Deep Neural Networks with Residual Connections for Precipitation Forecasting

Mao Nguyen*
VNG Corporation
John von Neumann Institute, VNU-HCM
Ho Chi Minh, Vietnam
nguyenxuanmao@gmail.com

Phu Nguyen
Neolab Vietnam
Da Nang, Vietnam
anphunl@gmail.com

Thu Vo
VNG Corporation
Ho Chi Minh, Vietnam
vo.tr.thu@gmail.com

Lam Hoang
Trusting Social
Ho Chi Minh, Vietnam
hoangvietlambk@gmail.com

ABSTRACT

This paper describes our deep learning system for precipitation forecasting. The main contribution of this work is to use recent architecture likes convolutional long short-term memory, residual networks to represent the spatial and temporal nature of the precipitation data. In experiments, we show that this model is easier to optimize and efficient in performance. We also propose some regularization techniques to deal with the over-fitting issue in this problem: One comes from the level of rainfall magnitude categorization, the other comes from the probabilistic labeling. We apply this model to CIKM AnalytiCup 2017 and archive a comparable result (rank 3/1395 on the first Season).

CCS CONCEPTS

• **Computing methodologies** → **Computer vision representations**; **Machine learning approaches**; **Neural networks**; *Regularization*;

KEYWORDS

CIKM 2017 proceedings, Rainfall Forecasting, Deep Residual Network, Convolutional Long Short-term Memory

ACM Reference format:
Mao Nguyen, Phu Nguyen, Thu Vo, and Lam Hoang. 2017. Deep Neural Networks with Residual Connections for Precipitation Forecasting. In *Proceedings of ACM International Conference on Information and Knowledge Management, Singapore, Nov 2017 (CIKM 2017),* 4 pages.
https://doi.org/10.475/123_4

*Mao Nguyen insisted his name be first.

CIKM 2017, Nov 2017, Singapore

ACM ISBN 123-4567-24-567/08/06.
https://doi.org/10.475/123_4

1 INTRODUCTION

Short-term precipitation now-casting such as rainfall prediction is a task to predict a short-term rainfall amount based on current observations. It has long been an important problem in the field of weather forecasting. The goal of this task is to give precise and timely prediction of rainfall intensity in a local region over a relatively short period.

In recent researches, some computer vision techniques, such as optical flow based methods, have proven useful for making accurate extrapolation of radar maps [1, 2] Especially, recent advances in deep learning lead to better representation in feature space and more efficient prediction model. In [8], the convolutional kernels in the dynamic convolutional layer are determined by a neural network encoding the information of weather images in previous time step. Xingjian Shi et al. combine the Convolutional Network (CNN) and the Long Short-term Memory to build an Encoding-Forecasting structure that have the ability to model the spatiotemporal sequence in the rainfall prediction problem. Inspire by these works, we use the residual connections [4, 5] (which is big advance of deep leaerning in 2016) to extend the Convlutional LSTM and build efficient model for precipitation predition. We apply this model on CIKM AnalytiCup 2017 challenge and archive a good score of MSE - 12.94 (rank 3/1395) on the first Season.

2 PROBLEM FORMULATION

According to the CIKM AnalytiCup 2017, the provided dataset is a set of radar maps at different time spans where each radar map covers radar reflectivity of a target site and its surrounding areas. In detail, radar maps contain the reflectivity of signal on following dimensions:

(a) Each radar map contains one target site that located at the centre of the map covering an area of $101 \times 101 km^2$ around the site.
(b) Radar maps are measured at different time spans, i.e., 15 time spans with an interval of 6 minutes, and different heights, i.e., 4 heights, from 0.5km to 3.5km with an interval of 1km

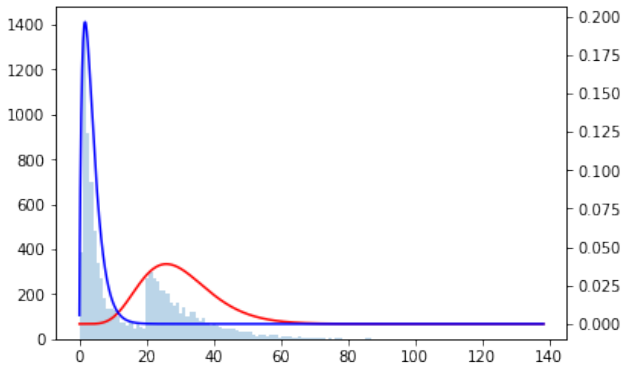


Figure 1: The distribution of rainfall

Our task here is to predict the total rainfall amount on the ground between future 1-hour and 2-hour for each target site.

