CNN-based Temporal Super Resolution of Radar Rainfall Products

Muhammed Sit  
University of Iowa  
Iowa City, IA  
muhammed-sit@uiowa.edu

Bong-Chul Seo  
University of Iowa  
Iowa City, IA  
bongchul-seo@uiowa.edu

Ibrahim Demir  
University of Iowa  
Iowa City, IA  
ibrahim-demir@uiowa.edu

Abstract

The temporal and spatial resolution of rainfall data is crucial for climate change modeling studies in which its variability in space and time is considered as a primary factor. Rainfall products from different remote sensing instruments (e.g., radar or satellite) provide different space-time resolutions because of the differences in their sensing capabilities. We developed an approach that augments rainfall data with increased time resolutions to complement relatively lower resolution products. This study proposes a neural network architecture based on Convolutional Neural Networks (CNNs) to improve temporal resolution of radar-based rainfall products and compares the proposed model with an optical flow-based interpolation method.

1 Introduction

From disaster preparedness to response and recovery needs, availability and quality of environmental datasets have gained significant importance in recent years with increased impact of natural disasters. Rainfall datasets have been an important component in many climate modeling applications such as flood forecasting [1], water quality [2], wastewater management [3], along with other data products. Since spatial and temporal distributions of rain exert importance in such modeling efforts, the quality and availability of precipitation maps hold the utmost importance in advancing climate change research.

Using Quantitative Precipitation Estimation (QPE), radar-acquired rainfall data could be four-dimensional. The first three dimensions reflect spatial coordinates on earth, the latitude, longitude and altitude. The fourth dimension here is temporal resolution. A three-dimensional rainfall data could be acquired by surpassing, or aggregating, the altitude component into each data point for the whole region.

The temporal resolution of rainfall products affects the accuracy of modeling efforts [4]. In an attempt to rectify the problem of low-quality rainfall products, this paper proposes a convolutional neural networks (CNNs) based deep learning architecture to increase the temporal resolution of rainfall products. The proposed CNN model, which will be referred to as TempNet throughout this paper, is compared with a baseline method that uses optical flow calculations. Along with the baseline method to compare, the proposed CNN produces an intermediate 2D rain map between two temporally consecutive rain maps. In other words, if two rain maps are given for $t_0$ and $t_{10}$, both the baseline method and the CNN-based model will produce a rain map for $t_5$.

The task of interpolating a 2D map between two 2D maps is not unique to rainfall products. The same problem has been a topic of interest in the field of computer vision also. Subsequently, various studies presented neural networks based approaches for video frame interpolation [5, 6, 7, 8]. On the other hand literature on temporal interpolation of rainfall datasets is rather limited. At the time of this writing, only study that presented a method for such problem was [9] that used advection correction in order to generate rainfall maps with 1-min resolution from 5-min ones.

Preprint. Under review.
The rest of this paper is structured as follows; in section 2, the overview of the methodology is presented. In section 3, the results for both the baseline method and the TempNet are presented and discussed. Finally, in section 4, the conclusions are summarized with final remarks.

2 Methodology

This section describes the methodology employed in this paper. In the following subsection, we start by describing the dataset used. We, then, provide details about the optical flow baseline and the CNN-based TempNet neural network architecture proposed in the study. Finally we describe how the TempNet was trained.

2.1 Dataset

The dataset used in this paper is a rainfall event dataset; namely, IowaRain presented by Sit et al. [10, 11]. IowaRain consists of 288 rainfall events from 2016 to 2019. Each rainfall event is formed by a set of temporally consecutive 2D rain maps, or snapshots, for each timestamp with a minute element that is divisible by 5.

The IowaRain dataset chronologically provides 64, 67, 76, and 81 events for each year in [2016, 2019]. In order to do the dataset separation in a fashion closer to 70/30 split, we decided to use the rainfall events for 2019 as the test set and all the rainfall events before 2019 as the training set, making our set lengths 207 and 81 for training and test sets respectively.

To prepare the dataset for training and testing, we formed a dataset entry for each snapshot $t_s$ in a rain event that has another snapshot coming right after ($t_s+5$) and right before ($t_s-5$) it. Each dataset entry consists of the $t_s-5$ and $t_s+5$ as the input and $t_s$ as the output (Figure 1). Then in order to augment the dataset for better training, we form additional dataset entries by reversing the order of snapshots in the event and double the number of dataset entries for the training dataset. The final dataset entries sum up to 35,258 and 6,725 snapshots for training and test sets, respectively.

