U-Net-based Precipitation Predict by Cloud Map

Tianyu Chen *

Department of Computer Science, University of Bristol, Bristol, United Kingdom

* Corresponding author email: uv21031@bristol.ac.uk

Abstract. Many traditional precipitation prediction methods in meteorology nowadays require many types of data to be input as parameters. This research is to investigate ways to use deep learning techniques for precipitation prediction using only input cloud maps. This paper establishes a technical route for predicting rainfall through cloud map data using U-Net, and experiments. Rainfall models were successfully trained using U-Net and predicted.

Keywords: Precipitation Prediction; U-net; Deep Learning.

1. Introduction

In current climate prediction, such as precipitation prediction, precipitation forecasting, as part of meteorological forecasting, is the act of predicting the amount of precipitation that will fall over a specified area over a specified period in the future, using a series of available measurements [1-4]. There are quite a few methods that use lot of different kind of data preparation for determine or calculate the result of prediction. If there are way to prediction precipitation only by few types of input, such as cloud map and time in day, it will be meaningful. Therefore, deep learning will be a good technical route to implement this method [5]. Moreover, benefit from computer powerful calculation performance, huge parament and part of black box effect, deep learning has some advantage, for example, low manpower of build model for product and fast prediction [6]. There are many predictions time long of deep learning program just take low than 1 second.

At present, there are few articles mention a full process of how to use deep learning to predict an exact climate aim. However, to be sure, there is a good way to implement precipitation prediction that is use image to image neural network, and the famous image to image network structure is U-Net, which is originally used as a network structure for image segmentation in the medical field.

Therefore, this research uses a typically U-Net structure which based on PyTorch platform to build a precipitation prediction model system, in global area [7]. In the process of building a precipitation prediction program using U-Net, it was found that U-Net showed very excellent adaptability [8]. It was able to approach the target quickly despite the small amount of data trained. Although the resolution of the resulting images presented by U-Net is reduced compared to the rainfall images provided during training, the amount of data used is very small, spanning a period of only two years, even at a frequency of every three hours, so that only nearly 2900 data are available for training. Therefore, if the time span of the data is extended to 10 or even 30 years, U-Net will show very good performance in precipitation prediction. This is especially the case when only images and possibly a small amount of data is input.

2. Data

The dataset used in this project was downloaded from the European Copernicus Weather Data Site (CDS). The dataset used is "ERA5 hourly data on single levels from 1959 to present". The dataset covers all meteorological data collated on a global scale from 1959 to the present, with a latitude and longitude resolution of 361°x720°. The data used in this project are 'Total precipitation' as precipitation data and 'Total cloud cover' as global cloud maps. The individual time interval between these two data is three hours. The precipitation data is in millimeters and the cloud maps are pre-processed data, divided into 10 cloud density classes from 0-10. However, this is not enough to feed directly into the neural network for calculation. Therefore, the data needs to be processed again in the

project. As "Total precipitation" is the training data for the output of the network, the precipitation level of this array of shape (361,720,1) needs to be cut from 0 (mm) to X (mm), setting the number of cuts L. The larger the L, the higher the accuracy. The reason for setting an upper precipitation limit is to prevent a small number of outliers from interfering with the overall precipitation prediction, as preliminary analysis of the dataset shows that the frequency plot of 'Total precipitation' shows that most of the data falls in the 0-20mm range, but many times the maximum precipitation falls in the 0-20mm range. However, in many cases the maximum precipitation can be as high as 40mm, and the frequency of data in the 30-40mm range was found to be less than 1% of the total. However, this does not mean that data above x cannot be predicted, and a separate model can be trained for the higher intervals.

3. Method

The tone of this project was set at the outset to use deep learning approaches for rainfall prediction on a global scale. Therefore, the mathematical modelling approach of classical meteorology and other machine learning methods are not considered here. Therefore, deep learning for global rainfall prediction is discussed here. The need demonstrated by the goal of rainfall prediction is not quite the same as the way neural network models usually work, because initially neural networks perform a simple data classification job, which is always in essence simple classification, no matter how much the application goal changes. Very often the output size is smaller than the input size. Rainfall prediction, however, cannot be as simple as directly inputting a cloud map with a specified GPS latitude and longitude and then outputting the amount of rainfall at a given place. This is because if the model is designed as described above, and regardless of whether the optimizer is able to achieve gradient descent, the dataset will grow exponentially. Therefore, using cloud maps to predict rainfall on a global scale must require a neural network structure from image to image. Setting the tone of image-to-image rules out most classical neural network structures, such as simple convolution followed by unfolding, or downscaling images to 1 dimension to violently shoehorn in a purely fully connected network, which will not work. The obvious choice is to use codec structured networks, of which the most famous is U-Net. Although U-Net is best known for its excellent performance in medical image segmentation, this does not in any way affect its application to other fields. In a sense, the precipitation prediction process, with its graded precipitation levels, is not a pixel-level image segmentation.

