Michael Robinson
Professor Tomuro
CSC 578 Section 901
08 June 2023

Executive Summary

The main goal of this project was to classify if future weather conditions would be mostly wet or mostly dry. Specifically, the conditions in a 40 km radius centered on King City, Canada were predicted four hours out using the last four hours of weather data. The initial proposal suggested that a CNN-RNN architecture might be suitable for this task. Although Keras has tools to achieve this (LSTMConv2D layer), it was found to be too computationally expensive (even after cropping the images). Instead, the images' sizes were reduced to 100x100 by average pooling after they were cropped. After flattening, they were fed into a simple RNN layer followed by a smaller LSTM layer. Then the output was flattened and passed into a dense layer before being output. The dataset was fairly imbalanced with the negative class rate (dry conditions) at about the 90^{th} percentile for the training data. Sample weighting, the Poisson loss function (y_pred – y_true * log(y_pred)) and Keras callback monitoring of the recall were used to increase the f1 score.

The nature of time series forecasting compounded with imbalanced classes made prediction challenging. The author suspected that up-sampling the minority class or down-sampling the majority class could cause interruptions in the time series. The Poisson loss function was used because given that the Sigmoid activation was being used for the final output, if y_true = 0 than the maximum Poisson loss for this output would be 1 when y_pred = 1. For y_true = 0 with the binary cross entropy loss, a y_pred = 1 gives a near positive infinity loss value. Although the occurrence of wet weather does not likely follow a Poisson distribution, the Poisson loss function did encourage more predictions of the positive class.

Due to missing files, the sequence was split into 6 subseries (between each subseries were one or multiple missing data points) and the two longest subseries were split each into three parts (with the first 80% of each subseries being training, the next 10% being validation and the final 10% being testing). The other subseries were discarded entirely. The simple baseline model that looked at the last input point before prediction (and if it was wet, it predicted wet for the next four hours, and vice versa) yielded the results: 98% accuracy with f1 score of .07 for subseries 1 and 80% accuracy with f1 score of .61 for subseries 2. Upon examining the test data (after all experiments were done, including the actual model) it was noted that the test data was 99% dry in subseries 1 and 75% dry in subseries 2. Although this had no effect on the baseline model it likely hampered the performance of the actual model (given the training data was near 90% dry and 10% wet for both subseries).

Given the sequential nature in this forecasting, I used two models with identical architectures (one for each subseries, that was trained on data immediately before predictions chronologically). After achieving validation results around 94% and 0.61 f1 score for the first subseries, I trained on the validation data of that subseries and then tested, getting 99% accuracy but an undefined f1 score (none of the 216 wet conditions were predicted). The validation data from the second subseries produced 81% accuracy and a .34 f1 score (poor but better than the baseline). After training on this, the test set yielded 81% with 0.54 f1 score. This is slightly below the baseline model for the f1 score.

Overall, although my model did not perform better than the baseline model, I think my model does show promise for this type of time series prediction. In the future, more emphasis

should be put on metrics like f1 score when training. The loss function, sample weighting and callbacks could all be more fine-tuned. With more computation power ConvLSTM layers could be explored. Finally, defining what constitutes wet or dry weather could be better defined – predicting weather for a 5000 km^2 area is quite hard, narrowing down the area to a smaller region or lowering the threshold for wet conditions could make the problem easier. In addition, multiple images could be overlayed and averaged, rather than predicting 24 10 minute images, each hour of weather could be averaged (mathematically), which would help eliminate noise and make prediction easier.