COMP 576 Final Project Report

Research on Time Series Deep Learning Models for COVID-19 Case Number Prediction

Dingrui Lei, December 2022

All completed by one person.

GitHub link: github.com/ldr426/RICE-COMP-554-Final-Project .

1. Background

Since Covid-19 epidemic has ravaged the world, it has become a major public health and safety issue in many countries. The national epidemic prevention policies is directly related to whether the epidemic can be effectively suppressed, and Covid-19 prediction model is very critical in policy formulation. The most commonly used epidemic prediction models are the traditional SI models. The traditional machine learning prediction models include linear regression and decision trees. The more popular machine learning prediction models include LSTM and GRU. A machine learning prediction model called Temporal Convolutional Networks (TCN) has recently emerged since 2018, which solves the gradient issue of other models and can handle longer time series. Time series Convolutional Networks can predict what is likely to happen in the next period of time by analyzing developments, directions, and trends in time series.

This project mainly focuses on the prediction of Covid-19 cases based on Temporal Convolutional Network. It aims to design and construct Temporal Convolutional Network, and on this basis, optimize and improve the model, integrating Temporal Convolutional Network, Long Short-term Memory Network and Gated Recurrent Unit Network to establish an optimized prediction model. The global COVID-19 case data collected by Johns Hopkins University is used as the dataset, and the prediction results should be compared and analyzed. The experimental results is used to proved the effectiveness of the Temporal Convolutional Network and the improved prediction models.

2. Existing Methods Analysis

2.1 SI Models

The SI model is a classic infectious disease prediction model. The simplest model is the SI model. It first divides the population into two categories. Another group of people is I (Infected, The Infected), which means people who are susceptible to infectious diseases. This model has two assumptions, that is, the total number of people during the epidemic period is constant, that is, the number of new births and natural deaths is not considered, and the number of people that individuals are exposed to every day is also kept constant.

Compared with the SI model, the SIR model has an additional marker population, namely R (The Removal), which represents the recovered population after illness. The key to this model is to study the relationship between the three groups of people. At the beginning of the outbreak, all individuals are unaffected and susceptible, but all individuals may be infected with infectious diseases; After contracting an infectious disease, it is transformed into an infected person; the

infected person will be affected by various recovery factors, including self-healing and receiving treatment, and finally become a recovered person. This model has three assumptions, that the total number of people remains constant, the rate of transition from susceptible to sick infected and the rate of transition from infected to sick recovered is constant, and the recovered are assumed to contain effective Antibodies will no longer cause disease.

The SIRS model is slightly more complicated than the SIR model. It no longer assumes that the recovered person will no longer get sick after recovery, but that the recovered person has a fixed immune time limit, and the antibody will be converted into a susceptible person after the antibody time limit. The model that best fits the spread of the novel coronavirus is the SEIR model, which incorporates the population E (The Exposed), which assumes that the virus has an incubation period.

2.2 Popular ML Models

Some scholars use the Long Short-Term Memory Model (LSTM, Long Short-term Memory) to predict the Covid. Compared with traditional feed-forward neural networks, this model adds the feature of feedback connections. The information flow throughout the memory is controlled by three gates, namely input gate, output gate and forget gate. The input and output gates control the flow of input activation information and the flow of output activation information, respectively. On the other hand, since the cell state is controlled by an input, the forget gate resets the memory cell when it is no longer needed. This feature solves the gradient disappearance problem of the traditional recurrent neural network model (RNN, Recurrent Neural Network), making this model a powerful tool for predicting the new Covid epidemic.

There are many variant models of the long-term short-term memory model, the most famous of which is the gated recurrent unit model (GRU, Gate Recurrent Unit). The purpose of GRU is to reduce the complexity of the long short-term memory model architecture and reduce its parameters. The update gate in the gated recurrent unit is a combination of the input gate and the forget gate in the long short-term memory model. Update gates provide control over how much memory is transferred to the new state. The function of the reset gate is similar to the forget gate in the long short-term memory model, which is responsible for forgetting information that is no longer used in the past. Compared with the long short-term memory model, the gated recurrent unit model can show better predictive performance on specific small-scale data sets.

2.3 Combined Models

Both models described above have their own drawbacks. The long short-term memory model cannot explain the dynamics of the diffusion process, and the error is too large. The infection rate and recovery rate of the traditional infectious disease model SEIR are constant, which is too simple to be of practical significance. Therefore, machine learning models and infectious disease models can be combined. First, learn and predict the dynamic parameters. The parameters in the SEIR model, including the infection rate and recovery rate, change over time and can be regarded as a set of time series, using the long-term short-term memory model and the gated recurrent unit model Learn and predict the dynamics of these two parameters and find the changing "reproduction rate" that is closely related to them. Then, discuss and analyze the relationship between the "reproduction number" and the epidemic trend of Covid cases through simple linear fitting. Afterwards, the dynamic parameters of the long-short-term memory model, the gated recurrent unit model and the SEIR model, which have been learned by the long-term short-term memory model and the gated recurrent unit model, are used to predict the epidemic trend of the Covid.

2.4 RNN Models Limitations

The network for time series modeling can be divided into recurrent neural network and convolutional neural network, among which time convolutional network is a kind of convolutional neural network. Although recurrent neural networks have achieved great success in many translation tasks, many recent studies have shown that convolutional architectures achieve state-of-the-art performance on many specific tasks. Because there is a issue with recurrent neural networks, which are relatively complex sequential models, they pass hidden activation vectors to propagate over time. Employing recurrent neural networks on longer time series tasks poses difficulties, since modeling very long sequences of observations of thousands of data points is often required, and hidden activation vectors must propagate from the beginning to the end of the sequence and propagate back again to update weights.

3. Goal and Feasibility

3.1 Goal

This paper discusses the composition principle of temporal convolutional network. Based on the existing machine learning framework, a model of temporal convolutional network is developed and some modifications are made to the model. The transformation includes combining with other models, combining temporal convolutional networks with long short-term memory networks and combining temporal convolutional networks with gated recurrent unit networks. Possible combinations also include adding a self-attention mechanism, but this modification is more suitable for the case of multi-dimensional variable input.

The Covid case data is divided into two groups, a large data set is used to predict 30 days, and a small data set is used to predict 7 days. The temporal convolutional network model, the long short-term memory model, the gated recurrent unit model, the combined model of the temporal convolutional network model and the long short-term memory model, and the combined model of the temporal convolutional network model and the gated recurrent unit model were trained, The predicted values were compared accordingly, and the training time between the models was compared.

3.2 Feasibility

As a Temporal Convolutional Network based on convolutional neural network, causal convolution, dilated convolution and residual connection are its most important features. Temporal Convolutional Networks have the following salient features:

1. The convolution kernel is a causal relationship, and future information cannot be known from the past;
2. A sequence of any size can be taken and mapped to an output sequence of the same length;
3. Use a combination of deep neural and dilated convolution kernels to capture extended effective historical information in the network;
4. Historical information can be used for future state prediction.

Compared with the Long Short-term Memory model and the Gated Recurrent Unit model, the Temporal Convolutional Network has the following advantages:

1. Due to dilated convolutions, Temporal Convolutional Networks have a larger receptive field of view, allowing longer input sequences to be processed. The view field size is flexible and can be easily adapted to different tasks by changing the number of layers and using different dilation numbers. However, the traditional recurrent convolution network can only input all the existing historical information, and cannot achieve fine control like temporal convolution.
2. Due to the acyclic architecture, i.e. using a different back-propagation path rather than the sequential direction, and the use of residual blocks and residual connections, Temporal Convolutional Networks are not the same as gradient-progressive as traditional recurrent networks such as LSTM networks, which does accumulation along the time dimension during back propagation, avoiding the gradient divergence problem of the traditional recurrent network, and the gradient is more stable.
3. Faster training can be processed once as an input sequence, without waiting for the prediction of the previous timestamp to be completed before the current timestamp, instead of being processed sequentially like a traditional recurrent network, that is, having to wait for the previous time period to complete, and the next time period can only be passed after the forward pass is completed. Convolutions can be computed in parallel because the processing is the same for each layer.
4. Temporal Convolutional Network has a simpler and clearer architecture and operation, because the Temporal Convolutional Network does not include gates and gate mechanisms like traditional recurrent convolutional networks such as LSTM models and GRU models.

4. Solution and Modeling

4.1 Structure Diagram

The entire experimental system architecture is shown in Figure 1 of the experiment structure. After preprocessing the obtained Covid data, it is input into the temporal convolutional network for training. The temporal convolutional network contains many residual blocks, and each residual block contains Two convolutional areas, each convolutional area contains one-dimensional causal convolution, weight normalization layer, activation function layer and dropout layer. The causal convolution of each residual block changes as the number of layers increases. The output vector can continue to be input into the long short-term memory model or the gated recurrent unit model for training, or it can be skipped. After adding the Dropout layer and the fully connected layer, make predictions. After drawing the prediction map, calculate the corresponding evaluation indicators, compare and analyze the prediction results with other models, and draw the final conclusion.

