeriesNet is a two subnetworks hybrid neural network architecture as illustrated in Fig. 3.2. Although it is a lightweight architecture, both of its subnetworks are not encoder-decoder structures. Therefore, it is not suitable for a high feature dimensional time series dataset that feature dimension higher than 15. We proved this point of view in our experiment section.

The CNN subnetwork conditions the multi-condition series on a target time series by simultaneously feeding them to an augmented residual learning module. With the multi-condition series's feature dimension increasing, the relation between multi-condition series becomes more complicated. Without prior feature extraction of raw multi-condition series may pollute the target time series to some extent.

The conditioning method of the RNN subnetwork reshapes the multi-condition series into the first GRU [33] layer's hidden state size in advance. The reshaped multi-condition series are fed to the first GRU [33] layer as its initial hidden state. When the multi-condition series own large feature dimensions, this method may lose some information during the reshaping process. The high dimensional multi-condition series are reshaped from $\mathbb{R}^{n\times T}$ to $\mathbb{R}^{1\times v}$, where $\mathbb{R}^{1\times v}$ is the first GRU [33] layer's hidden state size.

The RNN subnetwork only has a global HSAM attention mechanism which generates an attention weights vector for all hidden states after the first GRU [33] layer performed all its update steps. The GRU [33] layers and the HSAM are independent of each other. The mere HSAM can't detect each multi-condition series's importance for prediction results due to the reshaping preprocess.

The concatenation method of A-SeriesNet limits the number of its subnetworks. The overall prediction is liable to be impacted by either of its subnetworks.

3.4.1.2 Distinctions between A-SeriesNet and TA-SeriesNet

Aim to above disadvantages of A-SeriesNet, the improvement of TA-SeriesNet are summarized as:

- The DARLM-CNN subnetwork used ARLM-CNN as the encoder and augmented ARLM-CNN as the decoder. The TA-LSTM and TA-GRU subnet-

works simplify the DA-RNN's [38] framework and append HSAM between its encoder and decoder. All subnetworks of TA-SeriesNet are encoder-decoder structures more effective for high dimensional time series.

- The DARLM-CNN structure extracts the feature context vector from high feature dimensional multi-condition series by its ARLM-CNN encoder. The augmented ARLM-CNN is fed by the generated feature context vector and the target time series to reduce the raw multi-condition series's pollution for the target time series.

- In the TA-RNN subnetwork, the multi-condition series are fed to an RNN encoder as the input (not a hidden state) directly without shape variation, which ensures the information integrity of the multi-condition series.

- The TA-RNN subnetworks have one global HSAM attention and two local attention mechanisms, input attention and temporal attention. They extract the importance of each multi-condition series for feature context vector generation from the feature dimension axis and the time step axis. The input attention weights are updated by each time unit states of the RNN encoder. The learned each time input attention weights are element-wise multiplied by each time multi-condition series and fed to the RNN encoder again to generate its next unit states. This process will not terminate until the RNN encoder performed all its time steps. Similarly, this update process also happens between the temporal attention and the RNN decoder. The input attention, the temporal attention and the RNN encoder-decoder are not independent as illustrated in Fig. 3.8.

- This thesis adopts a new concatenation method instead of the element-wise multiplication of A-SeriesNet to free the parallel connection number of subnetworks. Each subnetwork's learnable output weight promotes to reduce the output dependence of TA-SeriesNet on a certain subnetwork.

3.4.2 Structure of DARLM-CNN subnetwork

This thesis first presents attention residual learning module-based encoder-decoder subnetwork as illustrated in Fig. 3.9 to learn the likelihood function in subsection 3.2.

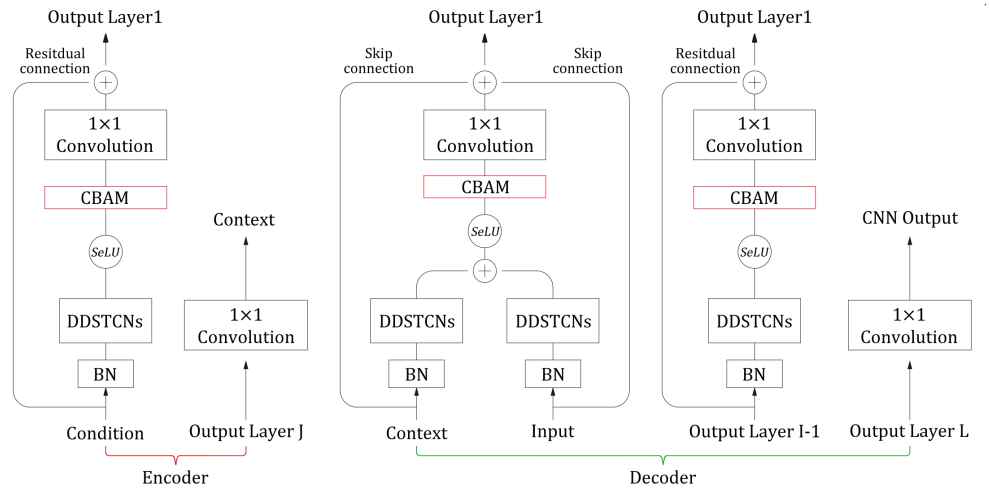


Figure 3.9: Structure of dual attention residual learning module-based convolutional neural network included ARLM-CNN encoder (red) and augmented ARLM-CNN decoder (green).

3.4.2.1 Encoder within ARLM-CNN

Traditional CNN-based [29] conditioning method [J-1,32] applied an augmented ARLM-CNN to condition multi-condition series (Condition) on target time sequence (Input) as introduced in Fig. 3.2. This structure can't extract the underlying features of multi-condition series in advance. With the increase of multi-condition series characteristic dimension (series number), the prediction accuracy will decrease gradually. Therefore, this thesis further considered an attention residual learning module as encoder to extract the latent features of the multi-condition series. As illustrated in Fig. 3.8 and 3.9, the multi-condition series $\mathbf{x} \in \mathbb{R}^{n \times T}$ are first fed forward to a causal convolution to ensure the causal relationship, followed by a batch normalization (BN) [34].

In deep neural networks, the input of the current layer is the output of the previous layer. When the neural network is trained by an optimization algorithm, such as the stochastic gradient descent (SGD) [52], each parameter variation in the previous layer will result in a different weight distribution in the current layer. The gradient vanishing named internal covariate shift [34] will occur during the training. The deeper the neural network, the more pronounced this problem is. The BN [34] layer normalizes neural network layers to keep their weight distribution stable.

The DDSTCNs [45] is the next layer, which merges the ability of DC-CNN [37] and depthwise separable convolution [45] to reduce computational parameters as illustrated in the left subgraph of Fig. 3.4. The depthwise separable convolution [45] first applies the depthwise convolution with different kernels to separate the input channels. The depthwise convolution of each DDSTCNs [45] layer adopts different dilation rate $1, 2, 4, ..., 2^n$ to expand the receptive field without adjusting the kernel size. The left zero padding is considered in the depthwise convolution of each DDSTCNs [45] layer to ensure the shape of output is same as the input. This process is similar to the stacked temporal DC-CNN [37] as shown in the right subgraph of Fig. 3.4. Then the pointwise convolution with 1×1 kernel size integrates each output channel of depthwise convolution together and tunes the number of output channels. The output of BN [34] layer $\overline{\mathbf{x}} \in \mathbb{R}^{k \times T}$ is fed forward to the DDSTCNs [45] layer followed by a scaled exponential linear unit (SeLU) [53] activation function as:

$$\mathbf{F} = SeLU(f_d^{1 \times h}(\overline{\mathbf{x}})), \tag{3.13}$$

where $f_d^{1 \times h}$ denotes the filter size $1 \times h$ and the dilation rate d of depthwise convolution. The pointwise convolution adjusts the channel number (feature dimension) of depthwise convolution to generate the output $\mathbf{F} \in \mathbb{R}^{j \times T}$ of DDSTCNs [45].

The followed CBAM [35] concentrates on the output of DDSTCNs [45] from the perspecitve of channel attention and spatial attention as illustrated in Fig. 3.6. The channel (feature) attention adopts average pooling and max pooling operation along the time step axis of $\mathbf{F} \in \mathbb{R}^{j \times T}$ to generate two context descriptors $\mathbf{F}_{avg}^j \in \mathbb{R}^{j \times 1}$ and $\mathbf{F}_{max}^j \in \mathbb{R}^{j \times 1}$. The channel attention map $\mathbf{M}_c \in \mathbb{R}^{j \times 1}$ is produced by a shared multi-layer perception (MLP) including two FC layers:

$$\begin{aligned} \mathbf{M}_c(\mathbf{F}) &= \sigma(MLP(AvgPool(\mathbf{F}))) \\ &\quad + MLP(MaxPool(\mathbf{F})) \\ &= \sigma(\mathbf{W}_1(\mathbf{W}_0(\mathbf{F}_{avg}^j)) + \mathbf{W}_1(\mathbf{W}_0(\mathbf{F}_{max}^j))), \end{aligned} \tag{3.14}$$

where σ represents the sigmoid activation function. The rectified linear unit (ReLU) [54] activation function is behind the first FC layer with weights $\mathbf{W}_0 \in$

$\mathbb{R}^{j/r \times j}$. The second FC layer with weights $\mathbf{W}_1 \in \mathbb{R}^{j \times j/r}$ is followed by a sigmoid activation function. The reduction ratio r is considered for parameters reduction of $\mathbf{W}_0$. The ReLU [54] activation function and bias parameters are ommitted in the above equation for brief. The channel attention map $\mathbf{M}_c \in \mathbb{R}^{j \times 1}$ element-wise multiplies by the DDSTCNs's [45] output $\mathbf{F} \in \mathbb{R}^{j \times T}$ to generate an intermediate map $\mathbf{F}^{'} \in \mathbb{R}^{j \times T}$, which will be fed forward to spatial attention module:

$$\mathbf{F}^{'} = \mathbf{M}_c\left(\mathbf{F}\right) \otimes \mathbf{F}, \tag{3.15}$$

where $\otimes$ indicates an element-wise multiplication. The spatial attention module adopts average pooling and max pooling along the feature axis to generate a concatenated feature descriptor $[\mathbf{F}^{'T}_{avg}; \mathbf{F}^{'T}_{max}] \in \mathbb{R}^{2 \times T}$. A standard convolution operation $f^{1 \times 7}$ with 1×7 kernel size followed by a sigmoid activation function is implemented on the concatenated feature descriptor to obtain the spatial attention map $\mathbf{M}_s \in \mathbb{R}^{1 \times T}$ as below:

$$\begin{aligned} \mathbf{M}_s(\mathbf{F}^{'}) &= \sigma(f^{1 \times 7}([AvgPool(\mathbf{F}^{'}); MaxPool(\mathbf{F}^{'})])) \\ &= \sigma(f^{1 \times 7}([\mathbf{F}^{'T}_{avg}; \mathbf{F}^{'T}_{max}])). \end{aligned} \tag{3.16}$$

The final feature map $\mathbf{F}^{''} \in \mathbb{R}^{j \times T}$ of CBAM [35] is updated by the element-wise multiplication between $\mathbf{M}_s$ and $\mathbf{F}^{'}$:

$$\mathbf{F}^{''} = \mathbf{M}_s\left(\mathbf{F}^{'}\right) \otimes \mathbf{F}^{'}. \tag{3.17}$$

The output of CBAM [35] passes through a standard 1×1 convolution to generate the output of each residual layer and unify their feature dimension. The skip-connection [28] is considered in residual layer to reduce the gradient vanishment [44]. The encoder stacks J residual layers to substitute the desired mapping $\mathcal{H}(\mathbf{x})$ by $\mathcal{H}(\mathbf{x}) - \mathbf{x}$. The identity mapping of the input can be learned by approximating their difference to zero. This method further improved the network degradation problem [44], where deep learning structure can't find the optimal weights via standard back-propagation. The residual-connection [28] followed by a 1×1 convolution reduces the number of filters back to one to produce a feature context vector $\mathbf{c} \in \mathbb{R}^{1 \times T}$, which will be input to the decoder.

