Precipitation Forecasting using U-Net Architecture

Pierce Coggins, Connor Stern, Miroslava Walekova
W251: Deep Learning in the Cloud and at the Edge
UC Berkeley School of Information
{tcoggins, connor.m.stern, walekova}@berkeley.edu

ABSTRACT

This project applies a U-Net convolutional network architecture to meteorological precipitation data to predict future precipitation radar imagery based on current radar imagery. As a second iteration, we added the difference between the two most recent radar images as an additional input channel to better forecast weather directionality. This model uses Rainviewer data, trained on IBM Cloud and set up to run inference on real-time data.

1. INTRODUCTION

People have been trying to forecast weather for millenia. In general, weather forecasts today are prepared by collating quantitative weather data which are processed by complex mathematical models to prepare weather forecast models. The types of data leveraged by these models are surface observations from weather stations, radiosondes aka 'weather balloons', weather satellite data, meteorological radar data and pulse Doppler weather radar (wind speed & direction). Modern data weather forecasting methods use an ensemble approach and collect nonparametric statistics through a series of perturbed simulations.[2] In effect, our current weather forecasting methods are built upon an ensemble prediction system using noisy partially synthetic data.

In this paper we will focus on exploring the potential of using precipitation radar imagery instead of perturbed synthetic data to forecast accurate precipitation radar imagery. Recent research has utilized the U-Net convolutional network architecture to predict weather uncertainty[2],so it is our objective to build upon this research and use the U-Net architecture to predict rain radar image one hour into the future.

The data set used for this proof of concept are rain radar images captured at hourly intervals. We have used approximately 49,700 image pairs from 530 locations across the US and Europe.

2. PROJECT OVERVIEW

2.1. Data Sourcing and Collection

We considered three weather data providers as our data source: NOAA, Rainviewer and Accuweather data. Both NOAA and Rainviewer data is free while Accuweather radar data is available only as a paid service.

The image data that we considered is only available for download using API services with limited historical data available. We have therefore created python scripts that, at one hour interval, execute an API call to download all available images from the providers and uploaded them to IBM cloud object storage. Through this process we aggregated sample data from each of the three providers before selecting Rainviewer.

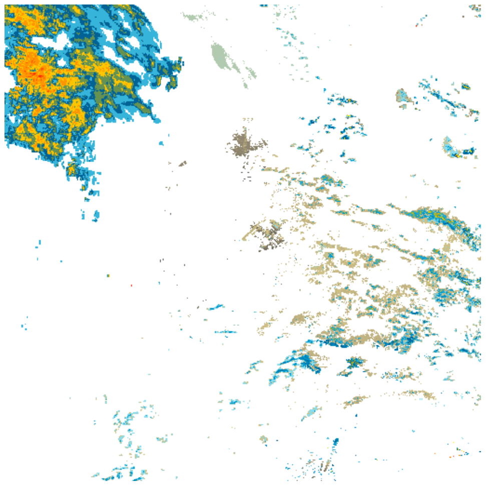
2.2. Dataset

We chose Rainviewer as our data provider primarily because it was a free source of data with no limit on the number of API calls per day. This allowed us to quickly aggregate a dataset large enough to train a U-Net convolutional network architecture.

Additionally, the Rainviewer radar images do not include any underlying topographic information, which makes it easier for our model to evaluate changes in the source-target image pairs as only relevant radar information is included in the images.

Lastly, NOAA and Accuweather radar data cover relatively large regions. For example, Accuweather radar image data is available only for Europe as a whole, and it is only available at 1 hour intervals, while NOAA provides unprocessed data that is not accessible through API and is periodically updated in an Amazon S3 bucket. As a result, we elected to not consider either of these data sources as we would not have been able to accumulate a large enough data set to train our U-Net model.

Image 1: Rainviewer image example



We were able to collect 49,700 rain radar image pairs, captured at one hour intervals across 530 different cities throughout the US and Europe.

2.3. Background Research

There are four main forms of clouds:

- Cirriform (fibrous)
- Cumuliform (heaped)
- Stratiform (layered)
- Nimbus (Rain-bearing)

Clouds are further subdivided based on the level of their bases above sea level. The cloud classification is not straightforward as clouds may take on many forms, many of which continually change.

