Rain forecasting with temporal GAN

Peter Kicsiny, Jongwon Lee
Technical University of Munich
Boltzmannstr. 15, 85748 Garching bei München, Germany
pkicsiny@gmail.com, jwonlee206@gmail.com

Abstract

The goal of this project was the prediction of rain intensities using radar echo images. We proposed to use generative adversarial networks and tackle the problem by improving the network with Wasserstein loss with gradient penalty and a two discriminator model.

1. Data

We have created our own dataset from the open data base of the German Weather Service (DWD) [1]. We used radar reflectivity measurements for precipitation which have five minutes temporal resolution.

1.1. Preprocessing

First we have cut 64×64 sized image sequences from the original 900×900 sized radar maps with an automated random cropper method. We have filtered images containing any number of mask pixels, empty and overly saturated images. Then we normalized the frames into $[0, 1]$.

We precomputed optical flow images between the last two frames of the input sequence with the Lukas-Kanade method [2, 3]. Then applied a median filter for smoothing and normalized them into $[-1, 1]$.

1.2. Data augmentation

Wind in the data was not blowing equally from all directions but from a few specific dominant ones. This caused our networks to learn the most dominant flow and push everything a bit in that direction. To solve this we did augmentation by 90°, 180° and 270° rotations.

In the end we have used 8000 training and 1000 validation sequences (for visual checking during training). For testing we have used a different set of 1000 samples but with longer sequences for the longer term sequence prediction where we generated the next five frames iteratively.

2. Model

We built several GANs to compare results when gradually improving the architecture [4, 5, 6, 7]. We trained with a one discriminator GAN, then we added Wasserstein loss with weight clipping. After that we used gradient penalty loss instead and then trained two discriminator models.

2.1. Generator

At first we tested different generator candidates. Our two best performing ones were an LSTM and a U-net. In the end we chose the U-net for our further trainings because its skip connections help to preserve global structure. We used batch normalization, 0.1 leaky ReLUs but no dropout.

2.2. Discriminators

Our discriminators are fully convolutional, have the same structure and use binary crossentropy loss or Wasserstein loss for WGAN trainings. The only difference is in the inputs they receive. The "spatial" discriminator is conditioned on the input sequence. The "temporal" discriminator is conditioned on a reconstructed version of the predicted frame or the ground truth using the discretized advection equation [8, 9, 10, 11]. We used a dropout rate of 0.25, batch normalization and 0.2 leaky ReLUs.

3. Results and evaluation

We trained the models for 5000 iterations. See Fig. 1 for training curves. For evaluation we used the following metrics[12]: we compared pixel intensities and we also calculated the intensities back to $\frac{mm}{h}$ values to get actual rainfall rates [13, 14]. Then we converted our predictions and ground truth images into binary images depending on the pixelwise relation to a threshold value of $0.5\,\frac{mm}{h}$ for the rainfall rates and 18 for the intensities and for each pair we computed the hits: pred=1 & truth=1, misses: pred=0 & truth=1 and false alarms: pred=1 & truth=0. From these we calculated three score values. Critical success index: $\text{CSI} = \frac{\text{hits}}{\text{hits+misses+falsealarms}}$, false alarm ratio: $\text{FAR} = \frac{\text{falsealarms}}{\text{hits+falsealarms}}$ and the probability of detection:

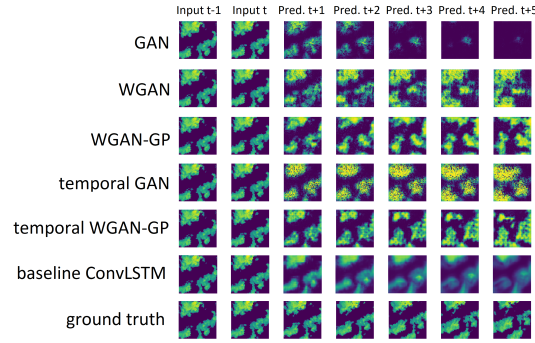
$\text{POD} = \frac{\text{hits}}{\text{hits+misses}}$. We also computed the correlation between predictions and ground truths.
We have compared our evaluation scores to a ConvLSTM baseline model from [12]. We present the scores for each of our networks in Table 1.

