CS230

CS 230 Project Final Report: United States City Temperature Forecasting

Tingkang Wang
Stanford University
wangtk@stanford.edu

Yitian Liang
Stanford University
ytliang@stanford.edu

Zixin Huang
Stanford University
alisa612@stanford.edu

Abstract

This project aims to build precise model to predict temperature of 210 cities in the United States to study the climate dynamics and explore the relationship between location and temperature. The project used the bidirectional LSTM model that incorporates location information to predict one next year's temperature from the input's time. We obtained daily temperature mean square error of less than 5 degrees Celsius for all 210 cities in the next one year.

1 Introduction

With intense artificial intelligence development these years, new technologies can be deployed and applied on many aspects to help tackle problems in many fields. This project will investigate the temperature trend of many major cities in the United States by building and experiment with deep learning model. We would like to make a model for daily temperature forecasting in many different places in the United States. Both time axis and location space axis are taken into account as input. The input would be a dataset consisting of 210 different cities in the United States and their corresponding approximate 120 years daily temperature. The model we experienced incorporates Recurrent Neural Network(RNN) architectures and location features to output next one year's daily temperature across 210 cities in the United States. By predicting temperature in the future in some modern and densely populated cities in the United States, the prediction can help people understand climate dynamics and develop policies in accordance with that.

2 Related work

Temperature forecasting is a popular topic in the deep learning field since it can be really helpful to study climate dynamics and its data sources are vast and relative easy to obtain. We found many papers relative to this topic and we are going to discuss some of them and compare their methods with ours in this section. Some uses purely RNN model like LSTM with some fully connected layers as architecture of the model(4)(3). We believe it is the most traditional way of making temperature forecasting since temperature in one place follows a relatively regular shape. Some paper used CRNN model which transfer temperature data among different cities in the country into heatmap at different time and then use CNN architecture to study and output a predicted heatmap.(2) We tried this method but later found that our dataset contains only 210 cities which leave many spaces in the heatmap empty. It is also hard to interpolate these data because these cities are not distributed in the United States uniformly as well. However, we do study from this paper by deciding to incorporate location information as part of the input for our model. We also found paper that incorporate both CNN and RNN features to build the model(5)(6). The paper studying the temperature change in sea surface, "A Novel Method for Sea Surface Temperature Prediction Based on Deep Learning"(5) utilize one location and some surrounding points' temperature data and cooperating them with CNN feature to build the model and make predictions. This method is similar to what we used. CNN is good at capturing local dependencies between features while RNN can be usable in modeling temporal dependencies in time series. We found this kind of method which utilizes both location and time series data can be used to perform a more wide range of tasks with improved accuracy and this is the method we used in this project.

3 Dataset and Features

The project plans to mainly utilize the dataset from KiltHub named "Compiled historical daily temperature and precipitation data for selected 210 U.S. cities"(1). We perform the test with all 210 cities data across 120 years. The main reason that we use this dataset is that it has daily temperature lasting for about 120 years, which serves as enough data for a valid prediction (2). On the other hand, by studying the temperature trend in U.S., we would hopefully see the general global climate change trend as well. This dataset also contains the highest temperature and lowest temperature of the day and precipitation which would be helpful for catching some other predictable features.

The dataset contains ID and geographical information (latitude, longitude) on 210 U.S. cities with separate .csv files named by corresponding ID. Each .csv file detailedly records the daily highest and lowest temperature as well as precipitation history in the last 120 years. Missing data points are marked as NA. Example plotting of data in the time axis and spacial axis are shown below. Figure 1 reveals obvious seasonal changes in daily average temperature (°F) with the data for a single city in several years. Figure 2 exhibits the effect of geographic variation on temperature using the daily average temperature (°C) for the same day in different U.S. cities. We can clearly see the trend that the farther a city is from the equator, the lower its daily average temperature.