Further, another reason for using U-Net is that U-Net extracts feature values at each down-sampling, which are then combined and computed during subsequent up-sampling at the same level (see figure). In contrast, in the process of predicting rainfall through cloud maps, it is inevitable that a large amount of rainfall data from adjacent, or even alternate adjacent, locations will need to be calculated, and these features will be extracted by the convolution process throughout the U-Net, right up to the same-depth up sampling layer. Also, during the multiple down sampling, the overall precipitation trend is sampled as a set of arrays containing the global rainfall trend and is provided as a 'seed' effect to the up-sampling process during the up-sampling process.

Here's how the U-Net is trained. Firstly, as mentioned above, the precipitation data in the training set is split into a specified X number of precipitation levels, which is set to 20 and the maximum precipitation to 30 mm, although these two parameters can be set to other values, so for the sake of example we will set them as such. Then, the output shape of U-Net is set to (20, 361, 720), which means 20 channels, and the Index of each channel is the corresponding rainfall, so when processing, only the value of the corresponding channel needs to be set to 1, and the rest to 0. However, for the model's predicted output, the shape of the output array is obviously also the same as the one at the time of training (20, 361, This can be transformed to (1, 361,720) using PyTorch's own softmax function, and the corresponding values will be normalized to 0-1 [9].

4. Experiment

The training environment used was a PyTorch 1.10.2 CUDA 11.3 based platform running under Windows with i9-11900K @ 3.50GHz 32.0 GB RAM hardware and a CUDA accelerated device with 24GB of RTX3090 video memory.

The training was performed using a 5-layer down sampling/5-layer up sampling U-Net network, and the training data was limited to 2 years from 2016/01/01-2018/01/01/01, with the validation set from 2019/01/01-2019/03/01. 1959 to present" data set downloaded from CDS. The optimizer was Adam, the loss rate was calculated using "CrossEntropyLoss", and the accuracy was calculated using an algorithm developed in-house.

In the training process, the batch_size was set to 8, taking into account the limitation of the video memory, which can be increased if the platform has a larger video memory. 730 sets of 8 data were used in each training round, and 1 set of 544 validation sets. Looking at the training process in terms of loss, we can see that when the model is trained with almost 1000 sets of data, the loss rate decreases from 3 at the beginning to 1, and then decreases to less than 0.5 at about 1700 sets, and the whole process loss rate starts with

$$f(x) = \frac{a}{x} \tag{1}$$

The whole process of loss rate decreases with a regular pattern of f(x) = a/x. Meanwhile, accuracy increased from 0% to 94% in a very short period, roughly 300 data sets, with the data growth following the following pattern

$$f(x) = 1 + a\log_c(bx)) \tag{2}$$

This is followed by a period of plateauing, and at roughly 2000 sets begins to continue its smooth, near-linear climb upwards at a rate equal to approximately 0.1% per 8000 data sets trained.

In this way, the results of the training look good overall.

As shown in figure 1, the effect after 730 sets of data is shown on the left, and the effect after 29200 sets of data is shown on the right. The first row shows the cloud photo, the second row shows the predicted result graph, the third row shows the real data graph, the second/third row shows the same time, and the first row shows the cloud graph 3 hours ago.