4. Conclusions

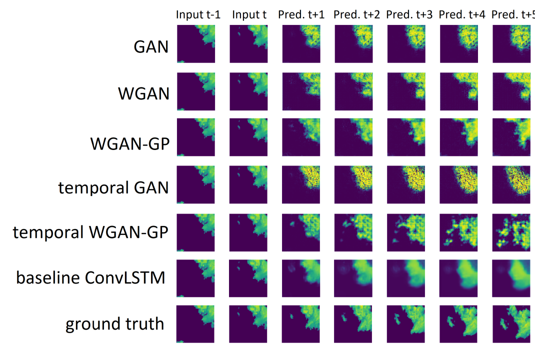
According to our scores the WGANs with gradient penalty can generate the best predictions. In terms of the FAR, the single GAN performs the best but as it can be seen on Fig. 2 it reduces the intensity of the predictions over time. This could be improved by predicting a sequence rather than a single next frame.

In conclusion we can say that the improved WGAN architecture helps more than adding a second discriminator in our case but still underperforms the baseline model in terms of the score values. The predictions of our GAN models are however sharper and more detailed. Additional temporal consistency due to the second discriminator can be observed on the second example where the predicted intensities can better follow the motion of the ground truth (rainy areas move downwards more) and small patches with the WGAN-GP are preserved (continuity).

Our results can be further improved by taking a larger dataset, training for more iterations and predicting longer sequences.

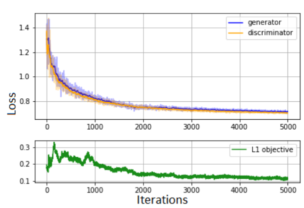


(a)

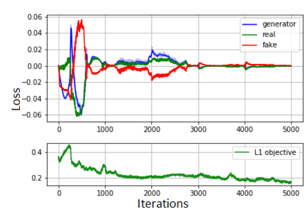


(b)

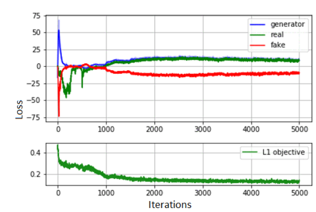
Figure 2: Sequence prediction examples. The next frame is always predicted from the preceding two. Best viewed zoomed in.



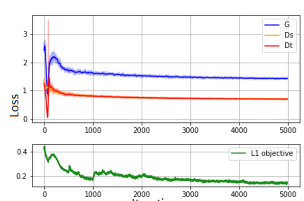
(a) Single discriminator GAN with $\lambda_{L_1} = 0.1$, batch size = 64.



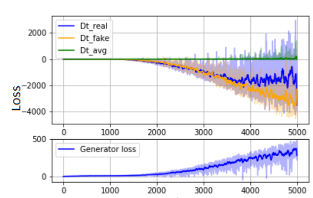
(b) Single discriminator WGAN with $c = 0.01$, batch size = 64.



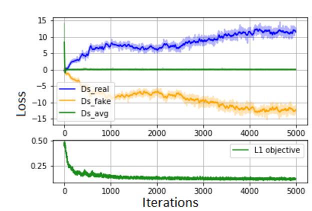
(c) Single discriminator WGAN with gradient penalty $\lambda_{gpl} = 10$, batch size = 16.



(d) Two discriminator GAN with $\lambda_{L_1} = 0.3$, batch size = 64. The L_1 weight was reduced to 0.1 after 1000 iterations.



(e) Two discriminator WGAN-GP $\lambda_{gpl} = 10$ and $\lambda_{D_t} = 0.5$, batch size = 16. Generator and temporal discriminator curves.