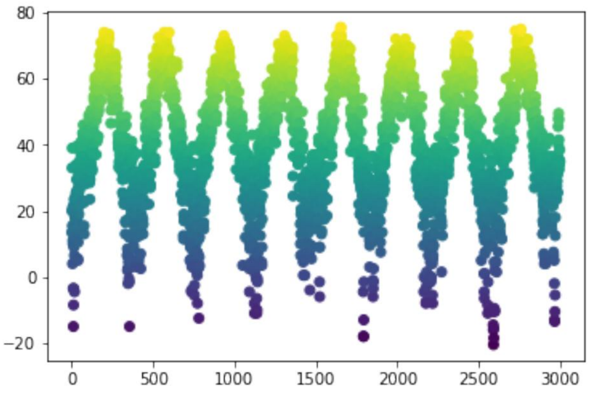

Figure 1. Single City Temporal Temperature Data Plot

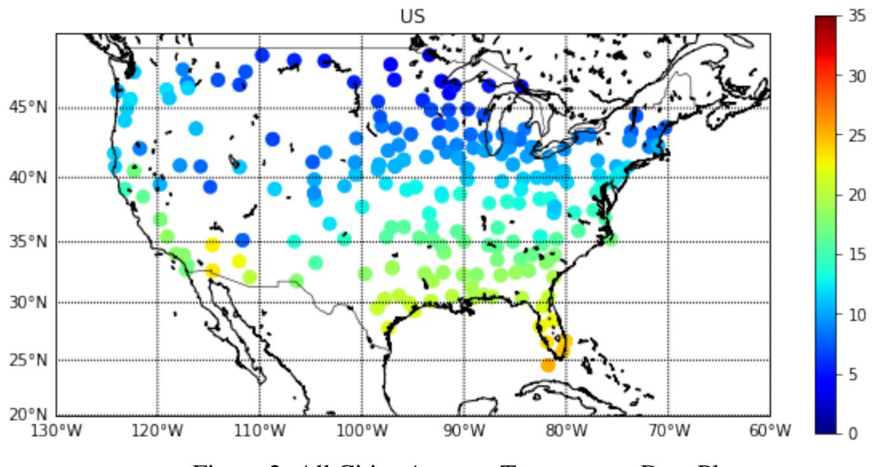

Figure 2. All Cities Average Temperature Data Plot

To utilize the dataset on our model, we use the sliding window method to iterate over our data. The sliding window step size is 90 days. On the other hand, there are some "nan" values distributed in different time period in different cities. Before input these data to the model, we preprocess them by dispose any window that contains more than 10 continuous "nan" value. When we encounter these windows, we would ignore them and slide to the next. Each window contains 2 years data, among them, the first year's data is served as input and the second year's data is the output. By doing so, the model would satisfy our problem statement of predicting the next year's temperature given the previous year's data as we trained it. Normalization has also been performed to make the training process more efficient. We slide per 90 days each time across approximate 80 years data as our training data and test use the next 40 years.

4 Methods

4.1 Baseline Model

We have tried to train 3 different RNNs with different architecture as the baseline model, including a simple RNN , seq2seq and LSTM model with ReLU as activation function and RMS loss as loss function for optimization. Among them, the seq2seq model performs the best as baseline. Its RMS loss function while training is shown below.

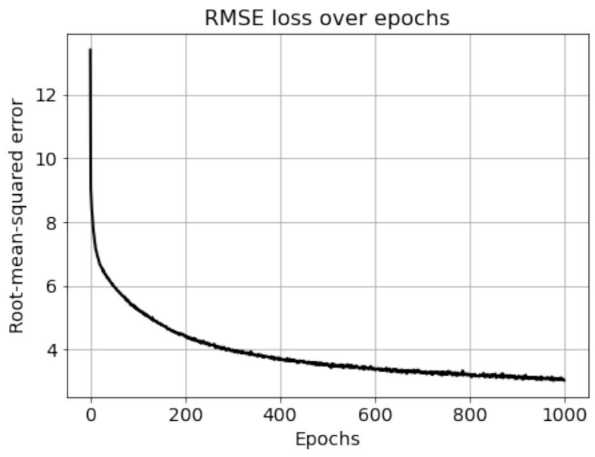

Figure 3. Seq2seq RNN Model RMS Loss Plot

We have also plotted the comparison between ground truth of the testing data and the predicted data for the seq2seq model. In this graph, the part before the red line is the trained part while afterwards is the predicted results.

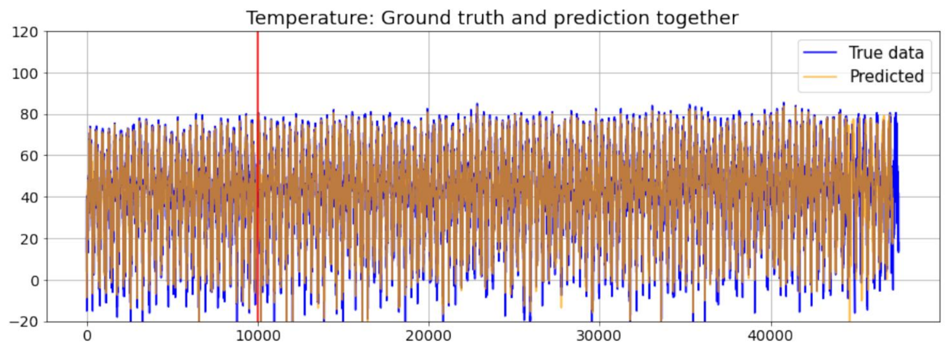

Figure 4. Comparison between ground truth and prediction

The result is relatively consistent but have many potential aspects for improvement.