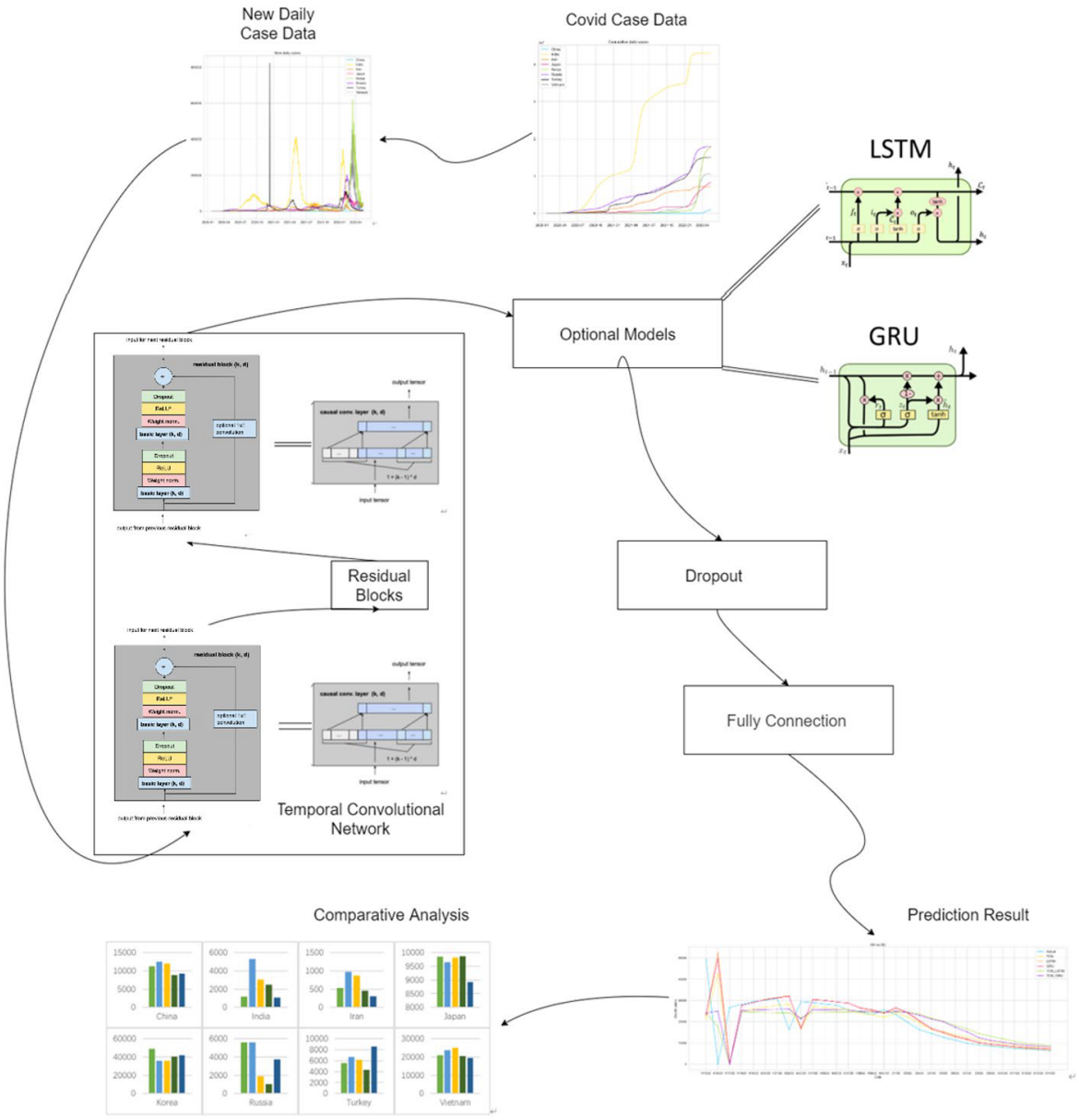

Figure 1.Experiment structure

4.2 TCN Construction

4.2.1 1D convolution

Typically, the input of a 1D convolutional network has three dimensions, and the output also has three dimensions. Our temporal convolutional network is also one-dimensional convolutional network. The input parameters include the number of parameters passed to the program for training at a time, the input length and the input size. The output parameters include the number of parameters passed to the program for training, the input length and the output size.

This experiment is a single variable experiment, so the input size and output size are set to 1 accordingly. But usually in experiments, there are often more than one variable, so the input size and output size are generally not the same as a single variable.

Figure 2 shows how a one-dimensional convolutional layer works. This is a very simple example, that is, a single variable, the input size and output size are both 1.

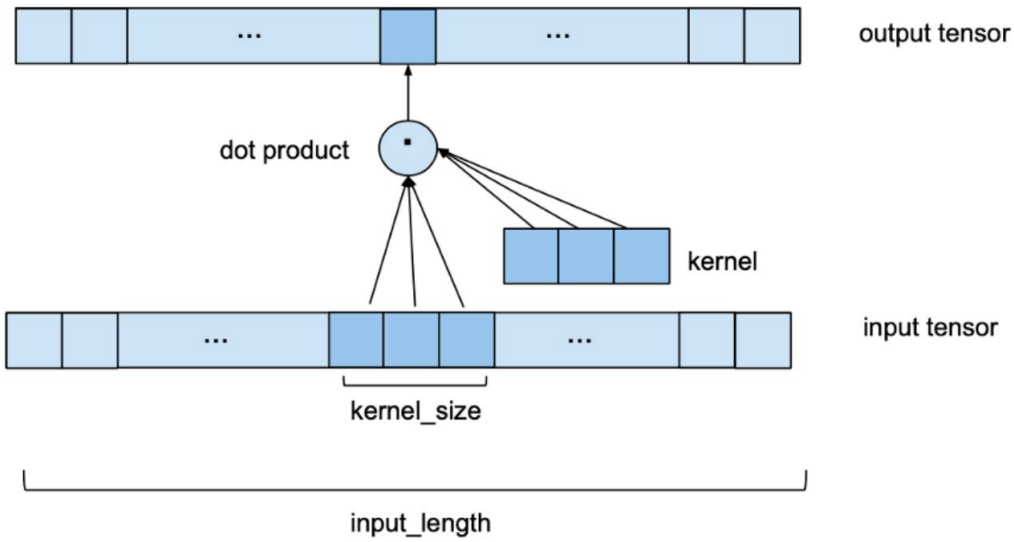

Figure 2. Convolutional network processes 1D input

Through the above description, it is much more convenient to understand the principle of one-dimensional convolutional network. Similar to the sliding window algorithm, the window size is the size of the convolution kernel. As shown in Figure 3, through a convolution kernel with a size of 3, we need to continuously input the input sequence of the same length and the vector in the convolution kernel that has learned the relevant weight value to perform the corresponding dot product operation. The window shift value is set to 1, which is called stride in machine learning. After the calculation is completed, the window slides to perform the calculation of the next sequence. In order to make our explanation more focused, the dot product operation in the figure is not shown, but the corresponding input and output are shown.
At this time, a problem will arise. Since a sliding convolution kernel is required, when the size of the convolution kernel is greater than 1, there will inevitably be a situation where the output length is smaller than the input length. In order to solve the problem of inconsistent input length and output length, we introduce padding, which is usually 0. so that the output length is equal to the input and output length. This issue will be discussed in detail later.

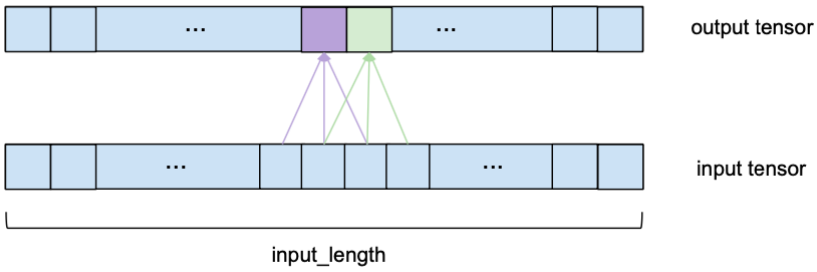

Figure 3. Two consecutive input and output

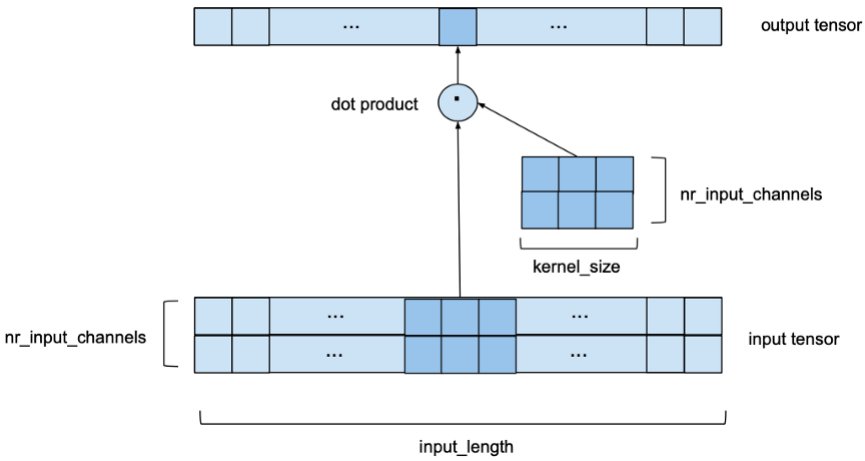

Figure 4. Input of Multidimensional Variables

Although this experiment is a univariate experiment, in general, the input size is often greater than 1, that is, it is not univariate. In this case, we need to increase the dimensionality of the convolution kernel.

The input of the corresponding row is dot-product of the convolution kernel of each row, and the results obtained are added together, which is the output result. In Figure 4, although the shape of the input sequence and the convolution kernel are the same and two-dimensional convolution is performed, the whole model looks like a two-dimensional convolution, but we still define it as one-dimensional, because the sliding window is still only Swipe in one direction.

4.2.2 Causal Convolution and Zero Padding

COVID-19 case data is usually in time series, meaning that the current state is influenced by previous states. Consider an input sequence [x0,...,xt], and an output sequence [y0,...,yt], the input sequence can be used for output prediction. According to the characteristics of time series, two principles should be considered: the output sequence generated by the network has the same length as the input sequence and information cannot be leaked from the future to the past.

As we mentioned in the previous section, in order to make the input length equal to the output length, some padding needs to be done on the left side of the input sequence, usually the padding value is 0, which can make the time series features reasonable. Because assuming that the right part of the input sequence is not filled, the last item of the input dependency of the last output can only be the last input, and the corresponding input window of the second-to-last element slides to the left by 1 bit, so The last item of the dependent input of the second-to-last output is the second-to-last input, and so on, the index of the last item in the input-dependent sequence of the output sequence corresponds to itself.

So it makes sense to fill in the corresponding missing values in front of the first item of the input sequence. The size of a convolution kernel is 3, the input length and output length are 4, and an example of the corresponding relationship after adding the header padding value is shown in Figure 5.

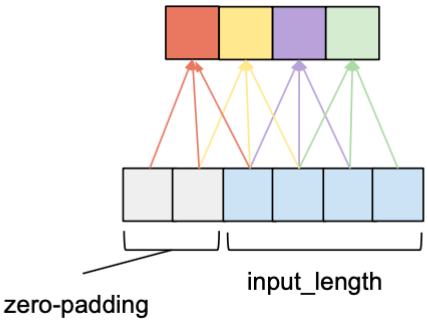

Figure 5. Header zero padding

4.2.3 Dilation

Header zero padding satisfies the hard requirement that the length of the input sequence must be equal to the length of the output sequence. The predictive model also needs to satisfy that information cannot leak from the future to the past, in other words, the current output only depends on the input before its corresponding index. But a high-quality prediction model needs to make the current output depend on all inputs before the corresponding index. We stipulate that the specific input subsequence on which the specified output depends is the receptive field, and when the size of the receptive field is the length of the input, it is a perfect record. Usually the output of a convolution layer depends on the receptive field of the convolution kernel size, and the index of the receptive field is less than or equal to the output index.

For example, if the convolution kernel size is 3, the 6th output of the output sequence depends on the 6th, 5th and 4th input of the input sequence. When layers and barriers are stacked, the original receptive field expands. As shown in Figure 6, when the sequence is stacked after the convolution operation with a convolution kernel size of 3, the receptive field size of the last output becomes 5.

From this we can easily deduce that the size of the convolution kernel is k, and the receptive field of a network with n layers is n×(k-1)+1. Conversely, if you know that the size of the receptive field is f, you can find the number of layers of the network (f-1)/(k-1), and you need to round up.

