Snow Prediction Using LSTM

Dane Laufer and Eli Cytrynbaum

Weather Forecasting:

Weather predictions have been crucial to human survival and enjoyment for millenia. Weather predictions are crucial to successful crop sewing and harvesting, fishing, and pretty much anything else we do outside. The first evidence of weather forecasting dates back to ~650 BCE in Babylonia (mostly based on cloud formations), and everyone seems to have their own homebrewed weather predictions – based on achy joints, “smelling like rain,” the presence of cirrus clouds, or a “red sky” in the morning or night.

Nowadays, most meteorology is based on a network of automated weather stations, buoys, and satellites. These provide data which is input into nonlinear partial differential equations based on a scientific understanding of fluid dynamics, gas laws and the like. These are generally impossible to solve analytically, leading meteorologists to instead rely on numerical approximations. Challenges with long term predictions arise from numerical instability of the approximations, chaos of the system, and incomplete data. As a result, certain areas have much more accurate weather predictions than others due to data availability and reduced turbulence and complexity/chaos of the local weather patterns.

Methods

We specifically wanted to predict the snowfall at Alta ski park in Utah. Dane will be working there this winter, and accurate snowfall predictions are crucial for awareness of hazards and optimizing ski conditions. More broadly, predicting snowfall in mountain regions is becoming increasingly important as weather patterns destabilize and many regions from Tibet to the Pacific Northwest are faced with summer droughts ameliorated mainly by winter snowpack.

We decided to design our own model rather than plugging our data into a preexisting model using a Long Short Term Memory Network (LSTM) neural network. LSTMs are used for a variety of applications, particularly making predictions based on time series, making it a great candidate for our model. LSTMs were first developed as a modification of Recurrent Neural Networks to protect against self perpetuating deviations in model training.

The basic concept of the model is that it has a certain “memory frame” which can be up to thousands of timepoints long – hence *long* short term memory. From this memory frame, the network determines which datapoints to keep or discard as it trains the model. The final model can be quite large – ours ended up with over 150,000 parameters – making it challenging to directly interpret. This being said, LSTMs have shown major promise in fields ranging from robot control to sign language interpretation, to modeling subcellular processes.

Process

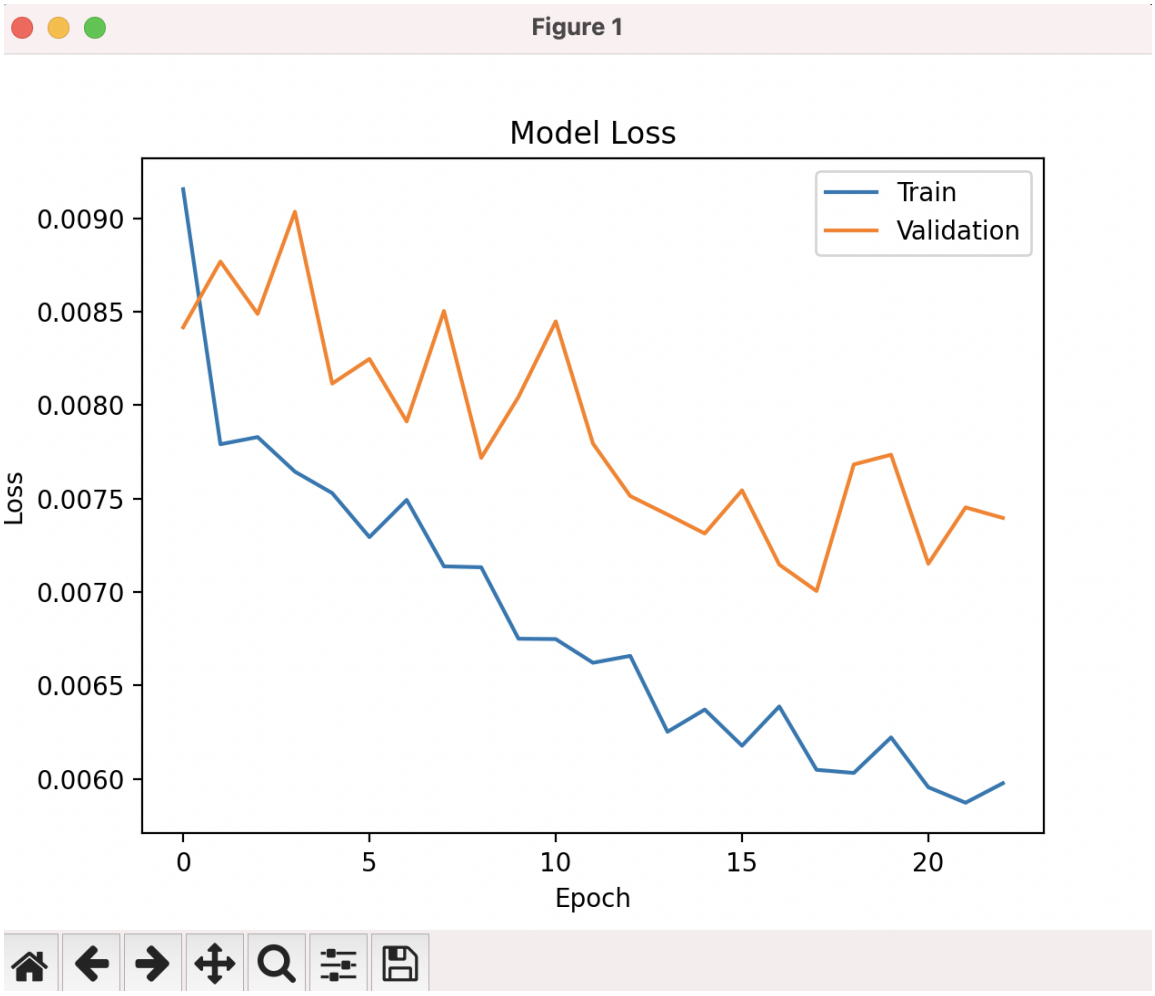
The first step of our process was cleaning up the data. We downloaded the data from mesowest; however, the data was not usable in the provided format.