4.2 Architecture

We mainly experienced with two models which are unidirectional LSTM and bidirectional LSTM. After many experiences, we found that the bidirectional LSTM model performs much better since it utilize the past one year data more efficiently to be able to train the model more precise.

The bidirectional LSTM model first contains the LSTM layer for the input of temporal data. Then, there is a repeat vector layer for the incorporation of location information. We used repeat vector because the location information for one city won't change across one year. After concatenate them, we pass the input to a dense layer and another LSTM layer. At last, there is one dense layer for the output. We have also tested with the dropout layer but it performs worse that given us a higher training error of around 7 degree Celsius. The general architecture of the bidirectional LSTM model is shown below.

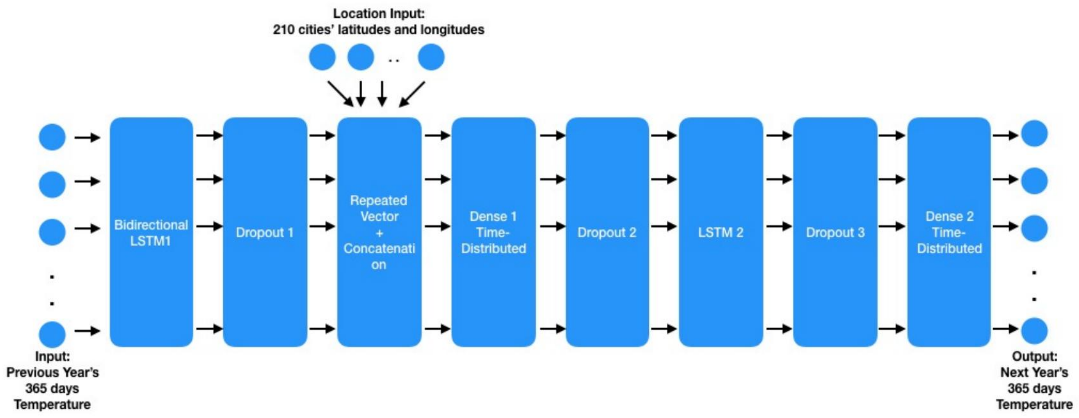
Location Input:
210 cities' latitudes and longitudes
Bidirectional LSTM1
Dropout 1
Repeated Vector + Concatenation
Dense 1 Time-Distributed
Dropout 2
LSTM 2
Dropout 3
Dense 2 Time-Distributed
Input: Previous Year's 365 days Temperature
Output: Next Year's 365 days Temperature

Figure 5. Architecture of Bidirectional LSTM Model

5 Results

5.1 Hyperparameter

LSTM layer 1,	neuron number: 20
Dropout layer $1, 2, 3$	dropout rate: 40%
Dense layer 1,2	hidden unit number: 5
Optimizer	learning rate: $0.001, \beta_1 = 0.9, \beta_2 = 0.999, \varepsilon = 10^{-7}$
Compilation	Batch size: 128, Epochs: 10

5.2 Evaluation Baseline

To set a reference for evaluation, we calculate one simple RMSE error from directly copying the temperature data from the previous year as the prediction of the next year as our evaluation baseline. The error calculated through this native method is around 6 degree Celsius. As a result, we expect our model's performance to exceed this native model.

5.3 Evaluation

The evaluation metrics we used are average root mean square error(RMSE). It is a very common metric for regression model. Its formula is shown below.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - y)^2}$$

In the formula, n is the number of samples, y_i is the output temperature and $\tilde{y}$ is the average of temperature evaluated while training.

When we evaluate the unidirectional LSTM model, we use the past 365 days data to predict the next one day's data and them concatenate them to the new input for prediction for the next day. As a result, we found that this way of prediction is not accurate and not efficient that it outputs a average mean square training error of around 16.7 degree Celsius which makes it a very inaccurate model.