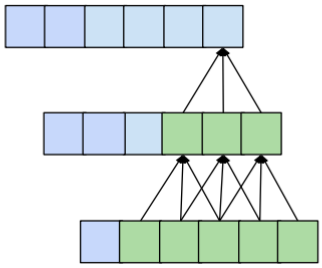

Figure 6. Receptive field

The experiments use a 1D fully convolutional network to preserve dimensionality, where each hidden layer has the same length as the input layer. In addition, causal convolutions are used in temporal convolutional networks to ensure that only layers from the corresponding time ago are used to obtain the output.

In the above discussion, we proved that all hidden layers have the same length as the input and output layers. However, the historical influence on the current state is limited, which leads to a rapid increase in computational complexity when the network becomes too deep, because the convolution kernel size is fixed, and if you want to get a high-quality predictive model, you must

ensure the integrity The input information coverage, the required number of network layers can be seen from the above derivation formula to increase linearly. In this case, it will inevitably consume a lot of computing power and resources for training. Moreover, if a large number of layers are added to the network, the gradient of the loss function will disappear or explode, which makes our model and the cyclic neural network have no essential difference, and the advantages will disappear. To solve this problem, the added hidden layer needs to consider more instances, so that dilated convolutions are added in the temporal convolutional network to achieve a larger receptive field without increasing the number of hidden layers. From this, the concept of dilation is introduced, in other words, increasing the size of the receptive field. Dilated convolutions can be used to replace traditional causal convolutions to reduce computational requirements while retaining significantly more historical information.

Dilation is the competitive weapon of convolutional neural network for recurrent neural network. The dilation between convolutional layers is achieved by expanding the distance between each input in the input sequence. In simple terms, it is "jumping" the input. We stipulate that the value of "jump" is the dilation value, and the transformed convolutional layer becomes the dilation layer.

The general convolutional layer dilation value is 1, that is, continuous input, without skipping any elements. Figure 7 shows an dilation layer with a dilation value of 2, a convolution kernel size of 3, an input and output length of 4, and a head padding of 0. Compared with the traditional convolution layer, that is, the dilation layer with the dilation value of 1, the receptive field length of the dilation layer with the dilation value of 2 is 5 instead of 3. But there is still a problem. This method only reduces the number of layers at a constant level, and the number of layers still increases at a linear level.

We need to reduce the number of layers by introducing an exponential method. Some neurons in the hidden layer are regularly ignored according to the decline rate. For this, a base dilation number, base, can be defined. The decline rate is determined by the base dilation number. Define the dilation value of the i-th layer as basei, which is the base dilation number i power. The network shown in Figure 8, the input sequence is 10, the basic dilation number is 2, and the convolution kernel size is 3. This allows only 3 dilation layers to achieve output predictions corresponding to the complete input sequence preceding its sequence.

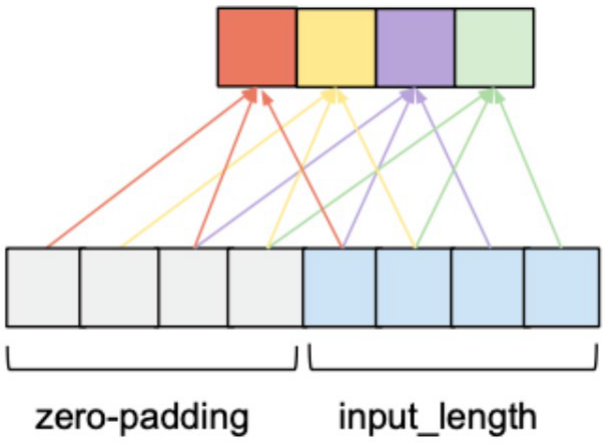

Figure 7. Dilation layer

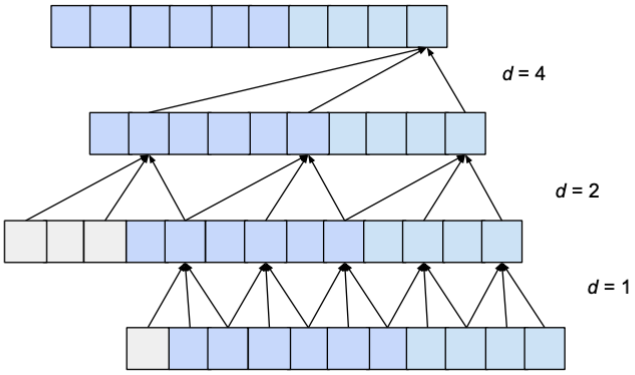

Figure 8. Dilation layer after adding the base dilation coefficient

For predictive models that pursue high-quality predictions, it is obvious that the last item output depends on all input items before the corresponding index. The above figure only shows the dilation layer for the last output and the related header 0 padding operation. Normally, the maximum input length is 15 when the input information receptive field is fully recorded.

Every time an dilation layer is added, the length of the receptive field in the i-th layer needs to be increased by k×bi, that is, the size of the convolution kernel multiplied by the dilation value. Given the basic dilation number b, the convolution kernel size k and the number of layers n, the length of the receptive field can be obtained as w.

$$\omega=1+\sum_{i=0}^{n-1}(k-1)\cdot b^i=1+(k-1)\cdot\frac{b^n-1}{b-1}$$

We've fixed the layer increase problem, changing the number of layers from a constant decrease to an exponential decrease. However, new problems have emerged. The combination of some convolution kernel sizes and basic dilation numbers will cause "seams" in the receptive field. As shown in Figure 9, when the basic dilation value is 3 and the convolution kernel size is 2, there are "seams" in the receptive field.

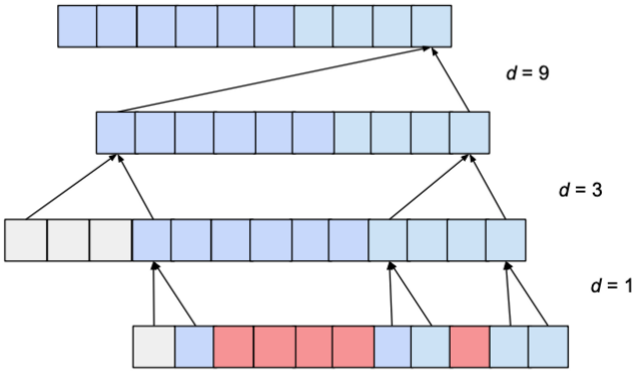

Figure 9. Receptive field where the "seam" occurs

What is "sewing"? As the name suggests, when there is a "seam" in the receptive field, the output item only depends on the corresponding part of the input item before its index value. As shown above, the output value of the last item has nothing to do with the input value of the red part. This resulted in our models not meeting high-quality standards. This can be avoided by changing the size of the base dilation number and the size of the convolution kernel. For example, reducing the number of base dilations to 2, or increasing the kernel size to 3, can resolve the seams that appear in the image above.

Usually, in order to avoid "seams" in the receptive field, the convolution kernel size cannot be smaller than the base dilation number. Given the convolution kernel size k, the basic dilation number b, the convolution kernel size must not be smaller than the basic dilation number, the number of layers n, and the input length l, in order not to appear "seams", the inequality must be satisfied:

$$1+(k-1)\cdot\frac{b^n-1}{b-1}\geq l$$

The current number of layers has changed from linear growth to logarithmic growth relative to the input sequence. And it can be achieved by avoiding "seams". We can now use below formula to calculate the number of layers n, the convolution kernel size is k, and the number of basic dilations is b, where k≥b.

$$n=\left\lceil \log_b\left(\frac{(l-1)\cdot(b-1)}{k-1}+1\right)\right\rceil$$

Now it is still necessary to calculate the number of 0s in the head of each layer, the number of 0s in the head of each layer is p, the size of the convolution kernel is k, the number of basic dilations is b, i is the layer index, k≥b .

$$p=b^i\cdot(k-1)$$

4.2.4 Overview of TCN

Temporal convolutional networks replace the last fully connected layer in traditional convolutional neural networks with a one-dimensional fully convolutional network architecture. Temporal ConvNets employ dilated causal convolutions for each layer. Causal convolution means that the output at a particular timestep is only computed based on the previous layer at or before that particular timestep, and there is no information leakage from future loading after a particular timestep.

Causally dilated convolutions make the effective receptive fields of temporal convolutional networks grow logarithmically with the number of layers. Specify the size of the convolution kernel, the number of basic dilations, the input sequence and the minimum number of layers. The most basic time convolutional network is shown in Figure 10.

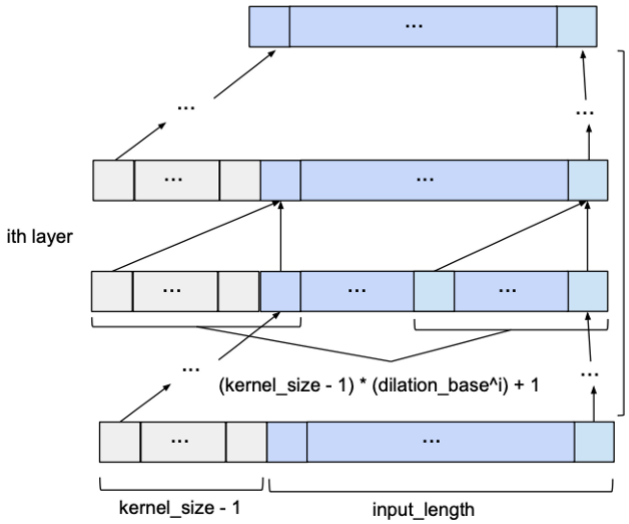

Figure 10. Basic Temporal Convolutional Networks

4.2.5 Prediction

How the input is related to the corresponding output is the prediction. In the above, we only pointed out the basic structure of the temporal convolutional network, but did not explain how to predict.

To enable forecasting, we split the time series used for the training set into input and indicator series. The index sequence is shifted forward by the length of the output sequence relative to the input sequence. In other words, the index sequence with the same length as the input sequence and the sequence of the input sequence minus the output sequence length share the first item, and the item after the input sequence slips the output sequence length is the last item of the output sequence. The maximum predicted length of the model is the length of the output sequence. Through slippage, we can predict a corresponding sequence from this, as shown in Figure 11.

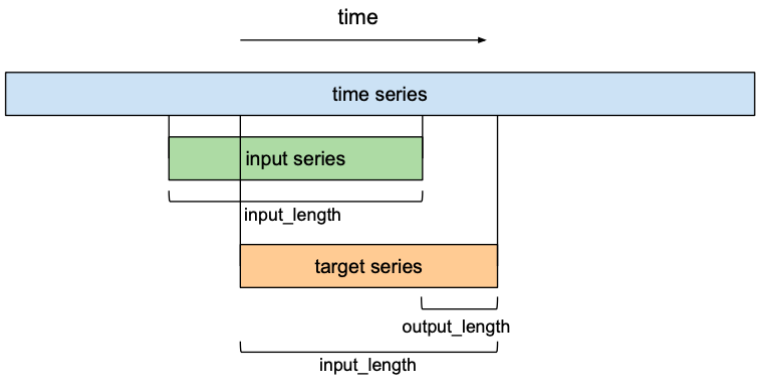

Figure 11. Input sequence, index sequence and output sequence

4.2.6 Residual Block

In addition to computational demands, deepening of hidden layers in neural networks can lead to vanishing or exploding gradients, where parameters are not efficiently updated forward, leading to abnormal termination of the algorithm. Residual connections can overcome this problem.