2.2 Optical Flow

In order to form a baseline method that is comparable to the model we propose, we selected to use a simple optical flow based temporal interpolation method. This method depends on the optical flow calculation between two 2D rain maps and creating an intermediate frame between those rain maps that is at the same distance to both rain maps temporally.

There are many optical flow calculation algorithms in the computer vision literature. In this paper, we employed the Gunnar-Farneback optical flow [12]. Gunnar-Farneback optical flow calculates pixel intensities for each pixel in the scene. For the rain map case, that would mean calculating the changes for each of the measurements for 0.5km x 0.5km areas that form the 2D rain map. Once the optical flow is calculated, the next step is to transfer every measurement depending on their motion vectors in calculated optical flow, and their location in the first and second frame.

For comparison purposes, the optical flow between input snapshots was calculated for each of the entries in the test dataset. Then, color propagation was done using those calculations in order to estimate each intermediate frame. Once all the estimations were done for the testing dataset,
performance metrics, reported in the next section, were calculated using estimated and actual snapshots. The entirety of the baseline method was implemented using NumPy [13] numeric computing library and the Gunnar-Farneback optical flow implementation in the OpenCV [14] library.

2.3 TempNet

In order to explore how the temporal resolution of 2D rain maps could be improved, we proposed a CNN-based neural network architecture, TempNet. Albeit simple, the TempNet provides a fast but effective alternative to the optical flow based color transfer method we described in the previous subsection.

CNNs typically are used with images, and since images are represented with three color channels in most applications, a CNN that processes an input image typically has three filters. The dataset we employed, on the other hand, by its nature, has only one channel, which is hourly rainfall measurements. Thus a CNN that works on a single channel of input was built. However, since the neural network expects two 2D rain map as input, we had to use two different series of convolutional layers to learn the pattern over them first. Then the difference of those convolutional layers' outputs was added to the first frame in a skip connection fashioned architectural design choice that was introduced to improve the convergence of neural networks over the training data [15]. Finally, that summation was fed to another series of convolutional layers to output the intermediate frame between two input frames (Figure 2).

The network described here was trained using L1 Loss (also known as Mean absolute error) as the cost function and Adam [16] as the optimizer with the help of a Reduce-on-Plateau learning rate scheduler over the training loss. The network was trained on the training dataset described in the Dataset subsection on NVIDIA Titan V GPUs using the PyTorch numeric computing library [17]. As for hyperparameters, batch size of 32, the initial learning rate of 1e-3, and the number of epochs of 50 without early stop were used.

3 Results & Discussions

This section defines performance metrics and presents results using those metrics for both the optical flow based interpolation method and the TempNet. The loss changes (L1 Loss) over both training and test datasets during 50 epochs of training are given in Figure 3 along with epoch times. As the figure suggests, the performance of the network steadily increases over epochs and gets stabilized after the 40th epoch. It is worth noting that the costs reported in Figure 3 are averaged over the respective datasets, and they are for normalized values.
Figure 3: Change in epoch time, training loss and test loss over 50 epochs of training.

We report three metrics for both methods, namely Mean Absolute Error (MAE), Non-Zero Loss (NZL), and Ground Non-Zero Loss (GNZL). MAE reports the mean of the absolute values of the differences between estimated and actual 2D rain maps in the test dataset. NZL reports the MAE but only for non-zero values over estimated and actual 2D rain maps. GNZL, on the other hand, reports MAE for non-zero values in the actual 2D rain maps. Table 1 summarizes the best training TempNet model’s performance over the test dataset as well as the optical flow based baseline’s performance using the metrics mentioned above.

| Method      | MAE    | NZL    | GNZL   |
|-------------|--------|--------|--------|
| Optical Flow| 0.851  | 3.182  | 3.622  |
| TempNet     | 0.743  | 2.750  | 2.930  |

As Table 1 suggests, TempNet significantly outperforms the optical flow based baseline interpolation method for all of the metrics defined above by significant margins. Thus for a more accurate temporal resolution increase, TempNet offers a better solution than calculating optical flow. Beyond accuracy, one upside of using TempNet is the runtime. Since calculating optical flow and building the new frame out of two sequential frames need work over individual pixels or measurements, parallelization is challenging and, consequently, time-consuming. Conversely, the same task takes a trivial amount of time on both GPUs and CPUs using TempNet once the training is done compared to the baseline method’s runtime.

We did some tests to understand the visual performance of the TempNet by increasing the temporal resolution of some events in the test dataset by three iterations, meaning increasing the temporal resolution from 5-minutes to 0.7-minutes. After visualizing the actual and generated rain maps with a rainfall map color scheme [13], even though we were able to see that the TempNet was not able to carry all the information regarding the rainfall in specific areas, it was still able to represent most accumulated rainfalls clearly. Alternatively, the baseline method would just visually average two input frames but not so accurately per the metrics.