3.4.2.2 Decoder within augmented ARLM-CNN

The decoder is similar to the encoder except the first residual layer of decoder, which contains two inputs, the target time series $\mathbf{y} = \{y_1, y_2, \ldots, y_T\} \in \mathbb{R}^{1\times T}$ and the context vector of the encoder $\mathbf{c} \in \mathbb{R}^{1\times T}$. This structure herits from the augmented ARLM-CNN of A-SeriesNet. After the given target time series passes through a causal convolution, the context vector $\mathbf{c} \in \mathbb{R}^{1\times T}$ and the causal convolution's output is input to the BN [34] layer, respectively. The target time series (Input) conditional on the multi-condition series (Condition) $\mathbf{x} \in \mathbb{R}^{n\times T}$ is calculated by the activation function of the convolution as below:

$$SeLU(f_d^{1\times k}(\overline{\mathbf{c}}) + f_d^{1\times h}(\overline{\mathbf{y}})), \tag{3.18}$$

where $\overline{\mathbf{c}} \in \mathbb{R}^{1\times T}$ and $\overline{\mathbf{y}} \in \mathbb{R}^{1\times T}$ indicates the output of BN [34] layer for the context vector and the target time series, respectively. The $f_d^{1\times k}$ and $f_d^{1\times h}$ differentially represents the $1\times k$ and $1\times h$ depthwise convolution of DDSTCNs [45] with dilation rate d. As illustrated in Fig. 3.8 and 3.9, the conditioned series sequentially accesses to the CBAM [35] and the 1×1 convolution in the first residual layer, and then sums with the two parametrized skip connections [32]. This layer's output connects with other residual layers that have only one input and repeated $L-1$ times. The last 1×1 convolution is applied to adjust the feature dimension (channel) of output layer L to produce the final DARLM-CNN output $\hat{\mathbf{O}}_{CNN} \in \mathbb{R}^{1\times T}$.

3.4.3 Structure of TA-RNN subnetwork

The structure of TA-LSTM (top) and TA-GRU (bottom) subnetworks included DA-RNN [38] and HSAM are illustrated in Fig. 3.10. The DA-RNN [38] applies encoder with input attention to extract the latent feature of multi-condition series as context vectors to reduce the high feature dimension's (series number) influence on prediction results. The decoder with temporal attention conditions context vectors on target time series for time series forecasting. The HSAM aims to further detect the importance of each hidden state of the encoder for context vector generation.

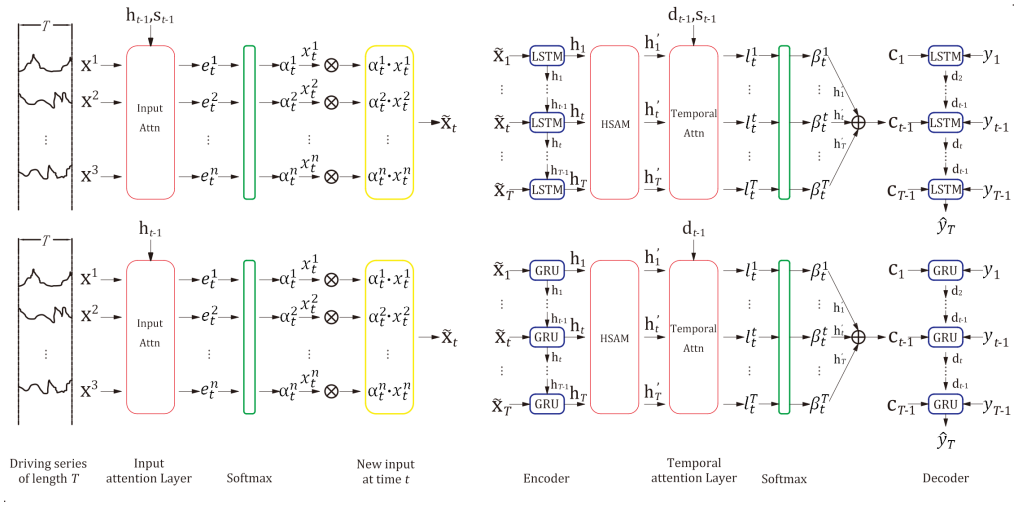


Figure 3.10: Structure of triple-stage attention-based long-short term memory (top) and triple-stage attention-based gated recurrent unit (bottom).

3.4.3.1 Encoder with input attention

The encoder learns a feature representation of the given multi-condition series $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\} \in \mathbb{R}^{n \times T}$ via a linear mapping from $\mathbf{x}_t$ to $\mathbf{h}_t$ as below:

$$\mathbf{h}_t = f_1(\mathbf{h}_{t-1}, \mathbf{x}_t), \tag{3.19}$$

where $\mathbf{h}_t \in \mathbb{R}^{m \times 1}$ is the hidden state of the encoder at time step t, f_1 denotes a non-linear relation could be learned by a RNN [31]. This thesis applies LSTM [30] as the first f_1 for learning holistic dependencies. The update formula of a LSTM [30] unit is summarized by a combination of three sigmoid functions:

$$\mathbf{f}_t = \sigma(\mathbf{W}_f[\mathbf{h}_{t-1}; \mathbf{x}_t] + \mathbf{b}_f), \tag{3.20}$$

$$\mathbf{i}_t = \sigma(\mathbf{W}_i[\mathbf{h}_{t-1}; \mathbf{x}_t] + \mathbf{b}_i), \tag{3.21}$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_o[\mathbf{h}_{t-1}; \mathbf{x}_t] + \mathbf{b}_o), \tag{3.22}$$

$$\mathbf{s}_t = \mathbf{f}_t \otimes \mathbf{s}_{t-1} + \mathbf{i}_t \otimes tanh(\mathbf{W}_s[\mathbf{h}_{t-1}; \mathbf{x}_t] + \mathbf{b}_s), \tag{3.23}$$

$$\mathbf{h}_t = \mathbf{o}_t \otimes tanh(\mathbf{s}_t), \tag{3.24}$$

where forget gate $\mathbf{f}_t$, input gate $\mathbf{i}_t$ and output gate $\mathbf{o}_t$ control the information access into the LSTM [30] unit at time step t and generate the corresponding cell state $\mathbf{s}_t \in \mathbb{R}^{m \times 1}$ and hidden state $\mathbf{h}_t \in \mathbb{R}^{m \times 1}$. $[\mathbf{h}_{t-1}; \mathbf{x}_t] \in \mathbb{R}^{(m+n) \times 1}$ represents a concatenation of hidden state $\mathbf{h}$ at time $t-1$ and multi-condition series $\mathbf{x}$ at time

t. $\mathbf{W}_f$, $\mathbf{W}_i$, $\mathbf{W}_o \in \mathbb{R}^{m\times(m+n)}$ and $\mathbf{b}_f$, $\mathbf{b}_i$, $\mathbf{b}_o \in \mathbb{R}^{m\times 1}$ are weight parameters and bias parameters to learn. σ and $\otimes$ denotes sigmoid activation function and element-wise multiplication, respectively. The given multi-condition series (Condition), the hidden state and the cell state are fed into the input attention simultaneously:

$$e_t^k = \mathbf{W}_e[\mathbf{h}_{t-1}; \mathbf{s}_{t-1}; \mathbf{x}^k], \quad 1 \le k \le n, \tag{3.25}$$

where $[\mathbf{h}_{t-1}; \mathbf{s}_{t-1}; \mathbf{x}^k] \in \mathbb{R}^{(2m+T)\times 1}$ is a concatenation of hidden state $\mathbf{h}$ at time $t-1$, cell state $\mathbf{s}$ at time $t-1$ and kth multi-condition series. $\mathbf{W}_e \in \mathbb{R}^{1\times(2m+T)}$ is the learnable weight parameters of a FC layer. This thesis adopts one FC layer in the input attention to simplify the structure, where DA-RNN [38] adopted two FC layers with a tanh activation function.

The GRU [33] is considered as the second f_1 in this thesis, which reduces three sigmoid gates of LSTM [30] unit to two gates: reset gate $\mathbf{r}_t$ and update gate $\mathbf{z}_t$. Given the same multi-condition series $\mathbf{x} \in \mathbb{R}^{n\times T}$, the update formulas of the GRU [33] unit are summarized as:

$$\mathbf{z}_t = \sigma(\mathbf{W}_z[\mathbf{h}_{t-1}; \mathbf{x}_t] + \mathbf{b}_z), \tag{3.26}$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_r[\mathbf{h}_{t-1}; \mathbf{x}_t] + \mathbf{b}_r), \tag{3.27}$$

$$\tilde{\mathbf{h}}_t = tanh(\mathbf{W}[\mathbf{r}_t \otimes \mathbf{h}_{t-1}; \mathbf{x}_t] + \mathbf{b}_{\tilde{h}}), \tag{3.28}$$

$$\mathbf{h}_t = (\mathbf{1} - \mathbf{z}_t) \otimes \mathbf{h}_{t-1} + \mathbf{z}_t \otimes \tilde{\mathbf{h}}_t, \tag{3.29}$$

where $\mathbf{h}_t$ is the hidden state with size m and $\otimes$ is an element-wise multiplication. σ and tanh represents sigmoid and tanh activation function, repecitively. $[\mathbf{h}_{t-1}; \mathbf{x}_t] \in \mathbb{R}^{(m+n)\times 1}$ is a concatenation of the hidden state $\mathbf{h}$ at time $t-1$ and the multi-condition series $\mathbf{x}$ at time t. $\mathbf{W}_z$, $\mathbf{W}_r$, $\mathbf{W} \in \mathbb{R}^{m\times(m+n)}$ and $\mathbf{b}_z, \mathbf{b}_r, \mathbf{b}_{\tilde{h}} \in \mathbb{R}^{m\times 1}$ are parameters to learn. Different from TA-LSTM, the input attention aims at the hidden state $\mathbf{h}$ at time t and kth multi-condition series for TA-GRU as:

$$e_t^k = \mathbf{W}_e[\mathbf{h}_{t-1}; \mathbf{x}^k], \quad 1 \le k \le n, \tag{3.30}$$

where $[\mathbf{h}_{t-1}; \mathbf{x}^k] \in \mathbb{R}^{(m+T)\times 1}$ and $\mathbf{W}_e \in \mathbb{R}^{1\times(m+T)}$ denote the concatenation input and learnble weight of the FC layer.