Residual blocks in temporal convolutional networks significantly guarantee the performance of deep models to maintain their ability to replicate linear relationships within an elastic range, as well as the ability to offload and reload. A residual block is formed by connecting two 1D causal convolutional layers with the same dilation number via the residual. Figures 12 and 13 show how a convolutional layer with a dilation number of 2 and a kernel size of 3 performs residual stacking.

Two layers of convolutional layers make up the residual block. For all residual blocks in the network, the first block is called the encoding block and the last block is called the decoding block. Because except for the encoding block and decoding block, the input and output widths of the remaining hidden blocks are the same. The encoding block and decoding block are adjusted in width by a convolution kernel of size 1. This change ensures the calculation of the minimum number of layers for complete information entry.

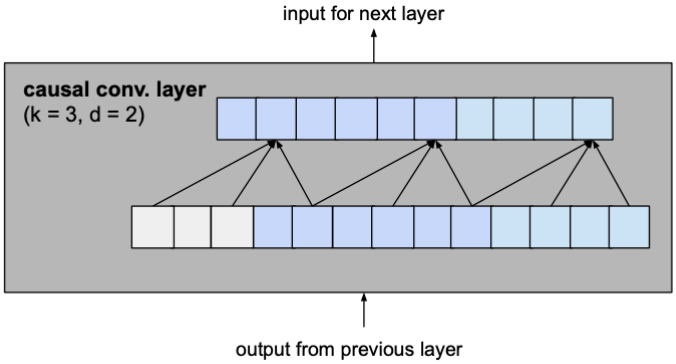

Figure 12. Convolutional layers before residual connections

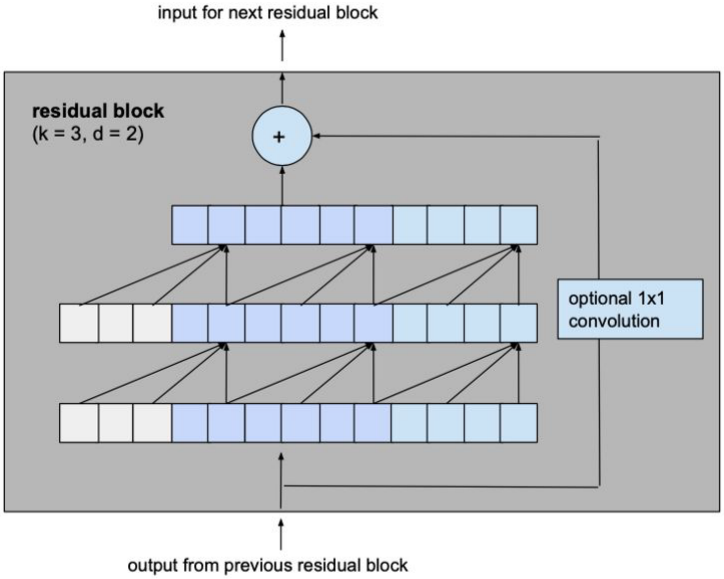

Figure 13. Convolutional layer after residual connection

It is easy to deduce that adding a residual block to a temporal convolutional network increases the receptive field twice as much as adding only one causally dilated convolutional layer. Equations below show how to calculate the total size of the receptive field r and the number of residual blocks n, the convolution kernel size is k, the number of base dilations is b, k≥b.

$$r=1+\sum_{i=0}^{n-1} 2\cdot(k-1)\cdot b^{i}=1+2\cdot(k-1)\cdot\frac{b^{n}-1}{b-1}$$

$$n=\left\lceil \log_{b}\left(\frac{(l-1)\cdot(b-1)}{(k-1)\cdot 2}+1\right)\right\rceil$$

So far, our temporal convolutional network seems to be just an abstract and complicated linear regression model. Therefore, it is necessary to add an activation function to the convolutional layer to make the network nonlinear. After inserting it into the two convolutional layers with the ReLU function, this extra path effectively enables the layers to learn to modify the corresponding mappings, rather than wasting computational power on overall transformations. This is very useful for deeper networks because the network is enhanced to maintain input vectors through deeper networks.

There are no pooling or fully connected layers in the architecture. We need to standardize the weight of the model. The commonly used method is norm weight. As the name suggests, it is to standardize the input of the hidden layer to solve the problem of gradient explosion. The model needs to be regularized, and the commonly used method is dropout, which is a means to prevent overfitting. A corresponding regularization operation is performed in each residual block, usually

introduced after each convolutional layer. Figure 14 shows a complete residual block, the asterisk indicates no activation in the last layer because the output may have negative values.

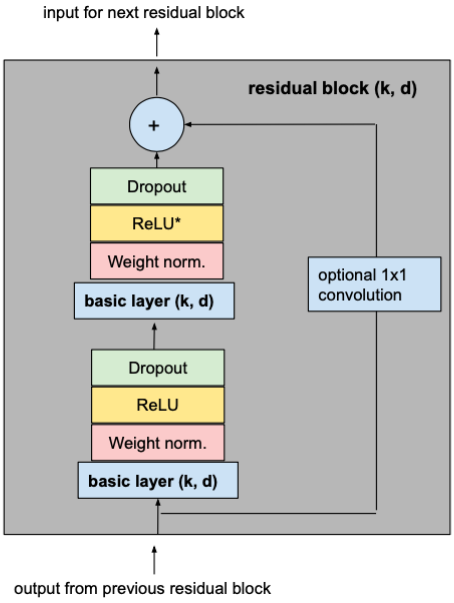

Figure 14. Complete Residual Block

4.2.7 Overall structure of TCN

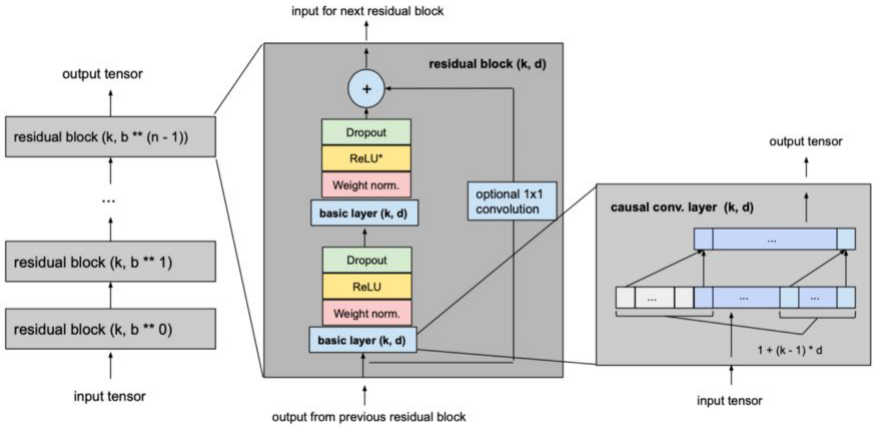

Figure 15. Complete Temporal Convolutional Network Model

4.3 Models in Experiments

1. Time Convolutional Network: the number of convolutions is 32, and the size of the convolution kernel is 3; the dropout layer is 0.2; one neuron is in the output layer to predict Covid-19 cases; the dilation value is [1, 2, 4, 8, 16]. The input shape is 1 time step, 1 feature.

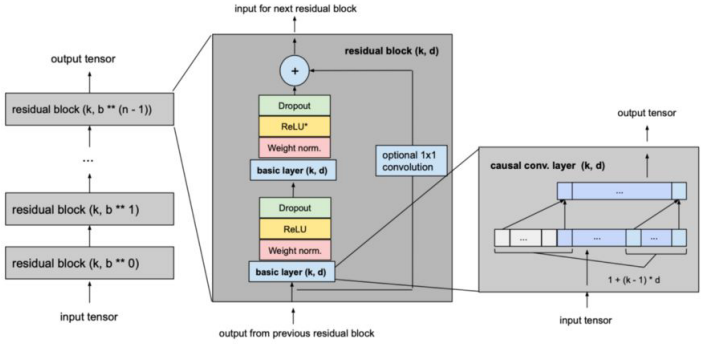

Figure 16. Temporal Convolutional Network Model

2. Long Short-term Memory Network: 100 units in the hidden layer; 0.2 in the dropout layer; 1 neuron in the output layer for predicting Covid-19 cases; the input shape is 1 time step and 1 feature.

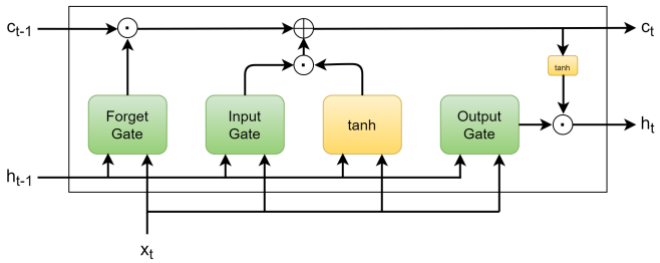

Figure 17. LSTM unit

3. Gated Recurrent Unit Network: the hidden layer unit is 100; the dropout layer is 0.2; 1 neuron is in the output layer, used to predict Covid-19 cases; the input shape is 1 time step and 1 feature.