Intermittent, continuous precipitation and heavy continuous rain are usually associated with a certain type of clouds.

We can forecast precipitation based on the cloud cover. However, it is also possible to use precipitation as a means of identifying cloud type - showers generally falling from cumuliform clouds and non-showery precipitation from stratiform clouds, mainly altostratus and nimbostratus'.[1]

In this paper we chose to explore the potential of predicting the cloud and rain radar imagery from actual cloud and rain radar images. Because of data availability we have focused on rain radar imagery, however subject to the result similar training approach can be used for cloud cover imagery.

As will be discussed in the next section, we elected to use the U-Net convolutional network architecture commonly used for image segmentation to generate realistic precipitation radar image forecasts. Although this architecture is most commonly used within image segmentation research [3,4], there are examples of this architecture being applied to weather forecasting[2].

2.4. Initial Model Selection

At the outset of our project we were interested in generating precipitation radar imagery based on collated tabular data pulled from a grid of locations based on latitude and longitude. However, after a deeper assessment of weather imagery models we realized that virtually no research had been completed on the topic, and rendering precipitation radar imagery from tabular data alone likely wouldn't yield the strongest results.

Although generating synthetic radar imagery based on tabular data was not feasible, we elected to forecast radar images based on the current state by using a precipitation radar image input, and generating the next sequential radar image as output.

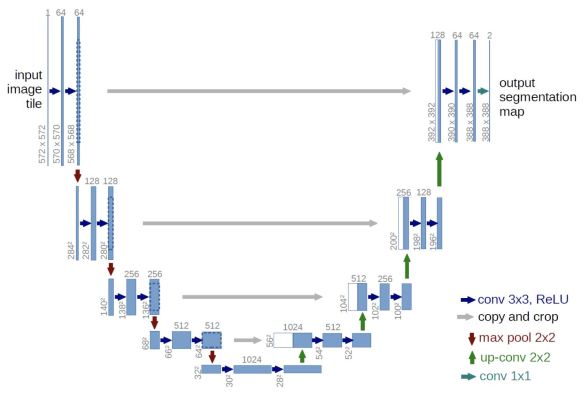
To take on this task we found the U-Net convolutional architecture, which is often used for image segmentation, would allow us to generate a forecasted precipitation radar image based on a current image. We also saw the potential of using the difference of the two most recent radar images as our input image in order to forecast based on a series of radar imagery.

3. MODEL

3.1. Overview of U-Net Architecture

The basic idea behind the U-Net convolutional network architecture is that a source image is converted into a feature map vector used for prediction. The predicted vector is then converted back into an image using the same mapping.

Image 2: UNet Architecture



The architecture consists of three main components:

1. **Contraction** is used to capture the context in an image. It is a repeated application of down sampling methods (convolutions & max pooling) to reduce the size of the image. Each contraction is referred to as a block, while the size of the image contracts at each iteration the number of feature maps doubles.
2. **Bottleneck** connects the contraction layer and the expansion layer.
3. **Expansion** is where the features that are learned while contracting the image are used to reconstruct it.

3.2. Implementation

Our U-Net model was built using the Keras API on top of Tensorflow. The Rainviewer images were resized to 128x128, and input into the model with four different image color channels (RGBA). When used for image segmentation, the U-Net architecture model makes a binary prediction for each pixel of the output image, which consists of a single image color channel. This model configuration was not sufficient for the purpose of this project, so we modified the architecture to predict output pixel values between 0 and 1 for the 4 image color channels present in the input image. Fully built, our model has 1,941,300 parameters, all of which are trainable. For training, we use the adam optimizer and binary cross-entropy loss function. Our evaluation metric for measuring model performance was the mean squared error (for each pixel value). We note that accuracy and meanIOU are typically the metrics of choice when implementing a U-Net architecture. However, these are only useful when predicting binary pixel outcomes, as they score each pixel on a binary scale. While this is appropriate for image segmentation, it is insufficient for the purposes of our model where we would like our output to resemble our RGBA input image by predicting pixel values between 0 and 1. Training was performed on a NVIDIA Tesla V100 GPU in the cloud.