(f) Two discriminator WGAN-GP $\lambda_{gpl} = 10$, batch size = 16. Spatial discriminator curves.

Figure 1: Training curves of our models. In case of the WGAN trainings the L_1 loss was only a metric. ADAM was used in the generator and SGD in the discriminators. For WGAN-GP trainings we changed both to RMSProp.

Scores for rainfall rates (R.) and pixel intensities (I.)								
Model	R. Corr.	R. CSI	R. FAR	R. POD	I. Corr.	I. CSI	I. FAR	Int. POD
sGAN	0.34091	0.16724	**0.63911**	0.2489	0.77111	0.55474	**0.23899**	0.67601
sWGAN	0.32419	0.17105	0.71565	0.29974	0.68716	0.51030	0.38910	0.74111
sWGAN-GP	**0.38742**	**0.20912**	0.68938	0.37673	0.77107	0.58216	0.31013	**0.77934**
tGAN	0.36682	0.19116	0.73631	***0.43248***	0.74307	0.56242	0.25583	0.69944
tWGAN-GP	0.38022	0.195964	0.69428	0.3604	**0.78057**	**0.59385**	0.29325	0.77838
ConvLSTM	***0.64380***	***0.29479***	***0.33134***	0.35576	***0.89218***	***0.69969***	***0.22649***	***0.87616***

Table 1: Evaluation scores for our models. The rainfall rate values are obtained after back calculation to $\frac{mm}{h}$ with a mapping used in [12, 13, 14]. Best values are bold italic. Best values from our models are bold.

References

[1] *DWD Open Data*. `https://www.dwd.de/DE/leistungen/opendata/opendata.html` `https://opendata.dwd.de/weather/radar/composit/rx/`

[2] Robert Collins: Lucas Kanade Tracking. `http://www.cse.psu.edu/ rtc12/CSE486/lecture30.pdf`

[3] Sandipan Dey: Implementing Lucas-Kanade Optical Flow algorithm in Python. `https://sandipanweb.wordpress.com/2018/02/25/implementing-lucas-kanade-optical-flow-algorithm-in-python/`

[4] Code reference for WGAN. `https://github.com/eriklindernoren/Keras-GAN/blob/master/gan/gan.py`

[5] Code reference 2 for WGAN. `https://myurasov.github.io/2017/09/24/wasserstein-gan-keras.html`

[6] Code reference 3 for WGAN. `https://github.com/bobchennan/Wasserstein-GAN-Keras/blob/master/mnist_wacgan.py`

[7] Code reference 4 for WGAN. `https://github.com/keras-team/keras-contrib/blob/master/examples/improved_wgan.py`

[8] Code reference for advection. `https://github.com/thunil/tempoGAN/blob/master/tools/tests/test_0150_advect.py`

[9] Rony Keppens. Apr 2007. `https://perswww.kuleuven.be/ u0016541/Talks/advection.pdf`

[10] Mantaflow. `http://mantaflow.com/scenes.html`

[11] You Xie, Erik Franz, Mengyu Chu, Nils Thuerey. *tempoGAN: A Temporally Coherent, Volumetric GAN for Super-resolution Fluid Flow*. Jan 2018. `https://arxiv.org/pdf/1801.09710.pdf`

[12] Xingjian Shi, Zhouron Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, Wang-chun Woo. *Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting*. Jun 2015. `http://papers.nips.cc/paper/5955-convolutional-lstm-network-a-machine-learning-approach-for-precipitation-nowcasting.pdf`

[13] Deutscher Wetterdienst. *Beschreibung des Kompositformats*. Aug 2018 `https://www.dwd.de/DE/leistungen/radolan/radolan_info/radolan_radvor_op_komposit_format_pdf.pdf?__blob=publicationFile&v=11`

[14] Wikipedia page for dBZ. `https://en.wikipedia.org/wiki/DBZ_(meteorology)`