The softmax function follows with the input attention to ensure the sum of n attention weights at each time is one, which is denoted as:

$$\alpha_t^k = \frac{e_t^k}{\Sigma_{i=1}^n e_t^i}. \tag{3.31}$$

The n attention weights at each time element-wise multiply by the related multi-condition series as:

$$\tilde{\mathbf{x}}_t = (\alpha_t^1 x_t^1, \alpha_t^2 x_t^2, \ldots, \alpha_t^n x_t^n), \tag{3.32}$$

where $\tilde{\mathbf{x}}_t \in \mathbb{R}^{m\times 1}$ is the renewed multi-condition series at time t. The hidden state $\mathbf{h}_t$ is updated by $\tilde{\mathbf{x}}_t$ as follows:

$$\mathbf{h}_t = f_1(\mathbf{h}_{t-1}, \tilde{\mathbf{x}}_t). \tag{3.33}$$

3.4.3.2 Hidden state attention module

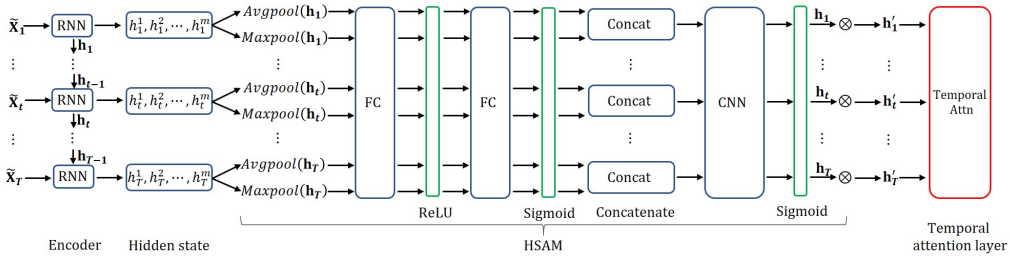


Figure 3.11: Overview of hidden state attention module between the encoder and the temporal attention.

The hidden state attention module (HSAM) is a variant of CBAM [35] as illustrated in Fig. 3.11. Given a two RNN [31] layers structure, the previous RNN [31] layer's hidden state at each time step (feature axis) is the next RNN [31] layer's corresponding input and will significantly impact each time output of the next RNN [31] layer. The HSAM focuses on global max pooling and global average pooling of hidden states between two RNN [31] layers along feature axis, which successfully transplanted the idea of CBAM [35] from CNN [29] to RNN [31]. The DA-RNN [38] presented input attention and temporal attention to deal with the performance deterioration of encoder-decoder framework for the long input sequence [39]. This thesis further considered HSAM between input attention and temporal attention to improve the above problem. The given hidden states of encoder is described by an intermediate feature map $\mathbf{h} \in \mathbb{R}^{m\times T}$. The HSAM produces feature context descriptors $\mathbf{h}_{avg}^T \in \mathbb{R}^{1\times T}$ and $\mathbf{h}_{max}^T \in \mathbb{R}^{1\times T}$ via average pooling and max pooling along feature axis. The feature context descriptors are fed forward to two shared FC layers. The outputs of the shared FC layers are concatenated together as

$[\mathbf{W}_1(\mathbf{W}_0(\mathbf{h}_{avg}^T)); \mathbf{W}_1(\mathbf{W}_0(\mathbf{h}_{max}^T))] \in \mathbb{R}^{2\times T}$ followed by a standard convolution layer to obtain the hidden state map $\mathbf{H}_T \in \mathbb{R}^{1\times T}$ as below:

$$\begin{aligned}\mathbf{H}_T(\mathbf{h}) &= \sigma(f^{1\times 7}([MLP(AvgPool(\mathbf{h}))); \\ &\quad MLP(MaxPool(\mathbf{h})])) \\ &= \sigma(f^{1\times 7}([\mathbf{W}_1(\mathbf{W}_0(\mathbf{h}_{avg}^T)); \\ &\quad \mathbf{W}_1(\mathbf{W}_0(\mathbf{h}_{max}^T))])),\end{aligned} \tag{3.34}$$

where $\mathbf{W_0} \in \mathbb{R}^{m/r\times 1}$ and $\mathbf{W_1} \in \mathbb{R}^{1\times m/r}$ are the learnable weights with reduction ratio r. The first share FC layer is sequentially followed by a ReLU [54] activation function, the second share FC layer and a sigmoid activation function. We also omit the [54] activation function and bias parameters in Eqn. (3.34). Finally, the hidden state map $\mathbf{H}_T$ element-wise multiplies by the intermediate feature map $\mathbf{h}$ to generate an updated feature map $\mathbf{h}^{'} \in \mathbb{R}^{m\times T}$, which is the input of temporal attention:

$$\mathbf{h}^{'} = \mathbf{H}_T(\mathbf{h}) \otimes \mathbf{h}. \tag{3.35}$$

3.4.3.3 Decoder with temporal attention

The temporal attention is applied to alleviate the encoder-decoder framework's performance deterioration one more time, which screens the HSAM's output before decoded by the next RNN [31] layer. Since the TA-LSTM subnetwork adopts a LSTM [30] decoder, the temporal attention of TA-LSTM concatenates the decoder's hidden state $\mathbf{d}$ and cell state $\mathbf{s}^{'}$ at time $t-1$ with the ith HSAM's output, which is denoted as $[\mathbf{d}_{t-1}; \mathbf{s}_{t-1}^{'}; \mathbf{h}_i^{'}] \in \mathbb{R}^{(2p+m)\times 1}$. p is the state size of the decoder. The concatenation is fed forward into a FC layer followed by a tanh activation function and another FC layer:

$$l_t^i = \mathbf{W}_d^{'}(tanh(\mathbf{W}_d[\mathbf{d}_{t-1}; \mathbf{s}_{t-1}^{'}; \mathbf{h}_i^{'}])), \quad 1 \le i \le T, \tag{3.36}$$

where $\mathbf{W}_d \in \mathbb{R}^{(2p+m)\times(2p+m)}$ and $\mathbf{W}_d^{'} \in \mathbb{R}^{1\times(2p+m)}$ are weight parameters to learn.

The temporal attention of TA-GRU subnetwork is different from TA-LSTM since TA-GRU adopts a GRU [33] decoder. Therefore, the Eqn. (3.36) becomes to:

$$l_t^i = \mathbf{W}_d^{'}(tanh(\mathbf{W}_d[\mathbf{d}_{t-1}; \mathbf{h}_i^{'}])), \quad 1 \le i \le T, \tag{3.37}$$

where the concatenation only contains the decoder's (GRU) hidden state $\mathbf{d}$ at time $t-1$ and the ith HSAM's output. The shape of learnable weight parameters satisfy $\mathbf{W}_d \in \mathbb{R}^{(p+m)\times(p+m)}$ and $\mathbf{W}_d^{'} \in \mathbb{R}^{1\times(p+m)}$. We omit the bias parameters in Eqn. (3.36) and (3.37) to be concise.

Then we keep the T temporal attention weights at each time step sum to one via the followed softmax function:

$$\beta_t^i = \frac{l_t^i}{\Sigma_{j=1}^T l_t^j}, \tag{3.38}$$

Each context vector is the weighted sum of element-wise multiplication between temporal attention weights per time step and corresponding HSAM's output:

$$\mathbf{c}_t = \Sigma_{i=1}^T \beta_t^i \mathbf{h}_i^{'}. \tag{3.39}$$

We concatenate the context vector $\mathbf{c}_t \in \mathbb{R}^{m\times 1}$ with the given target time series $\mathbf{y} = \{y_1, y_2, \ldots, y_T\} \in \mathbb{R}^{1\times T}$ at each time step as $[\mathbf{c}_{t-1}; y_{t-1}] \in \mathbb{R}^{(m+1)\times 1}$ and input it to a FC layer:

$$\tilde{y}_{t-1} = \tilde{\mathbf{W}}[\mathbf{c}_{t-1}; y_{t-1}] + \tilde{b}, \tag{3.40}$$

where the learnable weight $\tilde{\mathbf{W}} \in \mathbb{R}^{1\times(m+1)}$ and bias $\tilde{b} \in \mathbb{R}$ map the concatenation as $\tilde{y}_t$ to be input to the decoder. The decoder's hidden state is updated by the new input as:

$$\mathbf{d}_t = f_2(\mathbf{d}_{t-1}, \tilde{y}_{t-1}). \tag{3.41}$$

The TA-LSTM subnetwork adopts a LSTM [30] decoder as f_2. The update formula of $\mathbf{d}_t$ becomes to:

$$\mathbf{f}_t^{'} = \sigma(\mathbf{W}_f^{'}[\mathbf{d}_{t-1}; \tilde{y}_{t-1}] + \mathbf{b}_f^{'}), \tag{3.42}$$

$$\mathbf{i}_t^{'} = \sigma(\mathbf{W}_i^{'}[\mathbf{d}_{t-1}; \tilde{y}_{t-1}] + \mathbf{b}_i^{'}), \tag{3.43}$$

$$\mathbf{o}_t^{'} = \sigma(\mathbf{W}_o^{'}[\mathbf{d}_{t-1}; \tilde{y}_{t-1}] + \mathbf{b}_o^{'}), \tag{3.44}$$

$$\mathbf{s}_t^{'} = \mathbf{f}_t^{'} \otimes \mathbf{s}_{t-1}^{'} + \mathbf{i}_t^{'} \otimes tanh(\mathbf{W}_s^{'}[\mathbf{d}_{t-1}; \tilde{y}_{t-1}] + \mathbf{b}_s^{'}), \tag{3.45}$$

$$\mathbf{d}_t = \mathbf{o}_t^{'} \otimes tanh(\mathbf{s}_t^{'}), \tag{3.46}$$

where $[\mathbf{d}_{t-1}; \tilde{y}_{t-1}] \in \mathbb{R}^{(p+1)\times 1}$ represents a concatenation of previous hidden state $\mathbf{d}_{t-1}$ and decoder input $\tilde{y}_{t-1}$. $\mathbf{W}_f^{'}$, $\mathbf{W}_i^{'}$, $\mathbf{W}_o^{'}$, $\mathbf{W}_s^{'} \in \mathbb{R}^{p\times(p+1)}$ and $\mathbf{b}_f^{'}$, $\mathbf{b}_i^{'}$, $\mathbf{b}_o^{'}$, $\mathbf{b}_s^{'} \in \mathbb{R}^{p\times 1}$ are parameters to learn.