3.3. Baseline Model Performance

For baseline model, we seek to predict the rain radar image one hour into the future given a single input image of the current rain radar. Our training set therefore consists of each rainview image from our dataset (input) paired with the Rainview image of the same location one hour later (target output).

Using a validation split of 0.1, we trained the model on 44,765 samples and validated on 4,974. Model training and performance statistics can be seen in the table below.

Baseline Model Training Statistics			
# of Epochs	**Train time**	**Val Loss**	**Val MSE**
26	70s/epoch	0.10166	0.0054

3.4. Model Performance - 2nd Iteration

In our second iteration, we attempt to include additional input information to represent the weather "trend", or how the radar images for the given location have been changing over time. We did this by taking the difference between a given "current" radar image and the "previous" (T-1hr) radar image of the same location. This second, "difference" image data is passed into the model along with the input image of the "current" time. Model training and performance statistics are shown below.

Second Model Training Statistics			
# of Epochs	**Train time**	**Val Loss**	**Val MSE**
31	82s/epoch	0.10067	0.0052

3.5. Model Performance - 3rd Iteration

Finally, we implemented a third iteration of our model, taking only the difference between radar images over the previous hour for a specific location as the input. By isolating the changes between sequential radar images, we hoped to better capture the movement between frames. Model training and performance statistics are shown below.

Third Model Training Statistics			
# of Epochs	Train time	Val Loss	Val MSE
18	66s/epoch	0.10561	0.0065

4. RESULTS

4.1. Collective Results

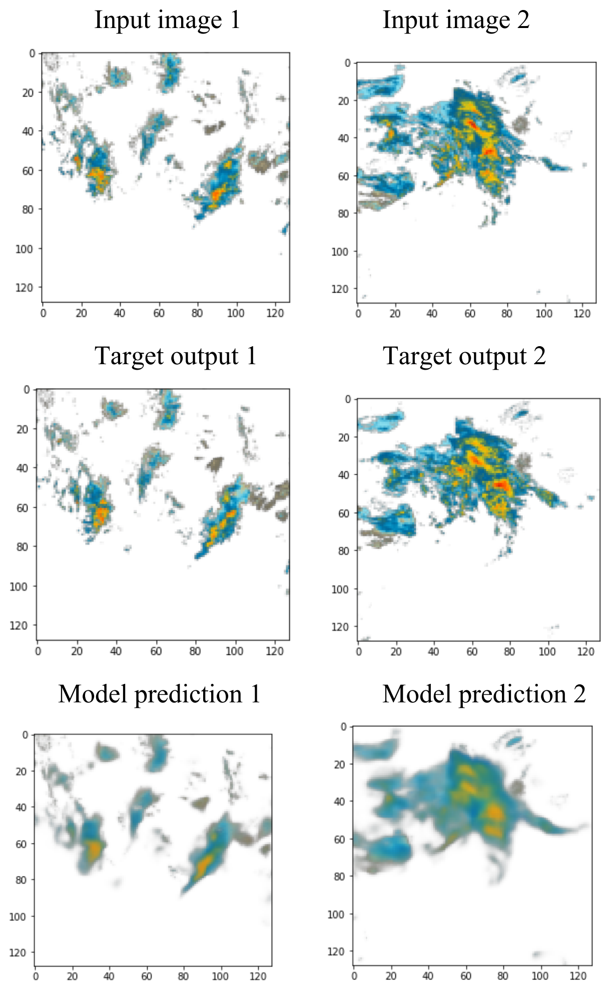
Below we compile the training and validation statistics for all three iterations of our model in a single table for comparison.

	Collective Results		
Iteration	**Time (s/epoch)**	**Loss**	**MSE**
V1-Train	**70s**	**0.0985**	**0.0057**
V1-Valid		**0.10166**	**0.0054**
V2-Train	**82s**	**.0974**	**0.0055**
V2-Valid		**0.10067**	**0.0052**
V3-Train	**66s**	**0.1019**	**0.0067**
V3-Valid		**0.10561**	**0.0065**

We note that neither the second nor the third iterations of our model provide significant improvement over our baseline implementation, as far as these metrics are concerned. As such, it is the baseline model that we elect to use for inference in the following stages. This model is also the simplest to deploy, as the input requires minimal pre-processing.