2.1 Loss function

From machine learning perspective, this problem can be regarded as a spatio-temporal sequence forecasting problem. Suppose we observe a dynamical system over a spatial region represented by an $M \times N$ grid which consists of M rows and N columns. Inside each cell in the grid, there are $T \times H$ measures, where T is the number of time-spans and H is the number of height level. Thus, the observation at any time can be represented by a tensor $X \in \mathbb{R}^{T \times H \times M \times N}$, where $\mathbb{R}$ denotes the domain of the observed features. In CIKM AnalytiCup 2017 challenge, the each sample of data has the size $T = 15$, $H = 4$, $M = 101$, $N = 101$. The rainfall forecasting problem can be consider as a task of building a function $\mathcal{F} : R^{T \times H \times M \times N} \mapsto \mathbb{R}$ that minimizes the mean of square error over S samples $[X_1, .., X_S]$ in the traing set:

$$J = \sum_{i=1}^{S} ||t_i - \mathcal{F}(X_i)||_2 \quad (1)$$

2.2 Regularization

If we use a deep Neural Network to approximate $\mathcal{F}$, its parameters can be estimated after solving a minimization problem with the loss function stated in equation 1. However, this approach is quite naive, because it does not take into account the fact that the rainfall does not have an uniformly distribution. Moreover, the rainfall spread over a large range of values, so without any suitable regularization, the regression process will easily lead to an over-fitting.

From the distribution of rainfall on training set (which is illustrated in Fig. 1), we categorize the rainfall value into C levels of magnitude and define an additional cost to penalty the error when miss-classify a sample.

$$J = \sum_{i=1}^{S} ||t_i - \mathcal{F}(X_i)||_2 + \lambda \mathcal{L}(\mathcal{G}(X_i), c_i) \quad (2)$$

Similar to $\mathcal{F}$, $\mathcal{G} : R^{T \times H \times M \times N} \mapsto \mathbb{N}$ stands for the classification function and $\mathcal{L}$ is the cross-entropy measuring the difference between the output of classifier $\mathcal{G}$ and the real level of rainfalls $\{c_i\}$.

Another way of regularizing is instead of hard-classify the rainfall to magnitude level, we use a probabilistic labeling by model the rainfall as a mixture of gamma function. We firstly estimate the parameters of this function (illustrate by the red and blue curves in Fig. 1) and after that, for each time of training, the class value c_i of a sample is indicated by sampling the index of a gamma functions in the mixture model. The classification function $\mathcal{G}$ is still remained the same, but the ground-truth label is dynamically changed respect to the probability of the rainfall value.

3 OUR MODELS

In this section, we propose a deep neural network for precipitation forecasting. The network is motivated from the state-of-the-art ResNet [4, 5] and the Convolutional Long Short-term Memory [16] which has capable of modeling the spatial and temporal nature of radar maps properly.

3.1 Framework Pipeline

Fig. 2 describes the overview of our architecture with essential components and the flow of this framework. The precipitation forecasting function $\mathcal{F}$ and the rainfall classification function $\mathcal{G}$ are approximated by over deep neural network over pipeline of three main steps:

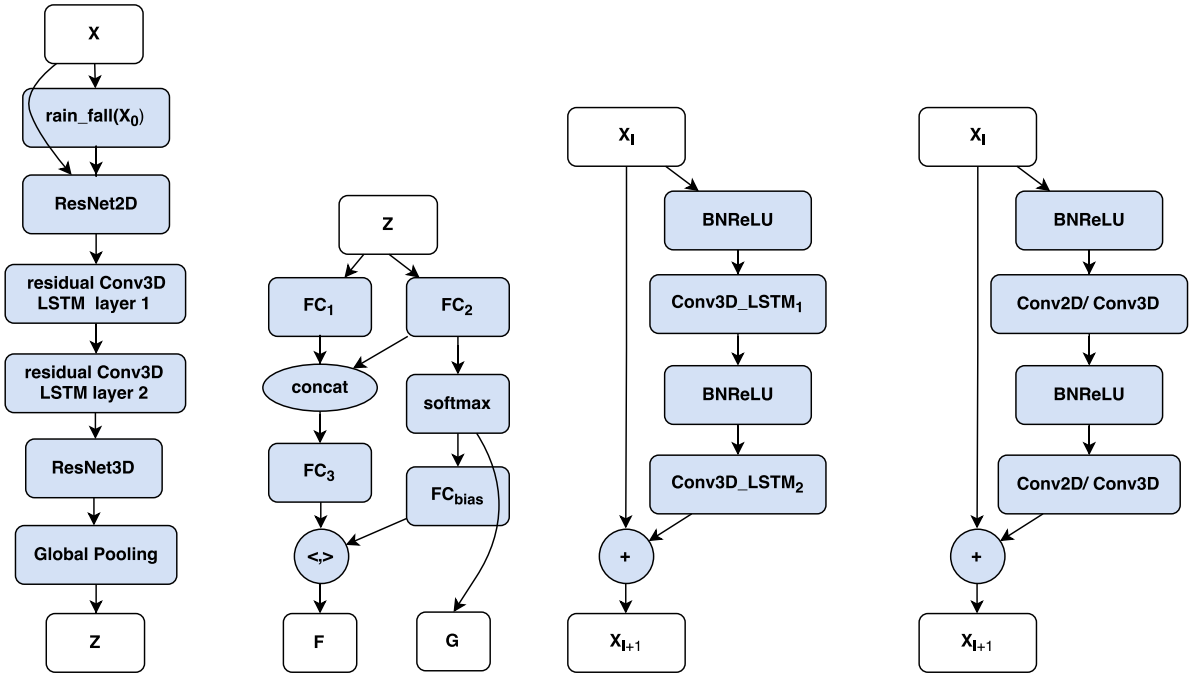
(1) Capture the 2D spatial information from radar maps X by passing it into a 2D residual network. The purpose of this block is to learn low level features (like interesting points, edges...) of the reflective signal

(2) Model the spatial-temporal nature of the data. Output features from step 1 are re-arranged to form a sequence in which each element is 3D cube merged from the 2D features over H level of heights. After that, a Convolutional LSTM in 3D with residual connections is used to not only capture the latent information of this sequence but also improve the optimization process. We will explain this block in more details in Section 3.3.

(3) Aggregate feature maps of the sequence of latent variables return from Step 2. In this block, we re-arrange the two dimensions of time-span and height-level into a new one to form a tensor in $\mathbb{R}^{TH \times M \times N}$ space. This cube is passed through a residual convolutional neural network (ResNet3D), a Global Pooling layer [10] and a final Fully Connected Unit (as described in Fig. 2b) to approximate the two both $\mathcal{F}$ and $\mathcal{G}$ functions.

3.2 Residual Convolutional Neural Network

Traditional convolutional feed-forward networks connect the output of the l^{th} layer as input to the $l + 1^{th}$layer [9], which leads to the following layer transition: $x_{l+1} = \mathcal{H}(x_l)$. This architecture has an issue that when the networks become "very



(a) Main Architecture **(b) Fully Connected Unit** **(c) Residual Conv3D LSTM Unit** **(d) Ressidual Conv Unit**

Figure 2: Our framework overview.

deep", accuracy gets saturated and then degrades rapidly. Kaiming He et al. proposed a new scheme so-called Deep Residual Network (ResNet) [4] to overcome this problem. ResNets add a skip-connection that bypasses the non-linear transformations with an identity function:

$$x_{l+1} = \mathcal{H}(x_l) + x_l \tag{3}$$

The advantage of ResNet is that the skip-connection make these residual network easier to optimize and reduce the training error. Motivated by [5] we define the skip-connection $\mathcal{H}$ as a composite function of three continuing operations: batch normalization (BN) [7], followed by a rectified linear unit (ReLU) [3] and a 2D or 3d convolution (Conv). As illustrated in Fig. 2d, we use two skip-connections, to form a Residual Convolution Unit. A Residual Convolutional Neural Network (ResNet) is the set of many Residual Convolution Units, one after each other.

3.3 Convolutional LSTM with residual connections

For general purpose sequence modeling, Long Short-term Memory (LSTM) [6] as a special recurrent neural networks (RNN) structure has proven stable and powerful for modeling long-range dependencies in various previous researches [12, 13]

The major drawback of the traditional LSTM in handling spatio-temporal data (like sequence of radar maps) is its usage of full connections in input-to-state and state-to-state transitions in which spatial information is disregarded. To overcome this problem, in [16] Xingjian et al. use convolution operator to compute input, output and forget gates as well as other cells instead of the original fully connected matrix multiplication. The model offers a faster and more accurate representation this kind of data.