The TA-GRU subnetwork applies a GRU [33] decoder as f_2. The decoder hidden state $\mathbf{d}_t$ is updated by:

$$\mathbf{z}_t^{'} = \sigma(\mathbf{W}_z^{'}[\mathbf{d}_{t-1}; \tilde{y}_{t-1}] + \mathbf{b}_z^{'}), \tag{3.47}$$

$$\mathbf{r}_t^{'} = \sigma(\mathbf{W}_r^{'}[\mathbf{d}_{t-1}; \tilde{y}_{t-1}] + \mathbf{b}_r^{'}), \tag{3.48}$$

$$\tilde{\mathbf{d}}_t^{'} = tanh(\mathbf{W}^{'}[\mathbf{r}_t \otimes \mathbf{d}_{t-1}; \tilde{y}_{t-1}] + \mathbf{b}_{\tilde{d}}^{'}), \tag{3.49}$$

$$\mathbf{d}_t = (\mathbf{1} - \mathbf{z}_t^{'}) \otimes \mathbf{d}_{t-1} + \mathbf{z}_t^{'} \otimes \tilde{\mathbf{d}}_t^{'}, \tag{3.50}$$

where the concatenation of previous hidden state $\mathbf{d}_{t-1}$ and decoder input $\tilde{y}_{t-1}$ satifies $[\mathbf{d}_{t-1}; \tilde{y}_{t-1}] \in \mathbb{R}^{(p+1)\times 1}$. $\mathbf{W}_z^{'}$, $\mathbf{W}_r^{'}$, $\mathbf{W}^{'} \in \mathbb{R}^{p\times(p+1)}$ and $\mathbf{b}_z^{'}$, $\mathbf{b}_r^{'}$, $\mathbf{b}_{\tilde{d}}^{'} \in \mathbb{R}^{p\times 1}$ are weight and bias parameters to learn.

Finally, we feed the concatenation of the current decoder hidden state $\mathbf{d}_t$ and the current context vector $\mathbf{c}_t$, denoted as $[\mathbf{d}_t; \mathbf{c}_t] \in \mathbb{R}^{(p+m)\times 1}$, to an FC layer to approximate the current output $\hat{\mathbf{y}}_t$:

$$\hat{\mathbf{y}}_t = \mathbf{W}_y[\mathbf{d}_t; \mathbf{c}_t] + b_w, \tag{3.51}$$

where $\mathbf{W}_y \in \mathbb{R}^{1\times(p+m)}$ and $b_w \in \mathbb{R}$ are learnable parameters. Unlike the final step of DA-RNN [38], we simplify its two FC layers to one FC layer. The final output of TA-LSTM and TA-GRU subnetwork satisfies $\hat{\mathbf{O}}_{LSTM} \in \mathbb{R}^{1\times T}$ and $\hat{\mathbf{O}}_{GRU} \in \mathbb{R}^{1\times T}$, respectively.

3.4.4 Concatenation of subnetworks

This thesis presents a new concatenation method for the subnetwork number of hybrid neural network structure more than two. The final output of three subnetworks are concatenated together and fed forward to a convolution followed by a ReLU [54] activation function as:

$$\hat{\mathbf{O}} = ReLU(f^{1\times 7}([\hat{\mathbf{O}}_{CNN}; \hat{\mathbf{O}}_{LSTM}; \hat{\mathbf{O}}_{GRU}])), \tag{3.52}$$

where $[\hat{\mathbf{O}}_{CNN}; \hat{\mathbf{O}}_{LSTM}; \hat{\mathbf{O}}_{GRU}] \in \mathbb{R}^{3\times T}$ denotes a concatenation of three subnetworks' ouputs. $f^{1\times 7}$ is a 1×7 convolution with same padding. $\hat{\mathbf{O}} \in \mathbb{R}^{1\times T}$ is the output of TA-SeriesNet. This idea is inspired by the CBAM [35] and HSAM to know each subnetwork's influence on forecasting results via learning their weight parameters.

3.5 Experiments of A-SeriesNet

This thesis uses five typical open time series datasets, including three economic data: S&P500 Index, Shanghai Composite Index, Tesla Stock Price and two temperature data: NewYork hourly temperature and Weather in Szeged as shown in Table 3.1 to evaluate the models. The attention-based SeriesNet is compared with the SeriesNet [27], the Augmented WaveNet [32], the SVR [42] and the GRU networks [33]. Each model is evaluated by 4 metrics: the root-mean-square error (RMSE), the mean absolute error (MAE), the coefficient of determination (R^2) and the computation time of specific epoch numbers. This thesis takes an average of ten times of training results as the final accuracy of each model.

Table 3.1: Time series dataset.

Time Series	Time Range	Train Data	Validation Data	Test Data
S&P500 Index	1950.01–2015.12	3297	320	320
Shanghai Composite Index	2004.01–2019.06	2430	280	280
Tesla Stock Price	2010.06–2017.03	1049	160	160
NewYork temperature	2016.01–2016.07	2430	320	320
Weather in Szeged	2006.04–2016.09	1700	240	240

This section uses A_SeriesNet and WaveNet instead of the attention-based SeriesNet and the augmented WaveNet for short. The experiments are executed on Windows 10 with 2.50 GHz Intel Core i7 and 8 GB memory and conducted on the python environment with Keras deep learning structure. The hyper-parameters of A_SeriesNet shown in Table 3.2, Tables 3.3 and 3.4 are slightly adjusted when it applies to different datasets. The reduction ratio of CBAM and HSAM shown in Tables 3.3 and 3.4 is one. The padding of depthwise convolution and pointwise convolution of DDSTCNs is causal and valid, respectively. In the case of three economic datasets, this thesis uses daily average stock price as the target time series (Input) by taking the average of daily high and low stock price. The part of the other time series in two economic datasets, such as the daily trading volume and the daily close stock price, is chosen as the conditions. For the two temperature datasets, the temperature is considered as the target time series (Input), the Dew point and the humidity are chosen as

the conditions. This thesis adopts the MAE as the loss function as below:

$$loss_{min} = \frac{1}{T}\sum_{t=1}^{T}|F_t - A_t|\,, \tag{3.53}$$

where F_t and A_t denotes the target value and predicted value at time t, respectively. The weights of all CNN layers of A_SeriesNet are initialized with a truncated normal distribution with zero mean and constant variance of 0.05. The GRU layers of A_SeriesNet are initialized with he_normal distribution. The Adam optimizer [55] is used with the learning rate 0.001 and β_1 of 0.9. The related layer numbers of SeriesNet and WaveNet are unified with A_SeriesNet as shown in Table 3.2. This thesis removed CBAM and HSAM and used DC-CNN and LSTM instead of DDSTCNs and GRU in Figure 3.2 as the conditional structure of SeriesNet. All the models except for SVR used the conditioning method for the experiments.

This thesis computes each layer's complexity for detecting our model's computational performance, as demonstrated in Tables 3.2–3.4. The shape of input time series and condition is respectively specified to $x \in \mathbb{R}^{1\times T}$ and $y \in \mathbb{R}^{1\times T}$ for easy calculating the complexity. The evaluation is only limited to the forward propagation of the computational process. The complexity of a standard 1D CNN layer is defined as below:

$$Complexity \sim O(M \cdot K \cdot C_{in} \cdot C_{out}), \tag{3.54}$$

where M is the width of the output feature map, K denotes the width of the kernel, C_{in} and C_{out} represents the channel input and channel output, respectively. We ignore the bias of all CNN layers and full connection layers for convenience to compute the complexity. The complexity of a 1D DDSTCNs layer is computable as follows:

$$Complexity \sim O(M \cdot K \cdot C_{in} + M \cdot C_{in} \cdot C_{out}). \tag{3.55}$$

Table 3.2: Hyper parameters and complexity of attention-based SeriesNet.

Type	Units/Filters	Size	Dilation Rate	Padding	Output	Complexity
Conv1D(Input)	1	30	1	causal	(50, 1)	1500
BN					(50, 1)	0
DDSTCNs	8	7	1	causal/valid	(50, 8)	750
Conv1D(Condition)	1	20	1	causal	(50, 1)	1000
BN					(50, 1)	0
DDSTCNs	8	4	1	causal/valid	(50, 8)	600
Add					(50, 8)	0
SeLU					(50, 8)	0
CBAM					(50, 8)	956
Conv1D	1	1	1	same	(50, 1)	400
Add					(50, 1)	0
BN					(50, 1)	0
DDSTCNs	8	7	2	causal/valid	(50, 8)	750
SeLU					(50, 8)	0
CBAM					(50, 8)	956
Conv1D	1	1	1	same	(50, 1)	400
Add					(50, 1)	0
		⋮				
BN					(50, 1)	0
DDSTCNs	8	7	16	causal/valid	(50, 8)	750
SeLU					(50, 8)	0
CBAM					(50, 8)	956
Conv1D	1	1	1	same	(50, 1)	400
Add(Skip-Connection)					(50, 1)	0
Conv1D	1	1	1	same	(50, 1)	50
Condition					(1, 20)	1000
GRU(Input, Condition)	20				(50, 20)	66,000
HSAM					(50, 20)	780
GRU	20				(50, 20)	123,000
FC	1				(50, 1)	20
Multiply					(50, 1)	0
ReLU					(50, 1)	0
Total						204,480

Table 3.3: Hyper parameters and complexity of HSAM.

Type	Units/Filters	Size	Dilation Rate	Padding	Output	Complexity
Lambda_Mean(GRU)					(50, 1)	0
FC	20				(50, 20)	20
ReLU					(50, 20)	0
FC	1				(50, 1)	20
Lambda_Max(GRU)					(50, 1)	0
FC	20				(50, 20)	20
ReLU					(50, 20)	0
FC	1				(50, 1)	20
Concatenate					(50, 2)	0
Conv1D	1	7	1	same	(50, 1)	700
Sigmoid					(50, 1)	0
Multiply(GRU, Sigmoid)					(50, 20)	0
Total						780

Table 3.4: Hyper parameters and complexity of CBAM.

Type	Units/Filters	Size	Dilation Rate	Padding	Output	Complexity
GlobalAvgPooling1D(SeLU)					(1, 8)	0
FC	8				(1, 8)	64
ReLU					(1, 8)	0
FC	8				(1, 8)	64
GlobalMaxPooling1D(SeLU)					(1, 8)	0
FC	8				(1, 8)	64
ReLU					(1, 8)	0
FC	8				(1, 8)	64
Add					(1, 8)	0
Sigmoid					(1, 8)	0
Multiply1(SeLU, Sigmoid)					(50, 8)	0
Lambda_Mean(Multiply1)					(50, 1)	0
Lambda_Max(Multiply1)					(50, 1)	0
Concatenate					(50, 2)	0
Conv1D	1	7	1	same	(50, 1)	700
Sigmoid					(50, 1)	0
Multiply2(Multiply1, Sigmoid)					(50, 8)	0
Total						956

On the other hand, LSTM is local in space and time, which means that the input length does not affect the storage requirements of the network and for each time step, the time complexity per weight is $O(1)$. Therefore, the overall complexity of an LSTM per time step is equal to $O(w)$, where w is the number of weights. The complexity of a standard LSTM layer per time step is calculated as:

$$Complexity \sim O(4 \cdot (I \cdot H + H^2 + H)), \tag{3.56}$$

where I denotes the dimension of input data, H represents the hidden unit numbers. The Complexity of a standard GRU layer per time step is simpler than LSTM, which is given as:

$$Complexity \sim O(3 \cdot (I \cdot H + H^2 + H)). \tag{3.57}$$

The overall complexity of our model is the sum of the complexity of all layers.