4.2. Accuracy and Error Types

Below are several samples of the input, target output, and predictions generated by our baseline model for data from the training and validation sets:



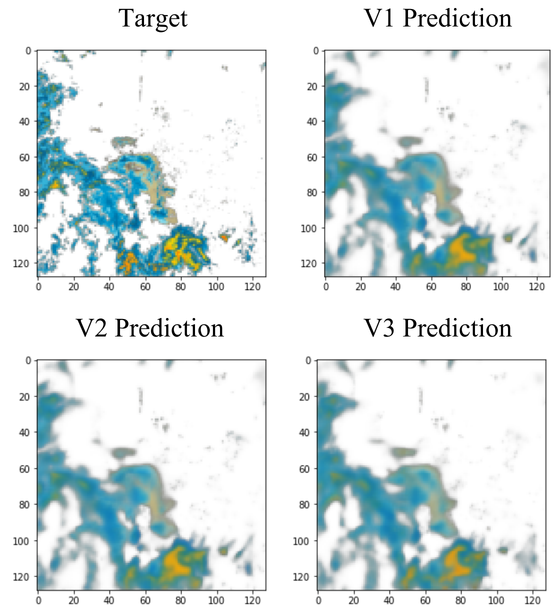
We notice the model is able to reasonably predict general shape, size, and color density of the rain "patches" in the image. However, the prediction image has a much more blurred quality to it than the target output image, and we rarely see significant movement of patches away from their original location in the input image.

4.3. Discussion of Errors

There are a couple potential shortcomings with the baseline model. To begin, there are only subtle differences between the input image and the target output image—the rain radar for a given location does not change significantly from hour to hour. As such, our model would do reasonably well by simply predicting the input image. Looking at several examples, our inputs and predictions do indeed look very similar.

Secondly, a single input image does not give our model any information about weather trends. In its current state, the baseline model predicts a blurry image because it cannot effectively predict which direction the color patches will move, as it essentially learns an average movement over all the training data. We conjecture that the additional knowledge of how the rain radar image has changed in the hour *previous* to the input image could help the model better predict how it will change in the next hour.

Unfortunately, the second iteration of our model which includes this additional information did not yield significantly better results. Below is an example comparing the predictions of the baseline, second, and third iterations of the model with the target output.



All three predictions share the same blurry quality, which is especially pronounced in the third prediction (where the input is only the difference in radar images over the previous hour). The additional information provided in the second iteration of the model did not benefit the quality of the predicted image, when compared to the baseline prediction. This may suggest that the differences between radar images from hour to hour are too small to provide enough information about the weather trend to significantly improve the quality of our prediction.

4.4. Real-time Inference

We have used an inference solution in jupyter notebook to implement real-time inference and monitor the progress of learning. We have downloaded new image pairs using the Rainviewer API.

Image 3: Inference in jupyter notebook

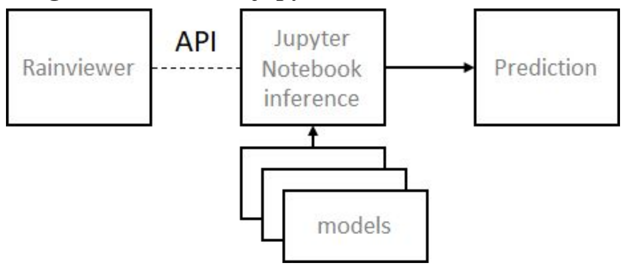
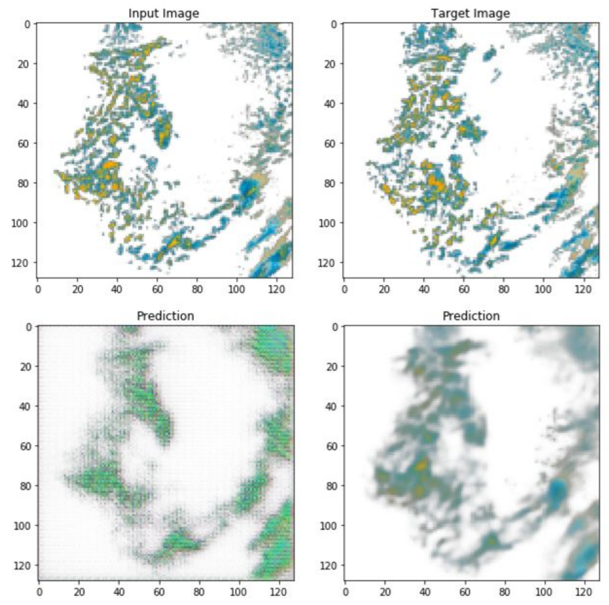