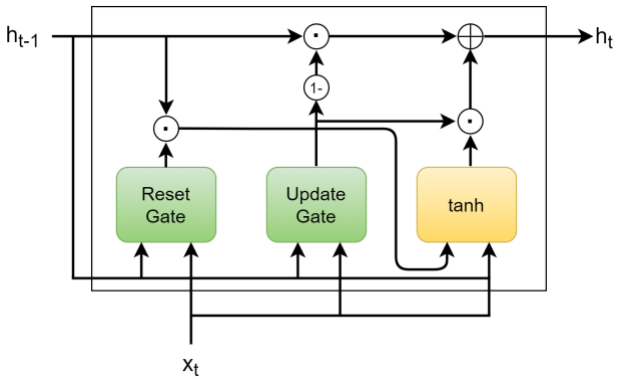

Figure 18. GRU unit

4. Temporal Convolutional Network combined with LSTM: a combination of (1) and (2).

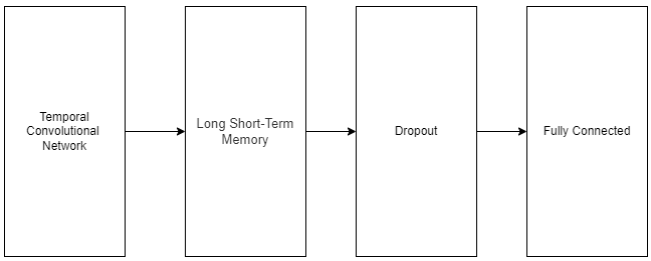

Figure 19. Combined LSTM and TCN

5. Time Convolutional Network combined with GRU: combination of (1) and (3).

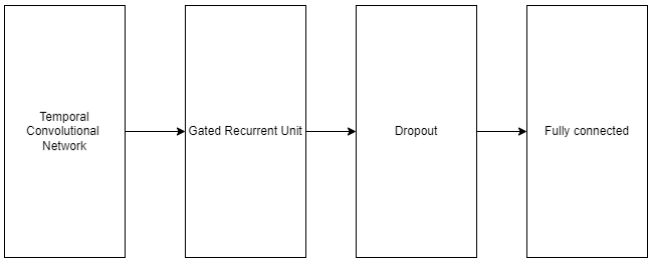

Figure 20. Combined GRU and TCN

5. Experiments

5.1 Flow Description and Hypothesis

This chapter is the experimental part, and the experimental process is shown in 4.1. The Covid data for the experiment will be collected first. Here, the data of 8 countries will be used, and the experiments will be divided into two groups for comparison. One group is to train a small data set for 7-day predictions, and the other group is to train a large data set for 30-day predictions. , divide the data set and perform normalized preprocessing on the data. Then the relevant models were networked. In this experiment, five models were set up for comparison, namely temporal convolutional network, long-term short-term memory network, gated recurrent unit network [14], temporal convolutional network and A combined network of long short-term memory networks, a combined network of temporal convolutional networks and gated recurrent units. After networking, the hyperparameters, optimizers and loss functions required for training will be defined. After training and prediction, the obtained results are denormalized and then drawn. Compare and analyze the results of corresponding experiments, including analyzing whether the predictions of each model are accurate and comparing training and prediction time. Prediction is TCN-LSTM and TCN-GRU are better than TCN, and TCN is better than LSTM and GRU.

5.2 Dataset

The Covid-19 case number collected from Johns Hopkins University is imported as dataset. We can get the data through the university's official GitHub, and then save it in a csv file. The data is saved in the form of Dataframe in python. Models train on two sets of data .

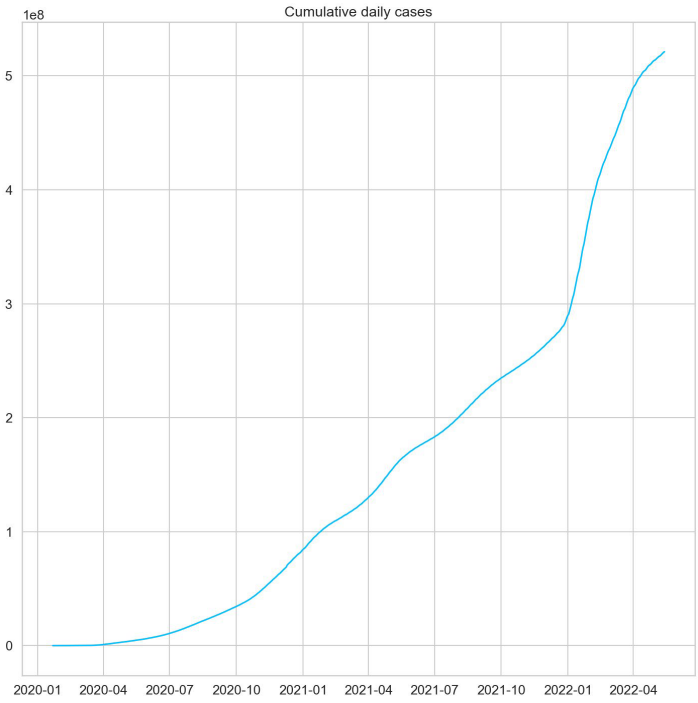

Figure 21. Cumulative line chart of global Covid cases

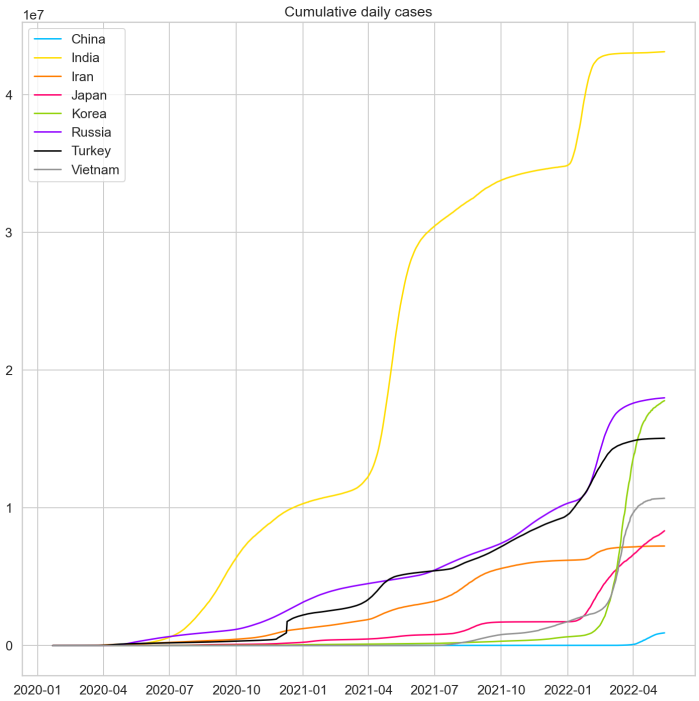

Figure 22. Line chart of cumulative cases of Covid in 8 countries in Asia

In order to highlight the comparative type of the experiment, this experiment analyzed and predicted the Covid-19 cases in 8 countries in Asia, including China (Mainland), India, Iran, Japan, South Korea, Russia and Vietnam. Generally speaking, for time series, we do not directly train its value, and usually perform a first-order difference operation on the data. First-order difference, as the name implies, in the case of the same time series element interval, the new sequence obtained by subtracting one value from the next value is the first-order difference sequence. In addition to the first-order difference, we can also take the second-order difference. The second-order difference sequence is to perform a first-order difference on the first-order difference sequence. Why take the first order difference of the time series? Because the time series usually has the characteristic of large volatility, that is, the random trend of dissociation, the use of difference can alleviate the irregular fluctuations between time series elements. After all, the stationarity of time series is usually difficult to achieve.

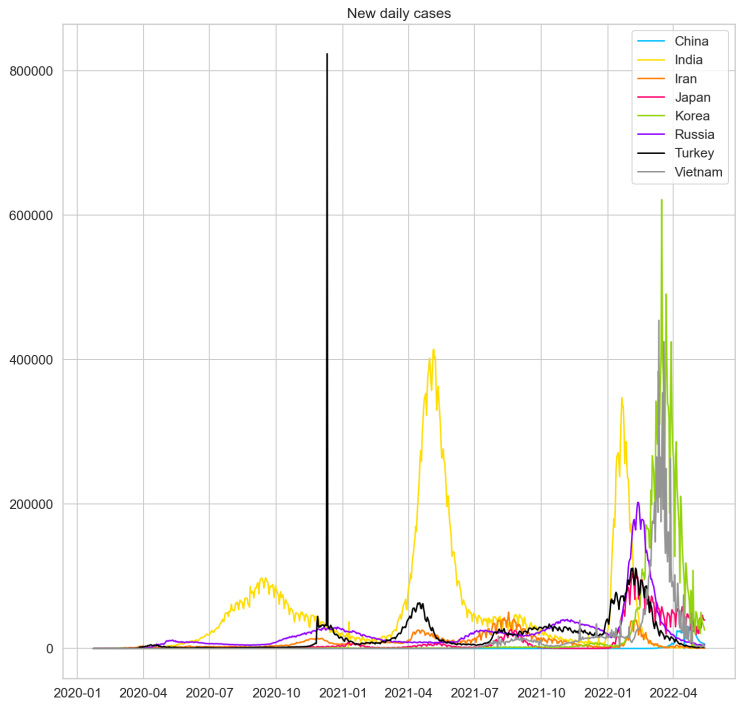

Figure 23. Line chart of daily increase in Covid cases in 8 countries in Asia

Figure 24 shows the data trend of the cumulative Covid cases in 8 Asian countries, including China (Mainland), India, Iran, Japan, South Korea, Russia and Vietnam, as shown in Figure 24, which shows the daily increase in Covid cases. As shown in Table 4.1, the data of the daily increase of Covid cases in 8 countries are also analyzed, and the minimum value (Min), maximum value (Max), average value (Mean), median (Median), standard deviation (Std), skewness (Skew) and kurtosis (Kurtosis).

Country	Min	Max	Mean	Median	Std	Skew	Kurtosis
China	0	49,439	1,094.07	16	4,810.34	5.27	30.21
India	0	414,188	51,152.55	22,890	79,931.31	2.68	6.97
Iran	0	50,288	8,573.76	5,960	9,217.63	1.76	2.83
Japan	0	104,345	9,887.14	1,333	20,115.59	2.58	6.18
Korea	0	621,317	21,093.78	556	70,253.63	4.3	20.17
Russia	0	202,211	21,339.34	13,742	30,150.23	3.85	16.29
Turkey	0	823,255	17,856.66	7,706	35,211.0	14.69	325.35
Vietnam	0	454,212	12,686.88	16	41,904.06	5.82	41.16

Figure 24. Analysis of Covid data in 8 countries in Asia

It needs to be explained that in the statistics of the agency, the number of new cases in China on the first day was -80, which is not in line with the routine, and the data will be adjusted to 0 cases in the next day.

The overall value of China's data is relatively small, relatively stable in the early stage, and explosive in the later stage, so the average and median are small, and the kurtosis and skewness are both high. The Vietnamese data is an enhanced version of the Chinese data. The kurtosis and skewness are similar to those of China, and the trend is similar to that of China, but the average data is much larger than that of China. India's data has three peaks, the total number of cases is the highest among several countries, and the standard deviation is the largest, indicating that the distribution is relatively scattered. The total number of cases in Iran is smaller, but there are several peaks. Japan and Russia exploded more strongly in the later stage, although there were also peaks in the early stage. South Korea broke out completely in the late stage, and the peak in the early stage was insignificant compared with the late stage. The data in Turkey is the most weird. The maximum value is particularly prominent, which can be explained as an outlier. Both the kurtosis and skewness become very large due to the maximum value.

In order to improve the performance and convergence ability of the model, feature scaling of the data is also required. Here we need to normalize the first-order difference data to scale the data values, including the maximum and minimum values, to the range of [0, 1].

The data set is divided into training data set and test data set. We set up two sets of experiments to compare the predictive ability of time series network for different test set lengths. One set is 30 days as the test set, and the other set is 7 days as the test set.

The first set of training sets is the data from January 23, 2020 to April 14, 2022, with a total of 813 data. The first set of test sets is from April 15 to May 14, 2022, with a total of 30 data.