Table 3.5 shows the experimental results when the forecast sliding window representing the future time span is 1. GRU^2_{20} denotes using 2 layers of GRU cell and each layer contains 20 neurons. The A_SeriesNet has the best performance on both non-linear and non-stationary economic datasets and relatively stationary time series temperature dataset compared with the other models. The lower RMSE, MAE and higher R^2 close to 1 means better model fitting. This thesis performs the models except for SVR for 64 epochs with 64 mini-batch size one time. This epoch number allows the models to achieve a satisfactory convergence on five datasets.

Table 3.5: The result of accuracy comparison.

Time Series	*A_SeriesNet*			*SeriesNet*			*WaveNet*			GRU^2_{20}			*SVR*		
	RMSE	MAE	R^2	RMSE	MAE	R^2	RMSE	MAE	R^2	RMSE	MAE	R^2	RMSE	MAE	R^2
S&P500 Index	**8.90**	**7.17**	**0.98**	10.08	8.11	0.97	11.13	8.73	0.97	10.57	8.39	0.97	16.16	12.61	0.96
Shanghai Composite Index	**56.69**	**36.49**	**0.98**	71.96	55.50	0.97	80.52	60.10	0.97	79.29	50.17	0.97	82.25	63.19	0.97
Tesla Stock Price	**4.56**	**3.36**	**0.97**	4.82	3.68	0.96	5.50	4.36	0.95	5.59	4.38	0.95	4.74	3.36	0.96
NewYork temperature	**1.63**	**1.20**	**0.97**	1.68	1.22	0.97	1.76	1.25	0.96	1.72	1.25	0.97	1.79	1.23	0.96
Weather in Szeged	**1.22**	**0.71**	**0.96**	1.29	0.79	0.96	1.44	0.90	0.95	1.42	0.88	0.95	1.41	0.83	0.95

Table 3.6 demonstrates the average computation time (in seconds) of the models for one-time training. The computation time of A_SeriesNet is in rank 3, which is faster than SeriesNet and slower than GRU^2_{50}. The SVR takes longer training time to obtain the results close to the other models.

Table 3.6: The result of performance comparison.

Time Series	$A\_SeriesNet$	$SeriesNet$	$WaveNet$	GRU^2_{20}	SVR
S&P500 Index	**100.80**	103.62	17.99	74.37	273.49
Shanghai Composite Index	**92.72**	94.42	18.73	76.10	237.40
Tesla Stock Price	**61.90**	64.69	36.36	41.37	124.97
NewYork temperature	**99.78**	101.38	17.80	74.73	107.08
Weather in Szeged	**89.24**	96.56	17.54	62.93	115.96

Table 3.7 shows the results of GRU combined with HSAM (HSAM_GRU) compared with GRU. This paper adopts GRU^2_{20} with 2 layers of GRU cell and each layer contains 20 neurons and GRU^4_{20} with 4 layers of GRU cell and each layer contains 20 neurons for the experiments. The results show that the different layers of HSAM_GRU are superior to related GRU networks. When the number of layers increased, the accuracy of GRU for 3 datasets decreases. HSAM can keep the accuracy of deep GRU networks. The computation time of HSAM_GRU is close to GRU as demonstrated in Table 3.8.

Table 3.7: The accuracy comparison of HSAM_GRU and GRU.

Time Series	$HSAM\_GRU^2_{20}$			GRU^2_{20}			$HSAM\_GRU^4_{20}$			GRU^4_{20}		
	RMSE	MAE	R^2	RMSE	MAE	R^2	RMSE	MAE	R^2	RMSE	MAE	R^2
S&P500 Index	**9.51**	**7.77**	**0.98**	10.57	8.39	0.97	**10.18**	**8.01**	**0.97**	12.23	9.92	0.96
Shanghai Composite Index	**78.44**	**48.49**	**0.97**	79.29	50.17	0.97	**80.46**	**54.74**	**0.97**	92.83	66.70	0.96
Tesla Stock Price	**5.02**	**3.89**	**0.96**	5.59	4.38	0.95	**6.24**	**4.63**	**0.94**	6.29	5.00	0.94
NewYork temperature	**1.68**	**1.23**	**0.97**	1.72	1.25	0.97	**1.73**	**1.27**	**0.97**	1.76	1.28	0.97
Weather in Szeged	**1.33**	**0.80**	**0.96**	1.42	0.88	0.95	**1.41**	**0.82**	**0.95**	1.56	1.00	0.94

Table 3.8: The performance comparison of HSAM_GRU and GRU.

Time Series	$HSAM\_GRU^2_{20}$	GRU^2_{20}	$HSAM\_GRU^4_{20}$	GRU^4_{20}
S&P500 Index	**81.94**	74.37	**167.42**	145.09
Shanghai Composite Index	**82.87**	76.10	**172.13**	141.04
Tesla Stock Price	**39.44**	36.36	**86.82**	76.52
NewYork temperature	**81.21**	74.73	**170.77**	147.92
Weather in Szeged	**66.20**	62.93	**125.95**	117.07

Tables 3.9–3.11 show the hyper parameters and complexity of SeriesNet and WaveNet in our experiments. The shape of input time series and condition in the tables is also appointed to $x \in \mathbb{R}^{1\times T}$ and $y \in \mathbb{R}^{1\times T}$, respectively. We also

ignore the bias of all CNN layers and full connection layers for computing the overall complexity of these models. The structure of GRU_{20}^{4} is similar to GRU_{20}^{2} in Tables 3.10 and 3.12 gives the complexity comparison results of deep learning models. The complexity of our model is between GRU_{20}^{2} and SeriesNet.

Table 3.9: Hyper parameters and complexity of augmented WaveNet.

Type	**Units/Filters**	**Size**	**Dilation Rate**	**Padding**	**Output**	**Complexity**
Conv1D(Input)	8	7	1	causal	(50, 8)	2800
ReLU					(50, 8)	0
Conv1D(Condition)	8	7	1	causal	(50, 8)	2800
ReLU					(50, 8)	0
Add					(50, 8)	0
Conv1D	8	7	2	causal	(50, 8)	22,400
ReLU					(50, 8)	0
Add					(50, 8)	0
		⋮				
Conv1D	8	7	16	causal	(50, 8)	22,400
ReLU					(50, 8)	0
Add(Skip-Connection)					(50, 8)	0
Conv1D	1	1	1	same	(50, 1)	400

Table 3.10: Hyper parameters and complexity of GRU_{20}^{2}.

Type	**Units/Filters**	**Size**	**Dilation Rate**	**Padding**	**Output**	**Complexity**
Condition					(1, 20)	1000
GRU(Input, Condition)	20				(50, 20)	66,000
GRU	20				(50, 20)	123,000
FC	1				(50, 1)	20
Total						190,020

Table 3.11: Hyper parameters and complexity of SeriesNet.

Type	**Units/Filters**	**Size**	**Dilation Rate**	**Padding**	**Output**	**Complexity**
Conv1D(Input)	1	20	1	causal	(50, 1)	1000
BN					(50, 1)	0
Conv1D	8	7	1	causal	(50, 8)	2800
Conv1D(Condition)	1	20	1	causal	(50, 1)	1000
BN					(50, 1)	0
Conv1D	8	4	1	causal	(50, 8)	1600
Add					(50, 8)	0
Conv1D	1	1	1	same	(50, 1)	400
Add					(50, 1)	0
BN					(50, 1)	0
Conv1D	8	7	2	causal	(50, 8)	2800
Conv1D	1	1	1	same	(50, 1)	400
Add					(50, 1)	0
		⋮				
BN					(50, 1)	0
Conv1D	8	7	16	causal	(50, 8)	2800
Conv1D	1	1	1	same	(50, 1)	400
Add(Skip-Connection)					(50, 1)	0
Conv1D	1	1	1	same	(50, 1)	50
Condition					(2, 20)	2000
LSTM(Input, Condition)	20				(50, 20)	88,000
LSTM	20				(50, 20)	164,000
FC	1				(50, 1)	20
Multiply					(50, 1)	0
ReLU					(50, 1)	0
Total						273,670

Table 3.12: The result of complexity comparison.

	$A\_SeriesNet$	$SeriesNet$	$WaveNet$	GRU^2_{20}	GRU^4_{20}
Complexity	**204,480**	273,670	95,600	190,020	436,020

3.6 Experiments of TA-SeriesNet

3.6.1 Training procedure and evaluation metrics

This thesis first uses the mean absolute error as the loss function: MAE $= \frac{1}{T}\sum_{t=1}^{T}\left|y_t - \hat{O}_t\right|$, where y_t and $\hat{O}_t$ represents the target value and predicted value at time t, respectively. We use the Adam optimizer [55] with learning rate 0.001 for training. The size of minibath is 64. The model accuracy are evaluated by 2 metrics: the root-mean-square error: RMSE $= \sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(y_t - \hat{O}_t\right)^2}$, and the coefficient of determination: $\mathrm{R}^2 = 1 - \frac{\sum_{t=1}^{T}\left(y_t - \hat{O}_t\right)^2}{\sum_{t=1}^{T}(y_t - \bar{y})^2}$. Oppositely, we apply RMSE as the loss function and adopt MAE and R^2 as the evaluation metrics to verify the first training results. We use tensorflow framework to generate and train the models.

3.6.2 Model parameter adjustment

The hyper-parameters of DARLM-CNN are illustrated in table. 3.13 and 3.14, where its encoder and decoder residual layer number $J = L = 5$ and time step $T = 10$. 'Type', 'Filters', 'Size', 'Dilation', 'Padding' and 'Output' represents layers of DARLM-CNN, filter number, kernel size, dilation rate, padding way and output shape of each layer, respectively.

Table. 3.15 and 3.16 introduce the hyper-parameters of TA-LSTM with the same hidden state size of encoder and decoder $m = p = 64$. We use a time series dataset with feature dimension six and unfold the structure of TA-LSTM by $T = 10$ time steps to generate these two tables. The structure of TA-GRU is similar to TA-LSTM. In these 4 tables, there are some layers followed by a bracket with some parameters in it, which denotes the input of that layer. We use the new concatenation method to integrate the above tables to generate the TA-SeriesNet model and compare our proposed models with the A-SeriesNet, the DA-RNN [38], the augmented ARLM-CNN and the augmented WaveNet [32].

Table 3.13: Hyper-parameters of ARLM-CNN (Encoder of DARLM-CNN).

Type	**Filters**	**Size**	**Dilation**	**Padding**	**Output**
Conv1D(**x**)	1	20	1	causal	(10, 1)
BN					(10, 1)
DDSTCNs	8	7	1	causal/valid	(10, 8)
SeLU					(10, 8)
CBAM					(10, 8)
Conv1D	1	1	1	same	(10, 1)
Add					(10, 1)
BN					(10, 1)
DDSTCNs	8	7	2	causal/valid	(10, 8)
SeLU					(10, 8)
CBAM					(10, 8)
Conv1D	1	1	1	same	(10, 1)
Add					(10, 1)
		⋮			
BN					(10, 1)
DDSTCNs	8	7	16	causal/valid	(10, 8)
SeLU					(10, 8)
CBAM					(10, 8)
Conv1D	1	1	1	same	(10, 1)
Add(Skip-connections)					(10, 1)
Conv1D	1	1	1	same	(10, 1)

Table 3.14: Hyper-parameters of augmented ARLM-CNN (Decoder of DARLM-CNN).