We changed the method of evaluation when we evaluate the bidirectional LSTM model by inputting the 1 years' data and ask it to perform the sliding window prediction as described in the dataset section. The final training RMSE error is 4.73 degree Celsius and the test RMSE error is around 4.64 degree Celsius. The predicted outputs is shown below.

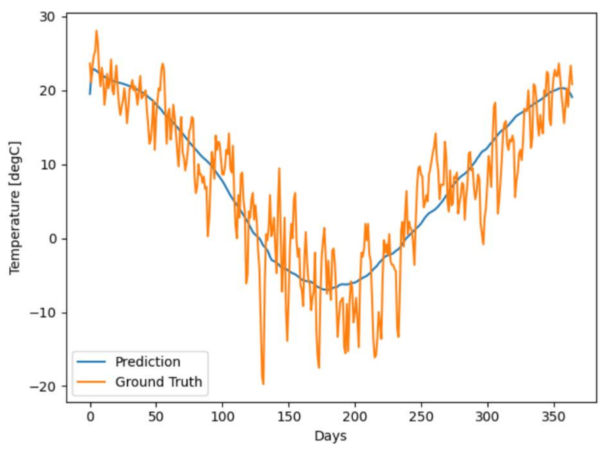

Figure 6. Bidirectional LSTM Model Result with 90 days step size

During the process of tuning our model, we have experienced overfitting when the sliding step size is too big which results in too less data points for training. For example, when the step size is 2 years, we get training RMSE of 0.7 degree Celsius but 6.237 degree Celsius for test RMSE. The plot is shown below. We can clearly see that there is overfitting since the prediction line is vibrating a lot.

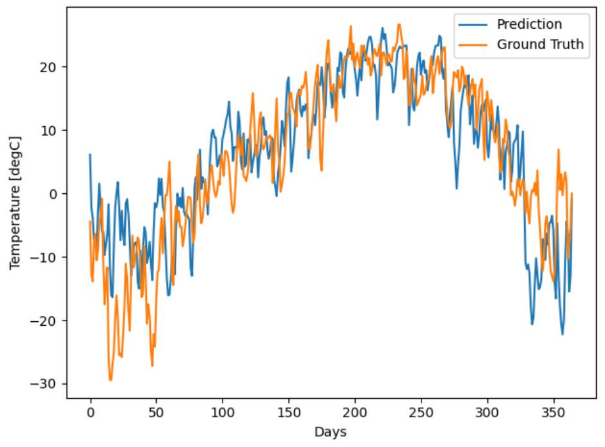

Figure 6. Bidirectional LSTM Model Result with 2 year step size

6 Conclusion/Future Work

In this project, we incorporate temporal data and location data to make temperature forecasting across 210 different cities in the United States. The bidirectional LSTM model with sliding window data input performs very well. We believe since relative data sources are vast and easy to obtain, it is better to utilize both temporal and local data in many fields to improve the model. For future work, more fine tuning such as experience with shorter sliding step size can be perform. More data can be collected and used to train the model as well. If we have enough data, we would probably test the CRNN model with heatmap as well.

7 Contributions

We work together to propose the topic and find the dataset. Zixin starts the project by experiencing with different baseline models and visualizing the dataset. Tingkang explored the algorithm and build the model. Yitian fine tuned the model and output the final result. We work on the video and report together.

References

[1] Yuchuan Lai (2021) *Compiled historical daily temperature and precipitation data for selected 210 U.S. cities*, kilthub.

[2] Zao Zhang, Yuan Dong(2020) *Temperature Forecasting via Convolutional Recurrent Neural Networks Based on Time-Series Data*, Wiley, Hindawi Complexity.

[3] Ben Abdel Ouahab Ikram, Boudhir Anouar Abdelhakim, Bassam Zafar, Bouhorma Mohammad, Astito Abdelali (2019) *Deep Learning architecture for temperature forecasting in an IoT LoRa based system*, Research Gate.

[4] Sungjae Lee, Yung-Seop Lee and Youngdoo Son(2020) *Forecasting Daily Temperatures With Different Time Interval Data Using Deep Neural Networks*, MDPI, applied sciences.

[5] Xuan Yu, Suixiang Shi, Lingyu Xu, Yaya Liu, Qingsheng Miao and Miao Sun (2020) *A Novel Method for Sea Surface Temperature Prediction Based on Deep Learning*, Wiley, Hindawi Complexity.

[6] Debneil Saha Roy (2020) *Forecasting The Air Temperature at a Weather Station Using Deep Neural Networks*, ScienceDirect, Procedia Computer Science.