The second set of training sets is the data from February 26, 2021 to May 7, 2022, with a total of 435 data. The second set of test sets is from May 8, 2022 to May 14, with a total of 7 data.

5.3 Experimental Setup

All training and predictions were performed on a Microsoft Surface PC with a CPU of Core i5-1035G4, 8GB of memory, and a processor speed of 1.5GHz. Because the training data is not very large, the GPU accelerated model training is not used, but the GPU settings can be enabled in the code.

The programming environment uses Jupyter Notebook. Use artificial neural network libraries when writing temporal convolutional networks, including the open source libraries keras and paddle. You need to use the Layer, Conv1D, Dense and other classes in the keras library to write the TCN model yourself. The sklearn library needs to be imported to normalize the data. Import matplotlib and pylab for drawing.

The five models used are:
(1) Time convolutional network: the number of convolutions is 32, and the size of the convolution kernel is 3; the dropout layer is 0.2; the dilation value is [1,2 ,4,8,16]. The input shape is 1 time step, 1 feature; one neuron is in the output layer to predict Covid cases;
(2) Long-term short-term memory network: 100 units in the hidden layer; 0.2 in the dropout layer; 1 neuron in the output layer for predicting Covid cases; the input shape is 1 time step and 1 feature.
(3) Gated recurrent unit structure: the hidden layer unit is 100; the dropout layer is 0.2; 1 neuron is in the output layer, used to predict Covid cases; the input shape is 1 time step and 1 feature.
(4) Temporal convolutional network combined with LSTM network: a combination of (1) and (2).
(5) Time convolutional network combined with GRU network: combination of (1) and (3).

All models set a unified optimizer, loss function and hyperparameters. Using the Adam optimizer, the loss function is the mean square error loss function. The Adam optimizer is used to iteratively optimize the weights of the network using mean absolute error (MAE, Mean Abstract Error) as a loss function. The reason why ADAM optimizer is chosen is because in some countries the Covid data is sparse, Adam provides adaptive learning rate, which is very suitable for such data, so we can optimize the learning rate to a large extent when using ADAM. Set the number of training times to 100. Through testing experiments, it is found that when the batch size is about 8, the result is the best, so set the batch size to 8. The learning rate is set to 1e-3. The default shuffle (shuffling) is false.

Use below evaluation indices for analyzing results:
RMSE (Root-mean-square deviation); NRMSE(Normalized root mean square error);
MAPE(Mean absolute percentage error); SMAPE(Symmetric mean absolute percentage error).

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=0}^{n-1}(y_i - \hat{y}_i)^2}$$

$$NRMSE = \frac{RMSE}{y_{\max} - y_{\min}}$$

$$MAPE = \frac{10}{n}\sum_{i=0}^{n-1}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

$$SMAPE = \frac{200}{n}\sum_{i=0}^{n-1}\left|\frac{y_i - \hat{y}_i}{y_i + \hat{y}_i}\right|$$

After the model training is completed, test the models trained on two sets of data. The obtained data is still normalized data, which needs to be denormalized, converted into original data, and then draw the corresponding predicted case curve and actual case curve, and calculate the corresponding four predictive indicators, and draw the predictive indicator table.

5.4 Results

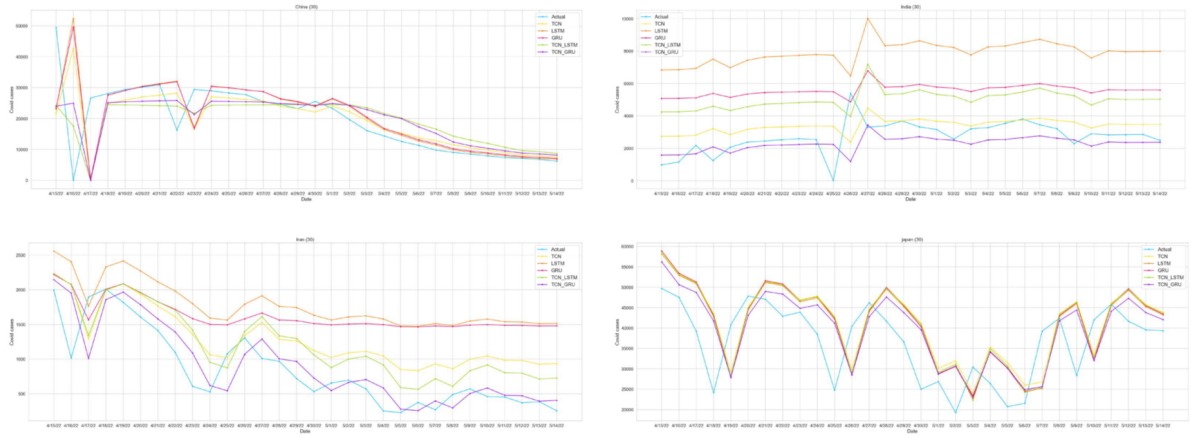

Figure 25. 30 days prediction results(China, India, Iran, Japan)

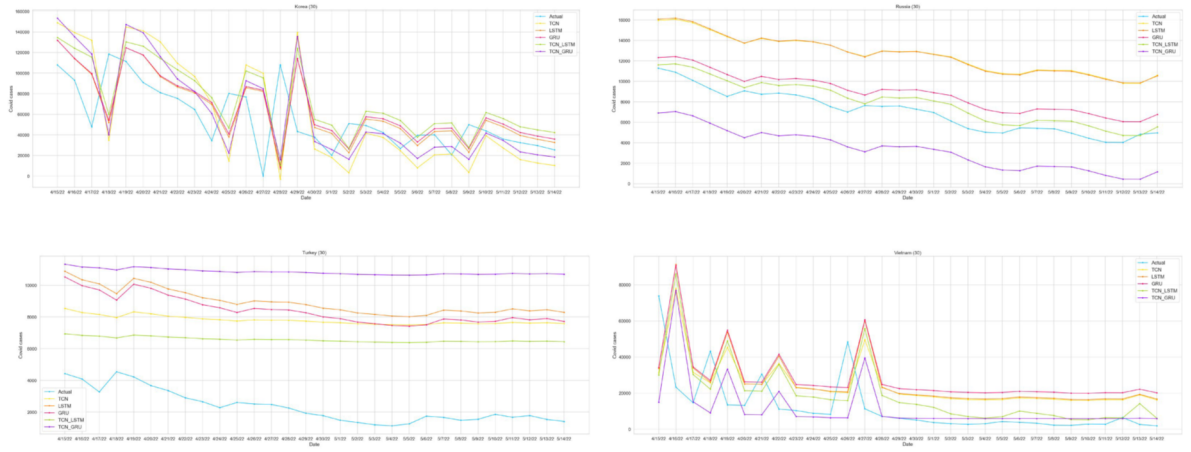

Figure 26. 30 days prediction results(Korea, Russia, Turkey, Vietnam)

Country	Model	RMSE	NRMSE	MAPE	SMAPE
China	TCN	11167.6772	0.2259	29.5808	30.5927
	LSTM	12440.0871	0.2516	29.2962	26.7535
	GRU	12029.4648	0.2433	29.411	28.0233
	TCN-LSTM	8796.1288	0.1779	32.1993	39.4308
	TCN-GRU	9154.2246	0.1852	30.4133	35.4929

Country	Model	RMSE	NRMSE	MAPE	SMAPE
Korea	TCN	48739.5866	0.4115	67.4075	70.6973
	LSTM	35735.5349	0.3017	46.9622	48.2183
	GRU	35790.6289	0.3022	48.363	49.0778
	TCN-LSTM	40333.8167	0.3405	58.7166	55.3314
	TCN-GRU	41804.2567	0.3529	55.0925	52.5964

Country	Model	RMSE	NRMSE	MAPE	SMAPE
India	TCN	1181.5873	0.2193	32.8627	34.4702
	LSTM	5285.6732	0.981	190.7179	100.5419
	GRU	3010.911	0.5588	106.4633	73.9931
	TCN-LSTM	2486.3296	0.4615	87.6439	66.0039
	TCN-GRU	1055.3844	0.1959	26.635	31.6509

Country	Model	RMSE	NRMSE	MAPE	SMAPE
Russia	TCN	5615.9114	0.7742	79.9078	59.7234
	LSTM	5611.044	0.7735	79.8662	59.6731
	GRU	1893.1688	0.261	26.3123	25.2092
	TCN-LSTM	1045.6952	0.1442	13.6354	13.4711
	TCN-GRU	3718.0695	0.5126	52.7438	83.1761

Country	Model	RMSE	NRMSE	MAPE	SMAPE
Iran	TCN	526.3353	0.2955	55.8775	56.0965
	LSTM	977.0845	0.5486	108.6347	82.1005
	GRU	866.652	0.4866	92.7583	75.7384
	TCN-LSTM	459.4107	0.258	47.593	48.3818
	TCN-GRU	310.9321	0.1746	25.6921	27.3357

Country	Model	RMSE	NRMSE	MAPE	SMAPE
Turkey	TCN	5516.897	1.6179	234.097	111.1507
	LSTM	6612.6704	1.9392	282.8772	120.4297
	GRU	6110.6825	1.792	261.4117	116.832
	TCN-LSTM	4323.4747	1.2679	181.4615	98.9834
	TCN-GRU	8551.817	2.5079	364.6515	131.2244

Country	Model	RMSE	NRMSE	MAPE	SMAPE
Japan	TCN	9849.1342	0.3236	23.1266	23.6299
	LSTM	9650.312	0.317	22.4706	23.1546
	GRU	9820.1018	0.3226	22.9084	23.4882
	TCN-LSTM	9860.8563	0.324	22.9724	23.6134
	TCN-GRU	8930.1258	0.2934	20.4223	21.6066

Country	Model	RMSE	NRMSE	MAPE	SMAPE
Vietnam	TCN	21005.0328	0.2916	145.5658	110.0516
	LSTM	23580.645	0.3273	157.0545	112.0843
	GRU	24829.6544	0.3447	172.7271	117.2366
	TCN-LSTM	20437.2901	0.2837	111.8296	82.8089
	TCN-GRU	19372.8376	0.2689	85.9936	64.7696