Type	**Filters**	**Size**	**Dilation**	**Padding**	**Output**
BN(**c**)					(10, 1)
DDSTCNs	8	4	1	causal/valid	(10, 8)
Conv1D(**y**)	1	30	1	causal	(10, 1)
BN					(10, 1)
DDSTCNs	8	7	1	causal/valid	(10, 8)
Add					(10, 8)
SeLU					(10, 8)
CBAM					(10, 8)
Conv1D	1	1	1	same	(10, 1)
Add					(10, 1)
BN					(10, 1)
DDSTCNs	8	7	2	causal/valid	(10, 8)
SeLU					(10, 8)
CBAM					(10, 8)
Conv1D	1	1	1	same	(10, 1)
Add					(10, 1)
		⋮			
BN					(10, 1)
DDSTCNs	8	7	16	causal/valid	(10, 8)
SeLU					(10, 8)
CBAM					(10, 8)
Conv1D	1	1	1	same	(10, 1)
Add(Skip-connections)					(10, 1)
Conv1D	1	1	1	same	(10, 1)

Table 3.15: Hyper-parameters of TA-LSTM (Encoder with input attention and HSAM).

Type	**Units**	**Output**
RepeatVector(5, $\mathbf{s}_0$)		(5, 64)
RepeatVector(5, $\mathbf{h}_0$)		(5, 64)
Permute($\mathbf{x}$)		(5, 10)
Concatenate($\mathbf{s}_0$, $\mathbf{h}_0$, $\mathbf{x}$)		(5, 138)
Full-connection	1	(5, 1)
Permute		(1, 5)
Softmax		(1, 5)
Multiply(α_1, $\mathbf{x}_1$)		(1, 5)
Encoder-LSTM$_1$	64	(1, 64)
$\vdots$		
RepeatVector(5, $\mathbf{s}_9$)		(5, 64)
RepeatVector(5, $\mathbf{h}_9$)		(5, 64)
Permute($\mathbf{x}$)		(5, 10)
Concatenate($\mathbf{s}_9$, $\mathbf{h}_9$, $\mathbf{x}$)		(5, 138)
Full-connection	1	(5, 1)
Permute		(1, 5)
Softmax		(1, 5)
Multiply(α_{10}, $\mathbf{x}_{10}$)		(1, 5)
Encoder-LSTM$_{10}$	64	(1, 64)
HSAM		(10, 64)

Table 3.16: Hyper-parameters of TA-LSTM (Decoder with temperal attention).

Type	**Units**	**Output**
RepeatVector(10, $\mathbf{s}_0^{'}$)		(10, 64)
RepeatVector(10, $\mathbf{d}_0$)		(10, 64)
Concatenate($\mathbf{s}_0^{'}$, $\mathbf{d}_0$, $\mathbf{h}^{'}$)		(10, 192)
Full-connection	128	(10, 128)
Tanh		(10, 128)
Full-connection	1	(10, 1)
Permute		(1, 10)
Softmax		(1, 10)
Permute		(10, 1)
Multiply(β_1, $\mathbf{h}^{'}$)		(10, 64)
Sum		(1, 64)
Concatenate($\mathbf{c}_1$, y_1)		(1, 65)
Decoder-LSTM$_1$	64	(1, 64)
Concatenate($\mathbf{d}_1$, $\mathbf{c}_1$)		(1, 128)
Full-connection	1	(1, 1)
$\vdots$		
RepeatVector(10, $\mathbf{s}_9^{'}$)		(10, 64)
RepeatVector(10, $\mathbf{d}_9$)		(10, 64)
Concatenate($\mathbf{s}_9^{'}$, $\mathbf{d}_9$, $\mathbf{h}^{'}$)		(10, 192)
Full-connection	128	(10, 128)
Tanh		(10, 128)
Full-connection	1	(10, 1)
Permute		(1, 10)
Softmax		(1, 10)
Permute		(10, 1)
Multiply(β_{10}, $\mathbf{h}^{'}$)		(10, 64)
Sum		(1, 64)
Concatenate($\mathbf{c}_{10}$, y_{10})		(1, 65)
Decoder-LSTM$_{10}$	64	(1, 64)
Concatenate($\mathbf{d}_{10}$, $\mathbf{c}_{10}$)		(1, 128)
Full-connection	1	(1, 1)

It's necessary to determine five parameters to know our proposed models' performance. Respectively are the time step T, the encoder's hidden state size m and decoder's hidden state size p for TA-LSTM, TA-GRU and TA-SeriesNet, the encoder's residual layer number L and decoder's residual layer number J for DARLM-CNN and TA-SeriesNet. First of all, we set up the DARLM-CNN by $J = L = \{5, 6, 7, 8, 9, 10\}$ for simplicity and choose the best performed layer number $J = L = 10$ over the validation set. Then we ensembly 3 subnetworks to generate the TA-SeriesNet. Since the DA-RNN [38] measured $T = \{5, 10, 15, 20, 25\}$, $m = p = \{16, 32, 64, 128, 256\}$ and verified $T = 10, m = p = 64$ achieved the best results in their experiments. We use the grid search over $T = \{10, 20, 30, 40, 50\}$ and $m = p = \{16, 32, 64, 128, 256\}$ to determine the parameters that the TA-LSTM, TA-GRU and TA-SeriesNet can achieve the best performance. The two TA-RNN subnetworks respectively attained the best performance when $T = 50$, $m = p = 64$ over the validation set. Coincidentally, the TA-SeriesNet also achieves optimum performance by the same parameters. We also choose the best parameters via the same method for the other models. This thesis trains each model ten times and records the average metrics for model comparison. The experiments are implemented on raw datasets without preprocessing.

3.6.3 Experimental results

3.6.3.1 Forecasting accuracy comparison

Table. 3.17 summarizes all models' prediction precision with time step $T = 50$, which uses MAE as the loss function and RMSE and R^2 as the evalution indicators for the testing set of four datasets. Table. 3.18 shows the inverse training results, which adopts RMSE as the loss function and uses MAE and R^2 as the evaluation metrics to verify the results of table. 3.17. We split the models into two types for both tables. The non-encoder-decoder type contains the first three models, which have the corresponding residual layer number written in the followed brackets. The two values behind the A-SeriesNet is the residual layer number of its ARLM-CNN subnetwork and the hidden state size of its HSAM-based GRU subnetwork. The others are the encoder-decoder type models with the same residual layer number or same hidden state size for their encoder and

Table 3.17: MAE loss function-based model accuracy comparison for the four datasets.

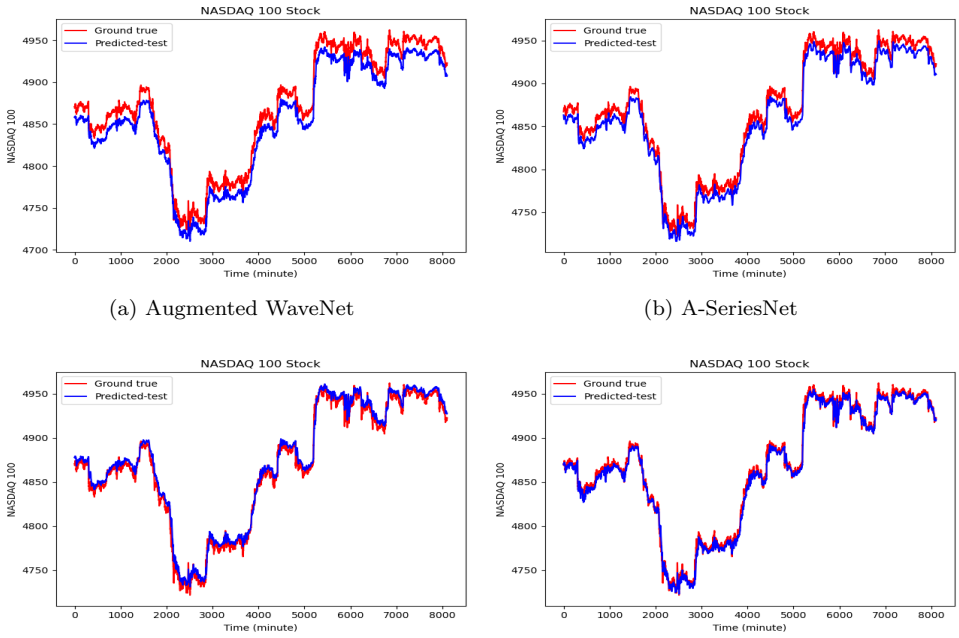
	NewYork temperature		**S&P500 Index**		**SML 2010**		**NASDAQ 100 Stock**	
Models	**RMSE**	R^2	**RMSE**	R^2	**RMSE**	R^2	**RMSE**	R^2
Augmented WaveNet (10)	2.14	0.96	15.67	0.97	0.38	0.97	13.65	0.95
Augmented ARLM-CNN (10)	1.99	0.96	15.02	0.97	0.34	0.97	11.46	0.96
A-SeriesNet (10, 64)	1.77	0.96	14.15	0.97	0.32	0.97	9.89	0.96
DA-LSTM (64)	1.63	0.96	13.31	0.98	0.27	0.98	7.43	0.97
DARLM-CNN (10)	1.83	0.96	14.79	0.97	0.29	0.98	7.64	0.97
TA-LSTM (64)	1.41	0.97	12.75	0.98	0.26	0.98	6.93	0.97
TA-GRU (64)	1.54	0.96	12.07	0.98	0.22	0.98	6.75	0.97
TA-SeriesNet (10, 64, 3)	**1.31**	**0.97**	**11.29**	**0.98**	**0.21**	**0.98**	**5.57**	**0.98**

Table 3.18: RMSE loss function-based model accuracy comparison for the four datasets.

	NewYork temperature		**S&P500 Index**		**SML 2010**		**NASDAQ 100 Stock**	
Models	**MAE**	R^2	**MAE**	R^2	**MAE**	R^2	**MAE**	R^2
Augmented WaveNet (10)	1.97	0.96	14.88	0.97	0.29	0.97	12.48	0.95
Augmented ARLM-CNN (10)	1.86	0.96	14.27	0.97	0.26	0.97	10.98	0.96
A-SeriesNet (10, 64)	1.64	0.96	13.11	0.97	0.22	0.97	8.15	0.96
DA-LSTM (64)	1.51	0.96	12.84	0.98	0.17	0.98	6.10	0.97
DARLM-CNN (10)	1.75	0.96	13.67	0.97	0.18	0.98	6.44	0.97
TA-LSTM (64)	1.29	0.97	11.25	0.98	0.15	0.98	5.29	0.97
TA-GRU (64)	1.37	0.97	11.32	0.98	0.14	0.98	5.79	0.97
TA-SeriesNet (10, 64, 3)	**1.22**	**0.97**	**10.14**	**0.98**	**0.12**	**0.98**	**4.45**	**0.98**

decoder. The three values of TA-SeriesNet indicate the residual layer number, the hidden state size of its related subnetwork and its subnetwork number, respectively. We apply the LSTM [30] for the encoder and decoder of DA-RNN [38] and replace the DA-RNN [38] with the DA-LSTM in the experiment section.