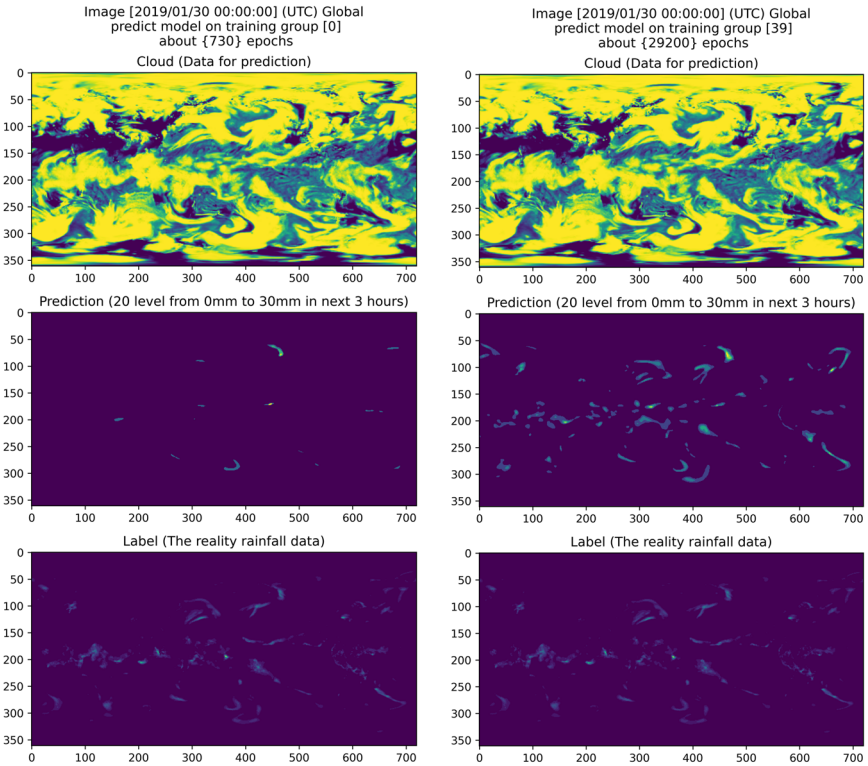

Fig 1. Precipitation Prediction

Obviously, we can see that the deep learning training has worked well, and the rainfall predictions have covered the corresponding real rain areas almost perfectly. It is only slightly inferior to the original data in terms of pixel resolution. However, this does not mean that U-Net will stop at this point for weather prediction, since it took 2 years of data and only 40 rounds of training. In addition, the slope of the correct rate did not see a significant decrease at 40 rounds, which means that more training will result in a finer resolution of the model, and more training data will result in more accurate predictions from the trained model. This is because although the project only used 2 years of data given the performance of the testbed, the original dataset had data from 1950 to the present. If all the data were used for training, it would make the model predictions more accurate.

5. Timing Enhancement

On top of the above research practices, this study also identified another way in which the prediction effect could be enhanced [10]. This is the inclusion of time series data. The reason why the leading time-series data is fed into the model is that the trend of cloud cover also contains some valid information. This is done by using the channel parameter of the U-Net convolutional layer, which is the most basic and easiest to implement. By modifying the value of the channel, the temporal width of the input cloud data is increased, thus achieving the effect of input with temporal data. First, the length of the temporal data forward is x. The shape of the input data will therefore be (x, 361,720).

In the course of the actual experiments, the training performance of the cloud map data set with the input of the temporalized data outperformed that of the data set without the inclusion of the temporalized data. The temporal order used in the experiments included a length of 3 in the forward direction, so the input shape was (3, 361,720). As shown in figure 2, the first graph is the accuracy data at training time and the second graph is the loss rate. The blue line is the training process with the temporalized data added, and the red line is the training process without the temporalized data.

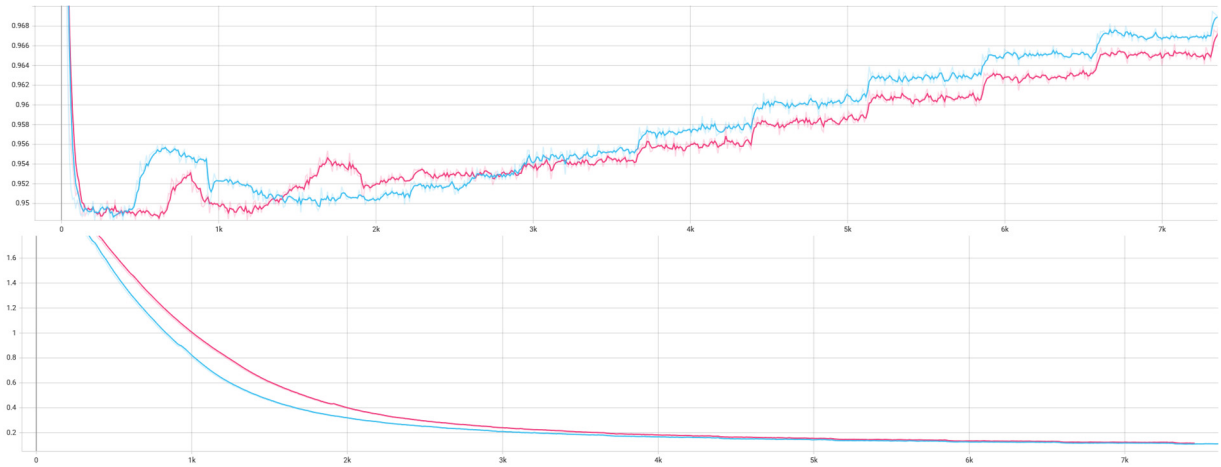

Fig 2. Training Process.

Therefore, we can clearly see that for the accuracy data, the accuracy of the training set grows faster than the training set without temporal data in general. Looking at the loss rate, the loss rate icon clearly shows that the loss rate of the temporalized data set decreases faster than that of the data set without temporalization.

Furthermore, this experiment only used a forward inclusion length of 3, meaning that only 3 sets of temporalized cloud data were input. Therefore, the effect would have been more significant if the training had used a larger time scale as the temporal inclusion length.