Residual Connections: In many researches [14, 15], deep stacked LSTMs often give better accuracy over shallower models. However, simply stacking more layers of LSTM works only to a certain number of layers, beyond which the network becomes too slow and difficult to train, likely due to exploding and vanishing gradient problems [11, 12]. In our network, we add residual connections between two Conv LSTMs in a stack to form a Residual Conv3D LSTM Unit. The structure of this unit is described at Fig. 2c

3.4 Fully Connected Unit

The purpose of this Unit is to approximate the two functions $\mathcal{F}$ and $\mathcal{G}$ by passing the feature maps z to different fully connected layers (FC). As described in Fig. 2b, the right side are layers designed to learn the classification function like FC_2 or to supply the information of predicted rainfall magnitude for regression task like FC_{bias}. Remained layers like FC_1, FC_3 combine the above information and the spatial-temporal features map to give a final result of rainfall by computing their inner product. In experiments, we will show that the scheme of these fully connected layers not only make

Table 1: Result over different type of regularization

	# iters to converge	Phase 1 RMSE	Phase 2 RMSE
Weight decay only	failed		
Weight decay + BN	1400	13.21	
Weight decay + BN + rainfall magnitude	5400	**12.64**	**12.94**
Weight decay + BN + probabilistic labeling	4000	12.68	13.09

the training process faster to converge but also naturally lead to an over-fitting avoidance.

4 EXPERIMENTAL RESULTS

4.1 Data and Setup

In CIKM AnalytiCup 2017 challenge, the training dataset contain 10000 samples of radar maps. We use all of them to train our model, for any signal has a value of -1 we use interpolation technique to recover a considerable from the signals of its neighborhood. For validation purpose, we generate a validation set by performing 2D augmentation on each radar map of the training set. The optimization process is stopped whenever the validation error increase over several epochs. For evaluation, we use the official scorers from CIKM AnalytiCup 2017, which compute the Root Mean Square Error over the testing set.

The parameters of our model were (chosen on the validation set) as follows: The size of 2D and 3D convolution kernel is set to 3, the number of Residual Conv2D Unit in ResNet2D is set to 15, which mean there are more than 30 layers of convolution in this block. The number of Residual Conv3D Unit is set to 5, dropout factor in fully connected layers are set to 0.5. The BatchNorm with fully training two parameters of scaling and shifting is applied to all skip-connections except those are in the ResNet3D layers, we only standardize the feature maps in the block. Because of the lack of hardware resources (12 Gb memory of a single Nvidia Titan X), we only use 8 time-span of radar maps as input and keep 4 time-span of feature maps from the output of the Residual Convolutional LSTM.

4.2 Regularization tunning

We run our model (with parameters are explained in section 4.1) on the testing set in Phase 1 and Phase 2 of the challenge to evaluate its performance. To deal with the complication of the data, we need to build a complex architecture and eventually it leads to a model with so many parameters. In this experiment, we have four option of regularization: Weight decay only, Weight decay + Batch Norm (BN), Weight decay + BN + rainfall magnitude and Weight decay + BN + probabilistic labeling. The first option fails easily to converge, the second one is better but still far from the top 20 scores in the leader-board. The best result come from combining the BN with the rainfall magnitude level prediction, it's also the final score of our team on this challenge. The probabilistic labeling regularization lost his strength in the second phase because distribution of the rainfall in testing set may be very different from the training one. But if we have more prior information about the testing set, this technique could be a promise approach in future

REFERENCES

[1] P Cheung and HY Yeung. 2012. Application of optical-flow technique to significant convection nowcast for terminal areas in Hong Kong. In *The 3rd WMO International Symposium on Nowcasting and Very Short-Range Forecasting (WSN12)*. 6–10.

[2] Urs Germann and Isztar Zawadzki. 2002. Scale-dependence of the predictability of precipitation from continental radar images. Part I: Description of the methodology. *Monthly Weather Review* 130, 12 (2002), 2859–2873.

[3] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. 315–323.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Identity mappings in deep residual networks. In *European Conference on Computer Vision*. Springer, 630–645.

[6] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[7] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*. 448–456.

[8] Benjamin Klein, Lior Wolf, and Yehuda Afek. 2015. A dynamic convolutional layer for short range weather prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4840–4848.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[10] Min Lin, Qiang Chen, and Shuicheng Yan. 2013. Network in network. *arXiv preprint arXiv:1312.4400* (2013).

[11] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2012. Understanding the exploding gradient problem. *CoRR, abs/1211.5063* (2012).

[12] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*. 1310–1318.

[13] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[14] Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, et al. 2017. Tacotron: A Fully End-to-End Text-To-Speech Synthesis Model. *arXiv preprint arXiv:1703.10135* (2017).

[15] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144* (2016).

[16] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. 2015. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*. 802–810.