We first had to get all of our datasets in the same format so they could be merged together. In other words, we needed to make the index of our data, the hourly datetime in the UTC timezone. Some of the data was recorded using the Mountain time zone so we had to

adjust it to the UTC standard. Another issue was that some of our data was in 10 minute intervals instead of hourly. Luckily the pandas python module has tools to help with this. When you have a pandas dataframe with a datetime index, you can use the resample function to change the data from 10 minute intervals to 1 hour intervals. It also allowed us to specify how we wanted our data congregated. For example, for snow gain we summed the values, for temperature, we got the mean, and for max wind speed, we got the maximum value.

Another issue we ran into was incomplete data. Luckily there are functions in pandas that can help with that. We were able to fill in empty values with a method that we specified. When there was a relatively small amount of missing values, we would use the forward fill method, which fills the data based on the value of the time before. For other values we used different methods. Like for Cloud Coverage, we filled the empty values with the default value of 1.0.

After we had the data set up, we set up the LSTM model. First of all, we split the data into training and testing sets (80% training, 20% testing). We set up the model with the mean squared error loss function. The model works by iterating through different weights to minimize the loss. This is shown by this graph:



As you can see, the model iterates and changes weights in order to try to minimize the loss. After this the model is trained and we can test it against our test dataset. We will talk more about that in our results section.

Data

To get the data required to train this model, we used the website https://mesowest.utah.edu/. This website has data available from many weather stations in the state of Utah. We decided to use data from 6 different weather stations surrounding Alta:

- ADG - Alta Guard House
  - This is a weather station at Alta. This is where we collected the snow amount, which was the goal data value of our training. We chose to use this one because it gives us weather data at Alta.
  - Variables collected: Snow, Temperature, Relative Humidity, Wind Speed, Wind Direction, Wind Gust
- HKCR- Heber Valley Airport
  - This is an airport that is southeast of Alta. We found that airport stations provide information on Cloud layers. We thought that would be important with snow prediction so we decided to include data from airports surrounding Alta.
  - Variable collected: Pressure, Temp, Humidity, Wind Speed, Cloud layer 1, Cloud layer 2, Cloud layer 3, visibility
- KU42- South Valley Regional Airport.
  - This is an airport that is directly west of Alta.
  - Variable collected: Pressure, Temp, Humidity, Wind Speed, Cloud layer 1, Cloud layer 2, Cloud layer 3, visibility
- KSLC - Salt Lake City International Airport
  - This is an airport northwest of Alta.
  - Variable collected: Pressure, Temp, Humidity, Wind Speed, Cloud layer 1, Cloud layer 2, Cloud layer 3, visibility
- KPVU - Provo Airport
  - This is an airport South of Alta
  - Variable collected: Pressure, Temp, Humidity, Wind Speed, Cloud layer 1, Cloud layer 2, Cloud layer 3, visibility
- F8379 - Park City
  - This is a weather station in Park city which is closely north of Alta
  - Variables Collected: Temp, Press, Humidity, Wind Speed