Figure 27. 30 days prediction indicator table

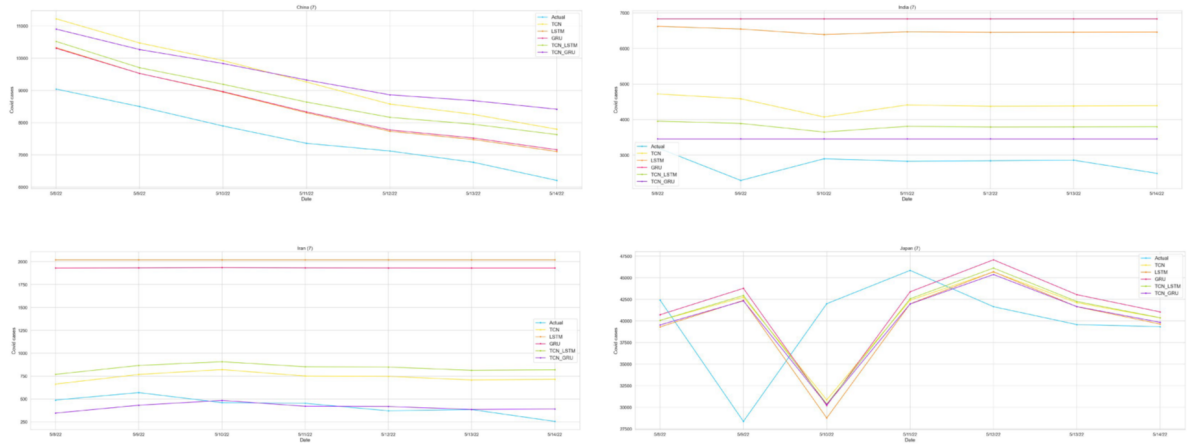

Figure 28. 7 days prediction results(China, India, Iran, Japan)

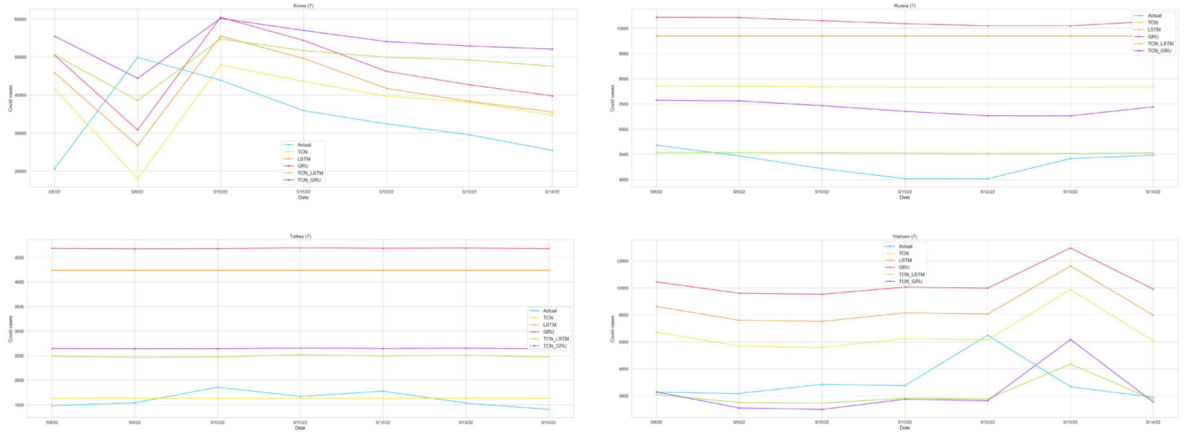

Figure 29. 7 days prediction results(Korea, Russia, Turkey, Vietnam)

Country	Model	RMSE	NRMSE	MAPE	SMAPE
China	TCN	1823.4214	0.6432	23.8769	21.3075
	LSTM	956.782	0.3375	12.3629	11.576
	GRU	975.0921	0.3439	12.6671	11.8859
	TCN-LSTM	1280.6461	0.4517	16.856	15.6998
	TCN-GRU	1920.7757	0.6775	25.3552	22.8364

Country	Model	RMSE	NRMSE	MAPE	SMAPE
India	TCN	1680.5311	1.8287	59.4444	46.0821
	LSTM	3723.68	4.0519	133.9575	80.4731
	GRU	4071.4082	4.4303	146.5293	84.8084
	TCN-LSTM	1080.1011	1.1753	37.5509	31.9966
	TCN-GRU	737.1739	0.8021	24.6495	22.3915

Country	Model	RMSE	NRMSE	MAPE	SMAPE
Iran	TCN	326.9341	1.0379	73.592	55.3618
	LSTM	1596.8573	5.0694	374.2149	130.8244
	GRU	1508.5715	4.7891	353.4642	128.2494
	TCN-LSTM	423.97	1.3459	97.1335	66.8137
	TCN-GRU	93.7736	0.2977	17.4585	18.3448

Country	Model	RMSE	NRMSE	MAPE	SMAPE
Japan	TCN	7294.3576	0.4169	13.939	14.6539
	LSTM	7721.5533	0.4413	14.5689	15.522
	GRU	7838.8651	0.448	15.0427	15.7069
	TCN-LSTM	7487.5021	0.428	14.3405	15.0732
	TCN-GRU	7306.1981	0.4176	13.8768	14.6735

Country	Model	RMSE	NRMSE	MAPE	SMAPE
Korea	TCN	15821.371	0.5401	37.825	38.0602
	LSTM	15878.8207	0.542	42.919	39.4555
	GRU	18680.1869	0.6377	52.733	45.6605
	TCN-LSTM	19195.177	0.6552	53.5649	45.8715
	TCN-GRU	22937.8532	0.783	62.878	50.802

Country	Model	RMSE	NRMSE	MAPE	SMAPE
Russia	TCN	3063.3893	2.293	65.0554	49.4515
	LSTM	5063.2452	3.7899	108.2783	70.5358
	GRU	5619.1408	4.2059	120.3877	75.4416
	TCN-LSTM	608.292	0.4553	10.3637	10.3064
	TCN-GRU	2208.3559	1.653	46.7928	38.2922

Country	Model	RMSE	NRMSE	MAPE	SMAPE
Turkey	TCN	154.1854	0.3419	8.6824	8.6375
	LSTM	2632.2831	5.8365	163.2373	90.0735
	GRU	3077.8477	6.8245	190.9556	97.8595
	TCN-LSTM	892.34	1.9786	54.6264	43.2371
	TCN-GRU	1047.1636	2.3219	64.361	49.0061

Country	Model	RMSE	NRMSE	MAPE	SMAPE
Vietnam	TCN	4163.7715	0.9044	122.7179	79.4067
	LSTM	5867.986	1.2745	182.2954	98.1431
	GRU	7587.6894	1.6481	243.4808	112.365
	TCN-LSTM	2017.2299	0.4381	46.1151	45.8017
	TCN-GRU	2439.6146	0.5299	60.0544	61.0339

Figure 30. 7 days prediction indicator table

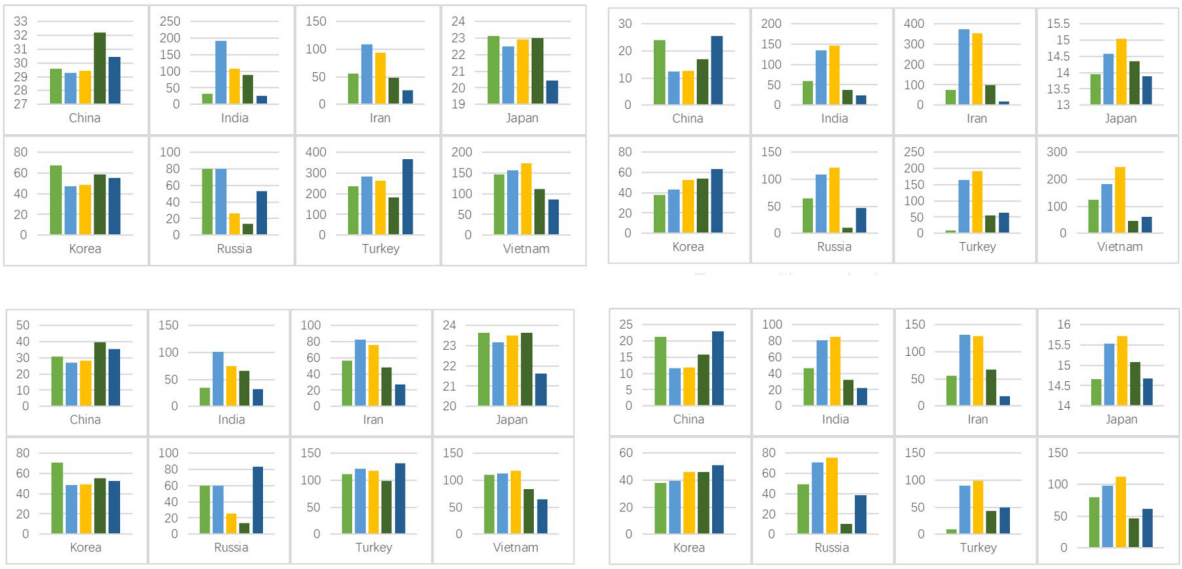

Figure 31. Top-left: RMSE (30 days); Top-right: RMSE (7 days); Bottom-left: NRMSE(30 days); Bottom-right: NRMSE(7 days) Left –Right in chart: TCN –LSTM –GRU –TCN combined with LSTM –TCN combined with GRU

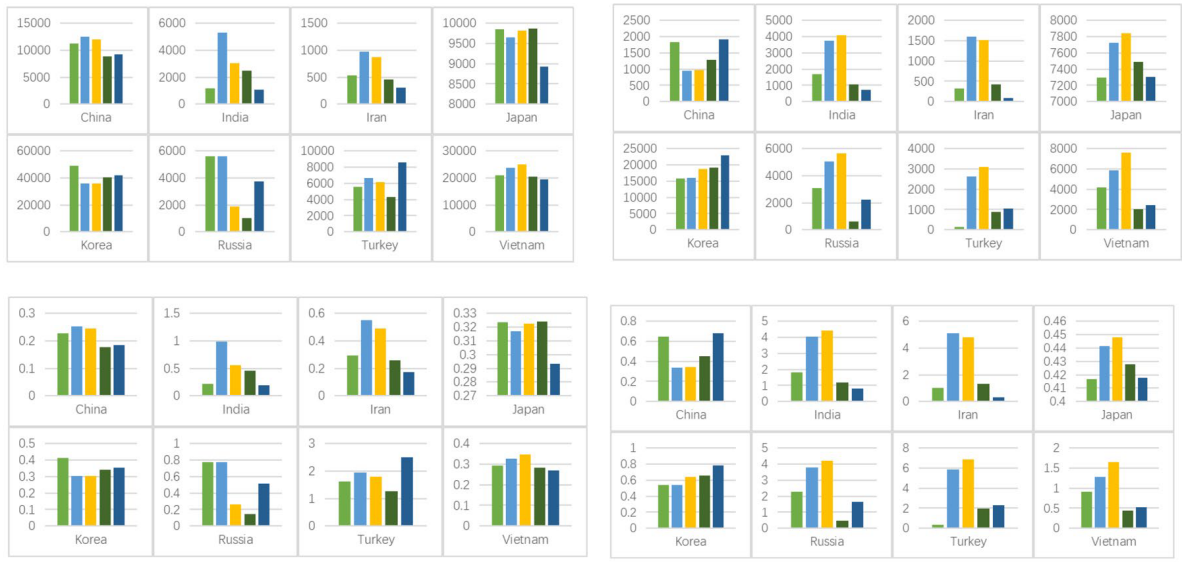

Figure 32. Top-left: MAPE(30 days); Top-right: MAPE(7 days); Bottom-left: SMAPE(30 days); Bottom-right: SMAPE(7 days) Left –Right in chart: TCN –LSTM –GRU –TCN combined with LSTM-TCN combined with GRU