In terms of improving the prediction with temporal data, the following two graphs should be compared, the left graph shows the prediction results of the model with 6570 sets of temporalized data and the right graph shows the prediction results of the model without 6570 sets of temporalized data

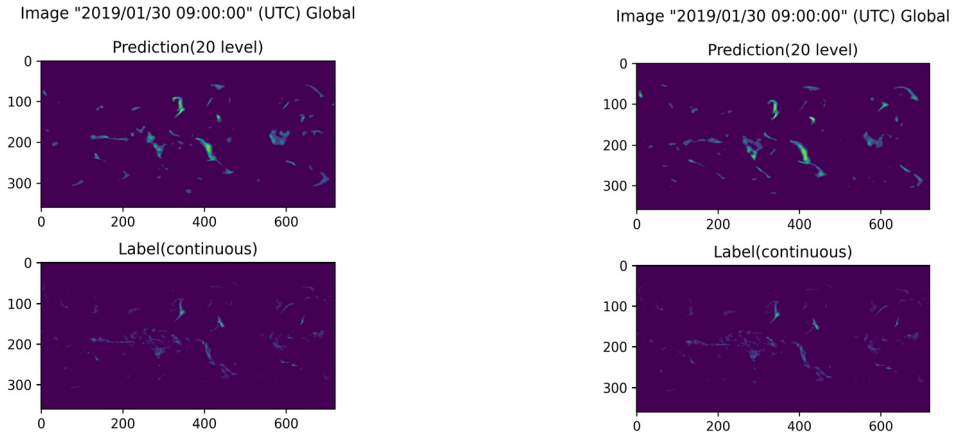

Fig 3. Results Comparison.

By comparing these two images in figure 3, it is clear that the model trained on the temporalized dataset performs better than the model trained on the dataset without temporalized data for the same number of training sets. For example, it is more coherent in terms of detail and has more correct predictions for parts of the rainfall (in the graph it is the depth of the color). The results of the no temporalization model sometimes have the problem of predicting precipitation for a small part of the area without precipitation than the temporalization dataset model, but the timed dataset model avoids this problem. This problem is avoided by the time-series model, for example at (700,90), (200,200), (600,100) and so on.

6. Summary

In general, the technical approach of using U-Net for precipitation prediction is feasible and effective. The results of the training experiments show that this approach has excellent performance in terms of adaptation to small data sets, temporal enhancement and so on. For the future of this technique, the viable route ahead is clear. The time span of the training dataset could be increased, to 10 or even 20 years. And the temporal enhancement part could be enhanced, for example by not using the channel parameters of the U-Net convolutional layer directly, but by adding a direct LSTM or GRU layer to enhance the temporal relationships between the temporal data. This could certainly make U-Net's effect on rainfall prediction twice as effective and further enhanced. In addition, another possible application scenario for this technology could be the future prediction of precipitation for precipitable planets among the extraterrestrial planets. Finally, deep learning techniques will offer more feasibility in the field of weather prediction in the future.

References

[1] De Andrade F M, Young M P, MacLeod D, et al. Subseasonal precipitation prediction for Africa: Forecast evaluation and sources of predictability[J]. Weather and Forecasting, 2021, 36(1): 265-284.

[2] Kanavos A, Trigka M, Dritsas E, et al. A regularization-based big data framework for winter precipitation forecasting on streaming data[J]. Electronics, 2021, 10(16): 1872.

[3] Coban V, Guler E, Kilic T, et al. Precipitation forecasting in Marmara region of Turkey[J]. Arabian Journal of Geosciences, 2021, 14(2): 1-10.

[4] Li W, Gao X, Hao Z, et al. Using deep learning for precipitation forecasting based on spatio-temporal information: a case study[J]. Climate Dynamics, 2022, 58(1): 443-457.

[5] Dong S, Wang P, Abbas K. A survey on deep learning and its applications[J]. Computer Science Review, 2021, 40: 100379.

[6] Liu J, Guo X, Yuan Y. Prototypical Interaction Graph for Unsupervised Domain Adaptation in Surgical Instrument Segmentation[C]//International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer, Cham, 2021: 272-281.

[7] Ronneberger O, Fischer P, Brox T. U-net: Convolutional networks for biomedical image segmentation [C]//International Conference on Medical image computing and computer-assisted intervention. Springer, Cham, 2015: 234-241.

[8] Huang H, Lin L, Tong R, et al. Unet 3+: A full-scale connected unet for medical image segmentation[C]// ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2020: 1055-1059.

[9] Paszke A, Gross S, Massa F, et al. Pytorch: An imperative style, high-performance deep learning library[J]. Advances in neural information processing systems, 2019, 32.

[10] Wauchope H S, Amano T, Geldmann J, et al. Evaluating impact using time-series data[J]. Trends in Ecology & Evolution, 2021, 36(3): 196-205.