Image 4: Training progression



5. FUTURE DEVELOPMENT

Upon visual examination we saw significant improvement in forecasting the more extreme areas of precipitation, and was able to capture some of the movement of rain coverage, although this was not precisely accurate. Therefore we believe there is additional value in further evaluating our precipitation forecasting model.

In future iterations, we would like to increase our existing corpus of radar imagery from 49k image pairs to 4.5M image pairs in order to improve our model performance through the application of a larger dataset.

Additionally, while the 128x128 radar image data pulled from Rainviewer served as a good basis for building our model, we would be interested in training our model on higher resolution data, or data that covers a larger area in order to make forecasts based on macro weather patterns rather than localized weather patterns. With this data, we would be interested in testing our image differencing technique (i.e. (T - 2hr) - (T - 1hr)) to better predict directional movement, and simplify input data by removing information that has remained unchanged. Once our forecasts are accurate enough we may be able to use our predictions to make a second 'T + 2hrs' forecast.

Lastly, although our model performed well when predicting precipitation intensity, we believe our model could benefit from supplementary information (i.e. wind speed and direction) to improve the ability to forecast the movement of precipitation areas. In order to isolate the subtle shifting of rainfall, we could have tested alternate evaluation metrics to set a baseline for accurately forecasting the precipitation region or "footprint" as a whole without also evaluating performance on correctly forecasting precipitation intensity.

6. CONCLUSION

In this project we have demonstrated the promise of precipitation forecasting using a simple U-Net convolutional network model trained on raw precipitation radar data instead of a complicated ensemble method build on perturbed synthetic data.

Throughout the training process we were able to dramatically improve the accuracy and resolution of forecasted radar imagery as we gradually increased our training set and refined our training process and hyperparameters. At the time of releasing this paper, our model was trained on nearly 45k radar image pairs, and reached a validation loss of 0.099, yielding a MSE of 0.0053. Through the subsequent iterations of our model we were not able to significantly improve upon our original validation MSE of 0.0053. Although the differencing of radar images was promising in theory, it did not provide an increase in model performance as either the sole input or an additional input channel. After sufficient training, we were able to implement our most performant model for inference on real-time data pulled from the Rainviewer API.

Overall, this project has laid a solid foundation for future research in the application of U-Net architecture for precipitation forecasting, and has identified a few areas of interest for future research including the use of higher resolution input data, and larger input images to capture macro weather patterns that likely have a significant impact on local weather conditions. With a model trained on larger, higher resolution imagery we believe there may still be promise in the differencing method we have used to capture the serial aspect of weather data. Additionally, we suspect the use of MSE to evaluate our models and guide our research may have emphasized correctly predicting the intensity of each pixel (i.e. precipitation intensity) instead of correctly predicting the basic area or "footprint" of expected precipitation. Evaluating on a metric such as MeanIoU, as is often used in image segmentation, could have helped us to equally value the edges of the radar imagery that are more difficult to predict, and ultimately evaluate which models best predicting the directional movement of our radar imagery.

Overall, we're very happy with the performance and progression of our model and look forward to continuing our research.

REFERENCES

[1] Air Law & Meteorology, Pooley's Air Pilot Publishing, 2015

[2] P. Grönquist, T. Ben-Nun, N. Dryden, P. Dueben, L. Lavarini, S. Li, T. Hoefler: *Predicting Weather Uncertainty with Deep Convnets*

[3] O. Ronneberger, P. Fischer, T. Brox: *U-Net: Convolutional Networks for Biomedical Image Segmentation*

[4] J. Gonzalez Villarreal, D. Bhowmick, C. Beltran Castañon, K. Sankaran: *Applying Knowledge Transfer for Water Body SEgmentation in Peru*