4 Conclusions

In this study, we presented a CNN-based temporal resolution improvement model, TempNet, and compared it to an optical flow based baseline method. Preliminary results clearly show that TempNet outperforms the baseline model according to various metrics as well as in a runtime test. We consider this work as a significant step towards creating better rainfall maps for hydrological modeling needs such as flood forecasting [19, 20] and climate change modeling [21, 22].
References

[1] M. Sit, B. Demiray, and I. Demir, “Short-term hourly streamflow prediction with graph convolutional gru networks,” arXiv preprint arXiv:2107.07039, 2021.

[2] M. K. Jha, P. W. Gassman, and J. G. Arnold, “Water quality modeling for the raccoon river watershed using swat,” Transactions of the ASABE, vol. 50, no. 2, pp. 479–493, 2007.

[3] L. B. Cahoon and M. H. Hanke, “Rainfall effects on inflow and infiltration in wastewater treatment systems in a coastal plain region,” Water Science and Technology, vol. 75, no. 8, pp. 1909–1921, 2017.

[4] A. Atencia, L. Mediero, M. Llasat, and L. Garrote, “Effect of radar rainfall time resolution on the predictive capability of a distributed hydrologic model,” Hydrology and Earth System Sciences, vol. 15, no. 12, pp. 3809–3827, 2011.

[5] S. Niklaus, L. Mai, and F. Liu, “Video frame interpolation via adaptive convolution,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 670–679.

[6] ——, “Video frame interpolation via adaptive separable convolution,” in Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 261–270.

[7] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, and A. Agarwala, “Video frame synthesis using deep voxel flow,” in Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 4463–4471.

[8] H. Jiang, D. Sun, V. Jampani, M.-H. Yang, E. Learned-Miller, and J. Kautz, “Super slomo: High quality estimation of multiple intermediate frames for video interpolation,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 9000–9008.

[9] B.-C. Seo and W. F. Krajewski, “Correcting temporal sampling error in radar-rainfall: Effect of advection parameters and rain storm characteristics on the correction accuracy,” Journal of Hydrology, vol. 531, pp. 272–283, 2015.

[10] M. Sit, B.-C. Seo, and I. Demir, “Iowarain: A statewide rain event dataset based on weather radars and quantitative precipitation estimation,” arXiv preprint arXiv:2107.03432, 2021.

[11] B.-C. Seo, M. Keem, R. Hammond, I. Demir, and W. F. Krajewski, “A pilot infrastructure for searching rainfall metadata and generating rainfall product using the big data of nexrad,” Environmental modelling & software, vol. 117, pp. 69–75, 2019.

[12] G. Farnebäck, “Two-frame motion estimation based on polynomial expansion,” in Scandinavian conference on Image analysis. Springer, 2003, pp. 363–370.

[13] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith et al., “Array programming with numpy,” Nature, vol. 585, no. 7825, pp. 357–362, 2020.

[14] G. Bradski and A. Kaehler, “Opencv,” Dr. Dobb’s journal of software tools, vol. 3, 2000.

[15] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[16] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” arXiv preprint arXiv:1412.6980, 2014.

[17] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga et al., “Pytorch: An imperative style, high-performance deep learning library,” Advances in neural information processing systems, vol. 32, pp. 8026–8037, 2019.

[18] I. Demir, E. Yildirim, Y. Sermet, and M. A. Sit, “Floodss: Iowa flood information system as a generalized flood cyberinfrastructure,” International journal of river basin management, vol. 16, no. 3, pp. 393–400, 2018.
[19] Z. Xiang, I. Demir, R. Mantilla, and W. F. Krajewski, “A regional semi-distributed streamflow model using deep learning,” 2021.

[20] M. Sit and I. Demir, “Decentralized flood forecasting using deep neural networks,” *arXiv preprint arXiv:1902.02308*, 2019.

[21] D. Rolnick, P. L. Donti, L. H. Kaack, K. Kochanski, A. Lacoste, K. Sankaran, A. S. Ross, N. Milojevic-Dupont, N. Jaques, A. Waldman-Brown *et al.*, “Tackling climate change with machine learning,” *arXiv preprint arXiv:1906.05433*, 2019.

[22] M. Sit, B. Z. Demiray, Z. Xiang, G. J. Ewing, Y. Sermet, and I. Demir, “A comprehensive review of deep learning applications in hydrology and water resources,” *Water Science and Technology*, vol. 82, no. 12, pp. 2635–2670, 2020.