In table. 3.17, all the models perform good forecasting results for the first three datasets with feature dimensions less than 20. The forecasting accuracy of non-encoder-decoder type models declined for the fourth dataset with high feature dimension. The encoder-decoder type models maintain a stable forecasting accuracy than the non-encoder-decoder ones. Our proposed TA-RNNs and



(a) Augmented WaveNet

(b) A-SeriesNet

(c) DA-RNN

(d) TA-SeriesNet

Figure 3.12: Model comparison for the testing set of NASDAQ 100 Stock dataset with 81 feature dimensions.

TA-SeriesNet are outperform to the other models for the four datasets. The precision of DARLM-CNN falls in between the A-SeriesNet and the DA-LSTM for the four datasets. Fig. 3.12 demonstrates the visual comparison of TA-SeriesNet with other three state-of-art models over the testing set of NASDAQ Stock 100 dataset. The TA-SeriesNet fits the ground truth better than the others. The results of table. 3.18 have the same tendency as table. 3.17 except for some nuances.

3.6.3.2 Model sensitivity analysis

The sensitivity detection of each model to the feature dimension variation is our second experiment. We separate the NASDAQ 100 Stock dataset with 81 feature dimensions into 5 subdatasets according to different feature dimension numbers. We successively select the top 16, 32, 48, 64, and 80 feature dimensions from the multi-condition series and respectively combine them with the target time series, the index of NASDAQ 100, as each subdataset. Similar to the first experiment, we respectively train subdatasets by loss function MAE and RMSE

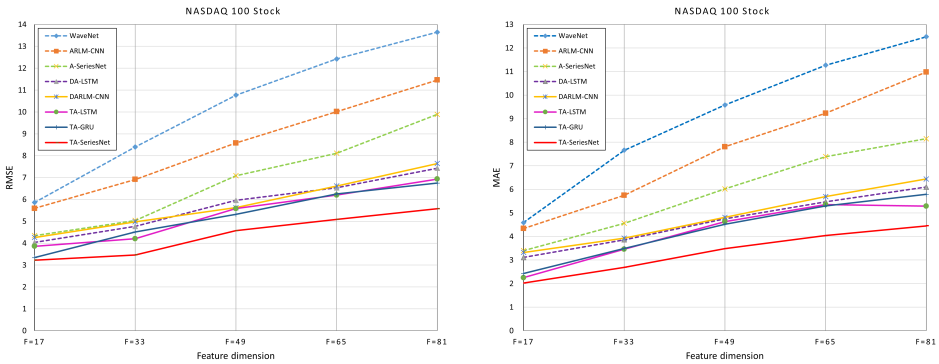
Table 3.19: MAE loss function-based model sensitivity evaluation for different feature dimensions.

	F=17		**F=33**		**F=49**		**F=65**	
Models	**RMSE**	R^2	**RMSE**	R^2	**RMSE**	R^2	**RMSE**	R^2
Augmented WaveNet (10)	5.86	0.98	8.40	0.97	10.77	0.96	12.42	0.95
Augmented ARLM-CNN (10)	5.59	0.98	6.91	0.97	8.58	0.97	10.01	0.96
A-SeriesNet (10, 64)	4.35	0.98	5.04	0.98	7.09	0.97	8.11	0.97
DA-LSTM (64)	4.04	0.98	4.77	0.98	5.96	0.98	6.53	0.97
DARLM-CNN (10)	4.26	0.98	4.98	0.98	5.62	0.98	6.61	0.97
TA-LSTM (64)	3.86	0.98	4.21	0.98	5.58	0.98	6.20	0.97
TA-GRU (64)	3.34	0.98	4.52	0.98	5.32	0.98	6.25	0.97
TA-SeriesNet (10, 64, 3)	**3.32**	**0.98**	**3.46**	**0.98**	**4.57**	**0.98**	**5.08**	**0.98**

Table 3.20: RMSE loss function-based model sensitivity evaluation for different feature dimensions.

	F=17		**F=33**		**F=49**		**F=65**	
Models	**MAE**	R^2	**MAE**	R^2	**MAE**	R^2	**MAE**	R^2
Augmented WaveNet (10)	4.59	0.98	7.66	0.97	9.58	0.96	11.27	0.95
Augmented ARLM-CNN (10)	4.34	0.98	5.75	0.97	7.81	0.97	9.23	0.96
A-SeriesNet (10, 64)	3.40	0.98	4.56	0.98	6.02	0.97	7.39	0.97
DA-LSTM (64)	3.11	0.98	3.86	0.98	4.75	0.98	5.47	0.97
DARLM-CNN (10)	3.32	0.98	3.93	0.98	4.81	0.98	5.69	0.97
TA-LSTM (64)	2.25	0.98	3.46	0.98	4.63	0.98	5.35	0.97
TA-GRU (64)	2.43	0.98	3.49	0.98	4.52	0.98	5.30	0.97
TA-SeriesNet (10, 64, 3)	**2.02**	**0.98**	**2.69**	**0.98**	**3.48**	**0.98**	**4.04**	**0.98**

and summarize the results in table. 3.19 and 3.20. The accuracy of the last subdataset with 81 feature dimensions is the results of NASDAQ Stock 100 dataset shown in table. 3.17 and 3.18. We detect the forecasting accuracy variation of each model from subdataset one to five. Table. 3.19 demonstrates that the accuracy of all models decreased with the feature dimension (F) increasing. The decline of non-encoder-decoder type models sharper than the encoder-decoder type ones. The same tendency also can be observed in table. 3.20. Fig. 3.13 visually shows each model's RMSE and MAE evalution indicator fluctuation, which is respectively trained by loss function MAE and RMSE for the 5 subdatasets. The sensitivity of DARLM-CNN is neck and neck with DA-LSTM. The



(a) MAE loss function-based RMSE indicator variation tendency.

(b) RMSE loss function-based MAE indicator variation tendency.

Figure 3.13: Evaluation metrics fluctuation of each model for different feature dimensions.

TA-LSTM, TA-GRU and TA-SeriesNet are superior to others.

3.6.3.3 Concatenation method evaluation

The third experiment is to verify the validity of the concatenation method. In this experiment, we increase the subnetwork number of TA-SeriesNet and evaluate its performance for the four datasets. We also train the generated models by loss function MAE first as shown in table. 3.21. The TA-SeriesNet (10, 64, 4) with four subnetworks consists of a TA-LSTM, a TA-GRU and two DARLM-CNN subnetworks. The TA-SeriesNet (10, 64, 5) with five subnetworks contains a TA-GRU, two TA-LSTM and two DARLM-CNN subnetworks. The TA-SeriesNet (10, 64, 6) with six subnetworks includes TA-LSTM, TA-GRU and DARLM-CNN, two for each. The prediction accuracy of them remains stable when the subnetwork number increased. The results of table. 3.22 are trained by loss function RMSE, where its results are similar to table. 3.21. All the models in these two tables maintain an excellent performance compare to the other models in table. 3.17 and 3.18 for both low and high feature dimensional datasets.

Table 3.21: MAE loss function-based concatenation method evaluation for TA-SeriesNet.

	NewYork temperature		**S&P500 Index**		**SML 2010**		**NASDAQ 100 Stock**	
Models	**RMSE**	R^2	**RMSE**	R^2	**RMSE**	R^2	**RMSE**	R^2
TA-SeriesNet (10, 64, 4)	**1.47**	**0.97**	**11.48**	**0.98**	**0.23**	**0.98**	**5.69**	**0.98**
TA-SeriesNet (10, 64, 5)	1.51	0.97	12.14	0.98	0.25	0.98	6.16	0.97
TA-SeriesNet (10, 64, 6)	1.56	0.96	12.80	0.98	0.28	0.98	7.01	0.97

Table 3.22: RMSE loss function-based concatenation method evaluation for TA-SeriesNet.

	NewYork temperature		**S&P500 Index**		**SML 2010**		**NASDAQ 100 Stock**	
Models	**MAE**	R^2	**MAE**	R^2	**MAE**	R^2	**MAE**	R^2
TA-SeriesNet (10, 64, 4)	**1.24**	**0.97**	**10.29**	**0.98**	**0.14**	**0.98**	**4.51**	**0.98**
TA-SeriesNet (10, 64, 5)	1.32	0.97	11.25	0.98	0.16	0.98	4.58	0.98
TA-SeriesNet (10, 64, 6)	1.35	0.97	11.60	0.98	0.17	0.98	5.74	0.97

Chapter 4

Conclusion

This thesis constructs the supermarket competition analysis models and makes loyal customer classification for a supermarket chain. In the experiments, this study estimates the RFM, RFM+ and RFM++ type model by accuracy and logistic regression coefficient analysis. All three type models can make loyal customer classification adequately. The RFM++ type model is superior to the other two type models from the viewpoint of accuracy and analysis diversity. The supermarket managers can grasp the influence degree of competitive supermarkets and understand the behavior of loyal customers. In the future, the presented models will implement on sensitivity analysis of neural networks for supermarket competition analysis and loyal customer classification.

Furthermore, this thesis proposed a deep learning neural network structure named attention-based SeriesNet, which desires to predict the future value of time series. The attention-based SeriesNet applies DDSTCNs and GRU instead of DC-CNN and LSTM in SerieNet to accelerate the training. Furthermore, this model adopts CBAM attention on residual learning module and proposed HSAM attention on GRU networks to better extract the potential features from the input time series. We succeeded in improving SeriesNet since our model's accuracy, and complexity is superior to the SeriesNet. The experiment results also show that attention-based SeriesNet has higher forecasting accuracy than other models. This thesis only explored the performance of the SeriesNet models on the economic and temperature datasets. Further analysis of different types of datasets is required to examine the capability of attention-based SeriesNet to forecast from different data distributions for varying forecast horizons. This

thesis didn't evaluate the performance of hidden state attention mechanisms on recurrent neural networks with deep structure. The only two or four layers GRU can not adequately describe its performance. It was also found that the forecasts were very sensitive to layer weight initialization, receptive field and training duration. The parameter tuning is necessary for different datasets.

Finally, this thesis presents a novel hybrid neural network that included two types of attention-based encoder-decoder architectures. Most of the encoder-decoder framework is based on the recurrent neural network at present. This thesis first considered the attention residual learning module-based encoder-decoder model and proved this architecture useful in predicting time series with high feature dimensions. Furthermore, this thesis testified that the performance of DA-RNN is improvable by adding the hidden state attention module between encoder and decoder architecture. The concatenation method is a successful idea to parallel connect different neural network architectures without losing their performance. The concatenation of subnetworks can learn a related weight for each subnetwork to reduce the overall output's dependence on low forecasting accuracy subnetwork. This thesis only measured the forecasting accuracy of the new proposals. Since the concatenation method realized the parallel connection of multiple neural network architectures, we can parallel train them for computational performance improvement. In the future, we'll improve the TA-SeriesNet from this perspective.

Bibliography

[1] M. Khajvand, K. Zolfaghar, S. Ashoori, S. Alizadeh, "Estimating customer lifetime value based on RFM analysis of customer purchase behavior: case study," Procedia Computer Science 3, pp. 57-63, 2011.