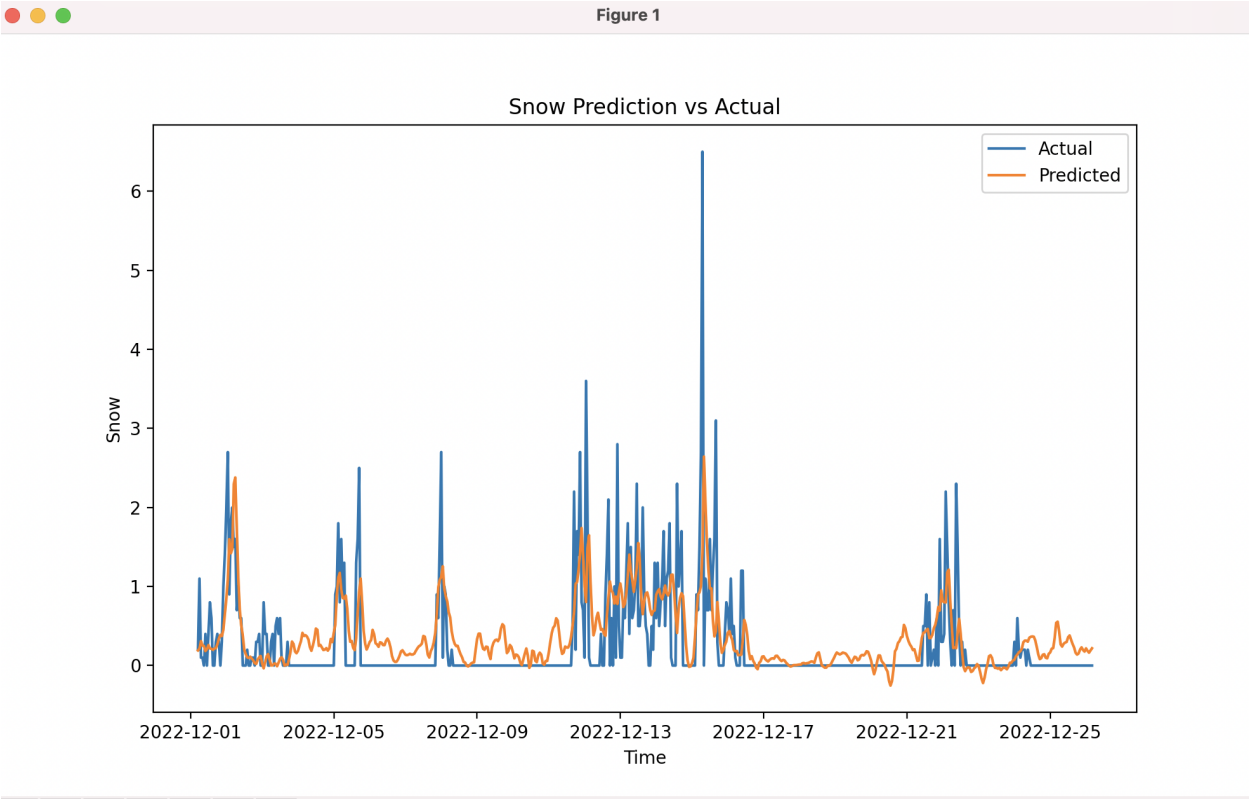
We used hourly data for all of these variables over the span of December 1st 2022 - April 20th 2023 to train our model. Overall there were 42 variables

Results

After we had our model trained, we tested the predicted vs Actual for our test data. We calculated the following error values:
Mean Absolute Error (MAE): 0.0516

Mean Squared Error (MSE): 0.00708
Root Mean Squared Error (RMSE): 0.0841

Here is a graph of our Prediction vs the Actual Snowfall.



Potential Improvements:

Our model is far from perfect. One slightly surprising outcome was that when we constrained the model to only consider the past five hours when making a prediction, it was more accurate than when we allowed it to use the past ten hours. This suggests that a major limitation might be the amount of training data we're providing the model with – we don't have enough training to effectively optimize parameters for the more complex ten day model. Unfortunately, cleaning the data is a major time suck, making adding more training data to the model a non-trivial task. Additionally, including a larger number of years decreases the number of parameters we can consistently provide as – for example – some weather stations only recently started including data on cloud cover.

This leads us to our second major hurdle, which is data availability. We ended up looking at several different websites and weather stations to try to find the most comprehensive data sets within the area; however, there is always more data which could be included. Most meteorological models also have significant data quality assurance – filtering out outliers and unrealistic data not mirrored by nearby stations. With our lack of expertise, we were mostly unable to do this, making our dataset much messier.

We also picked a challenging target – snowfall is very discrete. For most time intervals, there is no snowfall at all. Occasionally, there will be significant snowfall in a single interval. Compared to a continuous measure such as temperature, this yields very spiky and challenging to approximate target patterns. Potentially, looking at cumulative snowfall would have partially solved this – our predictions would end up looking something like a smoothed version of reality. For this to make sense though, we would ideally look at snow pack rather than simply cumulative snowfall – which would require us to consider settling, melting, and other factors which cause the snow pack to be lower than the cumulative snowfall.