Model	China	India	Iran	Japan	Korea	Russia	Turkey	Vietnam
TCN	27.4386	28.3418	32.5318	35.6615	27.3477	26.8681	42.2544	32.9170
LSTM	16.7928	15.4781	16.4926	16.3931	23.4229	15.9057	16.4240	39.6787
GRU	17.8688	17.0387	17.7213	17.3868	21.8618	18.9629	16.3195	38.6749
TCN-LSTM	29.7317	29.1681	30.4871	33.1216	30.5662	29.5052	33.0855	51.5300
TCN-GRU	35.0378	34.6820	34.0371	35.2461	54.3242	36.2617	34.5558	76.4242

Model	China	India	Iran	Japan	Korea	Russia	Turkey	Vietnam
TCN	0.3555	0.3203	0.4139	0.3944	0.3594	0.3249	0.4869	0.3607
LSTM	0.2213	0.2003	0.1930	0.1925	0.3117	0.1883	0.2033	0.5461
GRU	0.3309	0.3179	0.3137	0.3236	0.4177	0.3483	0.3160	0.8840
TCN-LSTM	0.4583	0.6287	0.4315	0.5202	0.4861	0.4255	0.5639	1.0255
TCN-GRU	0.6324	0.6150	0.5999	0.5925	0.7434	0.6372	0.5810	1.1960

Model	China	India	Iran	Japan	Korea	Russia	Turkey	Vietnam
TCN	35.2943	21.5192	26.7241	26.9162	25.4791	17.4714	22.9802	18.0345
LSTM	14.4262	12.6377	16.6775	16.9748	11.8299	11.9028	11.9628	11.9783
GRU	23.8433	11.9557	17.8347	17.2144	12.7449	12.3797	12.2981	13.2738
TCN-LSTM	25.2963	27.2453	31.0769	30.6976	22.7823	20.4871	20.6288	28.8232
TCN-GRU	27.5292	25.4552	34.3477	31.9891	28.5521	22.9976	22.8127	25.3623

Model	China	India	Iran	Japan	Korea	Russia	Turkey	Vietnam
TCN	0.7086	0.3827	0.4591	0.5230	0.4525	0.2859	0.4385	0.3199
LSTM	0.3129	0.2312	0.3136	0.2751	0.1762	0.1953	0.2063	0.2154
GRU	0.3199	0.3358	0.3990	0.4614	0.3051	0.3191	0.3189	0.3299
TCN-LSTM	0.5930	0.5909	0.5944	0.6564	0.5106	0.4774	0.4514	0.5902
TCN-GRU	0.6935	0.6578	1.2069	0.7283	0.6478	0.6548	0.5864	0.7225

Figure 33. Top-bottom: Training time (30 days); Prediction time (30 days); Training time (7 days); Prediction time (7 days) ; unit: seconds

5.5 Findings

From the overall prediction results, TCN-LSTM and TCN-GRU are better than TCN, and TCN is better than LSTM and GRU.

In long-term forecasting, the predictive ability of TCN and improved model is mostly better than LSTM and GRU. In China (30 days), India (30 days), Iran (30 days) and Vietnam (30 days), TCN and its corresponding combination model are better than GRU and LSTM. Among them, in the predictions of China (30 days), Iran (30 days) and Vietnam (30 days), the modified combination model is better than the original TCN. In the Japan (30-day) forecast, TCN-GRU stands out the most. In South Korea (30 days) forecast, the performance is similar, but the performance of TCN is the worst. The predictions of Russia (30 days) and Turkey (30 days) are abnormal. The former GRU and TCN-LSTM perform outstandingly, while TCN and LSTM perform similarly poorly.

In short-term forecasting, the forecasting ability of TCN and improved model is more prominent. In the forecasts of India (7 days), Iran (7 days), Russia (7 days) and Vietnam (7 days), the two improved models outperformed the original TCN. While in Japan (7 days) and Turkey (7 days) forecasts, TCN outperforms the two improved models. In China (7 days) and South Korea (7 days) forecasts, TCN and its improved models do not perform well. In the prediction of South Korea (7

days), the models performed similarly, while in the prediction of China (7 days), the prediction ability of LSTM and GRU was even better than that of TCN and its corresponding transformation model.

5.6 Limitations

The input data of this experiment is one-dimensional data. Considering that there are various factors affecting the cases of Covid data, including the epidemic prevention policies of various countries and the emergence of mutated pathogens, multi-dimensional data input can be considered for Covid prediction. After using multi-dimensional data, the attention mechanism and self-attention mechanism can be used for the network model, which can denoise the multi-dimensional data, focus on the data with prominent features and heavy weights, and the trained model will be more accurate.

5.7 Alternative Improvements

The Knowledge-Driven Temporal Convolutional Network (KDTCN) model can be considered. KDTCN addresses two shortcomings of temporal convolutional networks. The first disadvantage is that the temporal convolutional network cannot capture the impact of sudden factors on the number of cases, and is not sensitive enough to the sudden change in the number of cases. For example, the emergence of the Covid variants Delta and Omicron has greatly increased the number of cases. The second shortcoming is that the temporal convolutional network cannot simply analyze the inflection point of the increase and decrease of the number of cases from a simple mathematical level, making the prediction results uninterpretable. KDTCN can extract the factors affecting the Covid as a factor group, and then record the entities and relationships of the factors in the factor group through the event knowledge graph. Then, vectorize and concatenate the structured knowledge graph and influencing factors such as virus mutation and policy change, and finally embed the feedback into the temporal convolutional network.

6. Potential Impact

(1) The author analyzes the research background of Covid case prediction, discusses the research significance of Covid prediction, and investigates the existing models for predicting Covid cases. Long-short-term memory network model, and gated recurrent unit network model are briefly explained, and the superiority of temporal convolutional networks over recurrent neural networks is expounded.

(2) The author elaborates on the composition and deduction of the time convolutional network in detail, and gives the corresponding formula. The most important concepts include one-dimensional convolutional dilation causal network and residual block, and the corresponding system structure is designed. . The original model was modified and combined, including the network of time convolution network and long short-term memory combination, the network of time convolution network and gated recurrent unit combination, and the corresponding system structure was designed.

(3) The author downloaded and preprocessed the Covid data in real time. The data sets came from 8 countries, namely China (Mainland), India, Iran, Japan, South Korea, Turkey, Russia and Vietnam. Preprocessing included first-order difference and normalization, and the characteristics of the original data were analyzed, especially including kurtosis and skewness.

(4) The author designed 5 corresponding prediction models according to the corresponding system structure, including temporal convolutional network model, long-term short-term memory network model, gated recurrent unit model, combined model of temporal convolutional network and long-term short-term memory network, A combined model of temporal convolutional networks and gated recurrent unit networks. The corresponding network is networked, and the Adam optimizer is used to set the same hyperparameters.

(5) Carry out model training and prediction, divided into two sub-experiments. The first sub-experiment training set has 813 elements and predicts 30 days of data. The second sub-experiment training set has 435 elements and predicts 7 days of data. The evaluation indicators used include root mean square error, standard root mean square error, mean absolute percentage error and symmetric mean absolute percentage error.

(6) The corresponding experimental results were drawn and compared. From the overall prediction results, the combined network of temporal convolutional network and other models is superior to temporal convolutional network, and temporal convolutional network is superior to long-term short-term memory network and Gated recurrent unit network.

7. Reference

[1] Sina F. Ardabili,Amir Mosavi,Pedram Ghamisi,Filip Ferdinand,Annamaria R.Varkonyi-Koczy,Uwe Reuter,Timon Rabczuk,Peter M. Atkinson;COVID-19 Outbreak Prediction with Machine Learning; medRxiv and bioRxiv; April 22, 2020.

[2] Yifan Yang,Wenwu Yu,Duxin Chen; Prediction of COVID-19 spread via LSTM and the deterministic SEIR model; 2020 39th Chinese Control Conference (CCC).

[3] K.E.ArunKumar,Dinesh V.Kalaga,Ch.Mohan Sai Kumar,MasahiroKawaji,Timothy M.Brenza; Comparative analysis of Gated Recurrent Units (GRU), long Short-Term memory (LSTM) cells, autoregressive Integrated moving average (ARIMA), seasonal autoregressive Integrated moving average (SARIMA) for forecasting COVID-19 trends; Alexandria Engineering Journal Volume 61, Issue 10, October 2022, Pages 7585-7603.

[4] Domenico Benvenutoa1,Marta Giovanetti, Lazzaro Vassalloc, Silvia Angeletti, MassimoCiccozzi. Application of the ARIMA model on the COVID-2019 epidemic dataset.

[5] Farah Shahid,Aneela Zameer,Muhammad Muneeb; Predictions for COVID-19 with deep learning models of LSTM, GRU and Bi-LSTM; Chaos, Solitons & Fractals Volume 140, November 2020, 110212.

[6] Jia-Ji Wang,Chen Wang,Jian-Sheng Fan,Y.L.Mo. A deep learning framework for constitutive modeling based on temporal convolutional network;Journal of Computational Physics Volume 449, 15 January 2022, 110784.

[7] Yongjian Wang,Ke Yang,Hongguang Li; Industrial time-series modeling via adapted receptive field temporal convolution networks integrating regularly updated multi-region operations based on PCA; Chemical Engineering Science Volume 228, 31 December 2020, 115956.

[8] Yang Lin,Irena Koprinska,Mashud Rana; Temporal Convolutional Neural Networks for Solar Power Forecasting; IEEE, 19-24 July 2020.

[9] Deepak Kumar Sharma,Shikha Brahmachari,Kartik Singhal,Deepak Gupta. Data driven predictive maintenance applications for industrial systems with temporal convolutional networks; Computers & Industrial Engineering Volume 169, July 2022, 108213.

[10] Pedro Lara-Benítez,Manuel Carranza-García,José M.Luna-Romera José C. Riquelme; Temporal Convolutional Networks Applied to Energy-Related Time Series Forecasting; Applied Sciences Volume 10 Issue 7, 28 March 2020.

[11] Saroj Gopali,Faranak Abri;A Comparison of TCN and LSTM Models in Detecting Anomalies in Time Series Data;IEEE,2021.

[12] Mahmoud M. Bassiouni,Islam Hegazy,Nouhad Rizk,El-Sayed A.El-Dahshan;Automated Detection of COVID-19 Using Deep Learning Approaches with Paper-Based ECG Reports; Circuits, Systems, and Signal Processing (2022).

[13] Olof Almqvist;A comparative study between algorithms for time series forecasting on customer prediction: An investigation into the performance of ARIMA, RNN, LSTM, TCN and HMM; June 2019.