[2] A. M. Hughes, Strategic database marketing, Chicago: Probus Publishing Company, 1994.

[3] H. C. Chang, H. P. Tsai, "Group RFM analysis as a novel framework to discover better customer consumption behavior," Expert Systems with Applications 38, pp. 14499-14513, 2011.

[4] J. Wu, Z. Lin, "Research on customer segmentation model by clustering," In Proceedings of the 7th ACM ICEC international conference on electronic commerce, 2005.

[5] T. Tanaka, T. Hamaguchi, T. Saigo, K. Tsuda, "Classifying and Understanding Prospective Customers via Heterogeneity of Supermarket Stores," International Conference on Knowledge Based and Intelligent Information and Engineering Systems, pp. 956-964, 2017.

[6] D. W. Hosmer, S. Lemeshow, Applied Logistic Regression, 2nd ed. John Wiley & Sons, Inc, 2000.

[7] T. Tjur, "Coefficients of determination in logistic regression models," American Statistician: pp. 366-372, 2009.

[8] D. A. Freedman, Statistical Models: Theory and Practice, Cambridge University Press, pp. 128, 2009.

[9] F. E. Harrell, Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis. New York: Springer, 2010.

[10] J. A. Morris, M. J. Gardner, "Calculating confidence intervals for relative risks (odds ratios) and standardised ratios and rates," British Medical Journal, 1988.

[11] D. E. Farrar, R. R. Glauber, "Multicollinearity in Regression Analysis: The Problem Revisited," Review of Economics and Statistics, 49, issue 1, pp. 92-107, 1967.

[12] J. Correia, RFM-analysis, GitHub repository, 2016. [Online]. Available: https://github.com/joaolcorreia/RFM-analysis

[13] D. L. Huff, "Defining and Estimating a Trade Area," Journal of Marketing, vol. 28, pp. 34-38, 1964.

[14] W. J. Reilly, The law of retail gravitation, New York: Knickerbocker Press, 1931.

[15] M. Nakanishi, L. G. Cooper, "Parameter estimation for a multiplicative competitive interaction model-least squares approach," Journal of Marketing Research, 11, pp. 303-311, 1974.

[16] D. B. Segal, "Retail Trade Area Analysis: Concepts and New Approaches," The Journal of Database Marketing, vol. 6, no. 3, pp. 267-277, 1999.

[17] K. Chen, Y. H. Hu, Y. C. Hsieh, "Predicting customer churn from valuable B2B customers in the logistics industry: a case study," Information Systems and e-Business Management, vol. 13, no. 3, pp. 475-494, 2015.

[18] H. Abdi, L. J. Williams, "Principal component analysis," Wiley Interdisciplinary Reviews: Computational Statistics, vol. 2, no. 4, pp. 433-459, 2010.

[19] D. Freedman, R. Pisani, R. Purves, Statistics: Fourth International Student Edition. W.W. Norton & Company, 2007.

[20] J. Fan, I. Gijbels, Local Polynomial Modelling and Its Applications: Monographs on Statistics and Applied Probability, Chapman & Hall/CRC, 1996.

[21] S. Zenker, T. Gollan, N. V. Quaquebeke, "Using Polynomial Regression Analysis and Response Surface Methodology to Make a Stronger Case for Value Congruence in Place Marketing," Psychology and Marketing, vol. 31, issue 3, pp. 184-202, 2014.

[22] Ruth M. W. Yeung, Wallace M. S. Yee, "Logistic Regression: An advancement of predicting consumer purchase propensity," The Marketing Review, vol. 11, no. 1, 2011.

[23] C. Constantin, "Using the Logistic Regression model in supporting decisions of establishing marketing strategies," Bulletin of the Transilvania University of Braşov Series V: Economic Sciences, vol. 8, issue 57, no. 2, 2015.

[24] H. Anton, Elementary Linear Algebra, 7th ed. John Wiley & Sons, pp. 170-171, 1994.

[25] C. Cui, J. Wang, Y. Pu, J. Ma, G. Chen, "GIS-based method of delimitating trade area for retail chains," International Journal of Geographical Information Science, vol. 26, no. 10, pp. 1863-1879, 2012.

[26] A. I. Marqués, V. García, J. S. Sánchez, "On the suitability of resampling techniques for the class imbalance problem in credit scoring," Journal of the Operational Research Society, vol. 64, no. 7, pp. 1060-1070, 2013.

[27] Z. Shen, Y. Zhang, J. Lu, J. Xu, G. Xiao, "SeriesNet: A Generative Time Series Forecasting Model," In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, pp. 1–8, 8–13 July 2018.

[28] A.Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O.Vinyals, A. Graves, N. Kalchbrenner, A. Senior, K. Kavukcuoglu, "Wavenet: A generative model for raw audio," 2016, arXiv:1609.03499.

[29] A.Krizhevsky, I. Sutskever, G.E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM 2017, 60, pp. 84–90.

[30] S. Hochreiter, J. Schmidhuber, "Long short-term memory," Neural Comput. 1997, Neural Comput. 9, 8 (November 15, 1997), pp. 1735–1780. DOI:10.1162/neco.1997.9.8.1735.

[31] A. Sherstinsky, "Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network," 2020, arXiv:1808.03314.

[32] A. Borovykh, S. M. Bohte, C. W. Oosterlee, "Conditional time series forecasting with convolutional neural networks," 2017, pp. 729–730, arXiv:1703.04691v5.

[33] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, arXiv:1412.3555v1.

[34] S. Ioffe, C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," Int. Conf. Mach. Learn. 2015, 37, pp. 448–456, .

[35] S. Woo, J. Park, J. Y. Lee, I. S. Kweon, "CBAM: Convolutional Block Attention Module," Proceedings of the European Conference on Computer Vision (ECCV), pp. 3-19, 2018.

[36] M. Nauta, D. Bucur, C. Seifert, "Causal Discovery with Attention-Based Convolutional Neural Networks," Mach. Learn. Knowl. Extr. 2019, 1, pp. 312–340.

[37] A. I. Borovykh, S. M. Bohte, C. W. Oosterlee, "Dilated convolutional neural networks for time series forecasting," J. Comput. Finance 2019, 22, pp. 73–101.

[38] Y. Qin, D. Song, H. Cheng, W. Cheng, G. Jiang, G. Cottrell, "A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction," In Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17) AAAI Press: Palo Alto, CA, USA, pp. 2627–2633 2017.

[39] K. Cho, B. Merrienboer, D. Bahdanau, Y. Bengio. "On the properties of neural machine translation: Encoder-decoder approaches," arXiv:1409.1259, 2014.

[40] C. Liu, S. C. H. Hoi, P. Zhao, J. Sun, "Online ARIMA algorithms for time series prediction," In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16); AAAI Press: Palo Alto, CA, USA, pp. 1867–1873, 2016.

[41] S. Tong, S. Koller, "Support vector machine active learning with applications to text classification," J. Mach. Learn. Res. 2, pp. 45–66, March 2002.

[42] H. Drucker, C. J. C. Burges, L. Kaufman, A. Smola, V. Vapnik, "Support vector regression machines," In Proceedings of the 9th International Conference on Neural Information Processing Systems (NIPS'96), MIT Press: Cambridge, MA, USA, pp. 155–161, 1996.

[43] M. Mishra, M. Srivastava, "A view of Artificial Neural Network," In Proceedings of the 2014 International Conference on Advances in Engineering & Technology Research (ICAETR-2014), Unnao, India, pp. 1–3, 1–2 August 2014.

[44] R. Pascanu, T. Mikolov, Y. Bengio, "On the difficulty of training recurrent neural networks," Int. Conf. Mach. Learn. 2013, 28, pp. 1310–1318.

[45] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, pp. 1800–1807, 21–26 July 2017.

[46] G. P. Zhang, V. L. Berardi, "Time series forecasting with neural network ensembles: an application for exchange rate prediction," Journal of the Operational Research Society, vol. 52, no. 6, pp. 652–664, 2001.

[47] G. P. Zhang, "Time series forecasting using a hybrid arima and neural network model," Neurocomputing, vol. 50, pp. 159–175, 2003.

[48] R. Philipperemy, Conditional RNN (Tensorflow Keras). GitHub Repository. 2020. Available online: https://github.com/philipperemy/cond_rnn (accessed on 4 June 2020).

[49] A. Jain and A. M. Kumar, “Hybrid neural network models for hydrologic time series forecasting,” Applied Soft Computing, vol. 7, no. 2, pp. 585–592, 2007.

[50] J. Hu, L. Shen, G. Sun, ”Squeeze-and-Excitation Networks,” In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018, pp. 7132–7141.

[51] K. He, X. Zhang, S. Ren, J. Sun, ”Deep Residual Learning for Image Recognition,” In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, pp. 770–778, 27–30 June 2016.

[52] S. Ruder, ”An overview of gradient descent optimization algorithms,” 2014, arXiv:1412.6980.

[53] G. Klambauer, T. Unterthiner, A. Mayr, S. Hochreiter, ”Self-Normalizing Neural Networks,” 2017, arXiv:1706.02515.

[54] F. A. Abien, ”Deep Learning using Rectified Linear Units (ReLU),” 2018, arXiv:1803.08375v2.

[55] D. P. Kingma, B. Jimmy, ”Adam: A Method for Stochastic Optimization,” 2014, arXiv:1412.6980.

Publication List of the Author

Publications in this dissertation

Referred Journals

[J-1] Y. Cheng, Z. Liu, Y. Morimoto, "Attention-Based SeriesNet: An Attention-Based Hybrid Neural Network Model for Conditional Time Series Forecasting," Information, 11(6): 305, June 2020.

[J-2] Y. Cheng, Y. Morimoto, "Triple-Stage Attention-Based Multiple Parallel Connection Hybrid Neural Network Model for Conditional Time Series Forecasting," IEEE Access, vol. 9, pp. 29165-29179, February 2021.

Referred Conferences

[J-3] Y. Cheng, Y. Morimoto, Competitors' Influence Analysis of a Retailer by Using Customer Value and Huff's Gravity Model", ICPTSP 2020 : International Conference on Probability Theory and Stochastic Processes Vancouver, Canada, September 23-24, 2020. (World Academy of Science, Engineering and Technology International Journal of Physical and Mathematical Sciences, vol. 14, no. 8, pp. 67-75, 2020.)

Other Publications (not in dissertation)

Refered Journals

[J-4] Y. Cheng, H. Okamura, and T. Dohi, "A Comprehensive Performance Evaluation on Iterative Algorithms for Sensitivity Analysis of Continuous-Time Markov Chains," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol. E103-A, no. 11, pp. 1252-1259, November 2020.

Non-Refered Conferences

[N-1] Y. Cheng, Y. Morimoto, "Analysis of Supermarket Customer Behavior by Using RFM and Huff's Gravity Model," Bulletin of Networking, Computing, Systems, and Software, vol. 8, no, 2, pp. 81-86, July 2019.

[N-2] Y. Cheng, H. Okamura, and T. Dohi, "A Comprehensive Performance Evaluation on Iterative Algorithms for Sensitivity Analysis of Continuous-Time Markov Chains," Summer Seminar in Operations Research, Central Forest Park, September 2017.