AI Hackathon ISEP X GarageISEP: Weather Prediction for Agriculture (Approach)

Prem Prakash [1] **Ishwar Purushotham** [1] **Navdeep Kumar** [1] **Arunava Maulik** [1] **Rohil Gupta** [1]

Abstract

Weather is of prime concern for farmers. The quantity, quality, and survival of farmland is directly contingent on weather. A good and consistent weather according can enhance the output while on the other hand a sudden variation in weather can negatively impact the farming. In this hackathon we tackle the problem of weather forecast for next couple of days given the observed weather condition upto current time. Weather is a sequential information and is dependent on time thus we treat this as sequence-to-sequence problem which takes an observed weather sequence as input and outputs a sequence for forecast.

1. Introduction

This report is for the hackathon challenge organized by GarageISEP, Paris. A team composed of five persons were involved in the competition. This report primarily focus on modelling of weather prediction from the observed data. We take observed data from last few days (recommended 48 hours or more) and predict for next few days (again recommended for 24 hours or more). It is important to note that we only observed data as constrained by time. To model such problem we will use sequence-to-sequence Recurrent Neural Network (RNN) architecture also called many-to-many modelling.

2. Dataset

We have used weather forecast dataset coming from four different weather forecast stations. However, we have noticed that data from only two sources are reliable and thus only these two sources were retained. Each station receives forcast data from seven different weather forcast providers. However, we do not use forcast data for our model but can be used in future works. A sample fo extracted observed data from all the weather stations are given in Table 1.

Timestamp	Station Name	Humid	Temp	Precip
1551349357	La Tuilerie	61	16.9	0
1551350284	La Tuilerie	57	17.8	0
1551351210	La Tuilerie	52	19.2	0
1551352136	La Tuilerie	45	20.7	0
1551353063	La Tuilerie	41	21.8	0
1551353988	La Tuilerie	39	22.2	0
1551354915	La Tuilerie	37	22.3	0
1551356769	La Tuilerie	36	24.1	0
1551357695	La Tuilerie	30	24.1	0

Table 1. Observed data by a **La Tuilerie** weather station taken every 15 minutes.

3. Preprocessing of Dataset

Data from each weather stations are transformed in sequence of data which can be fed to RNN model. If we want to form a sequence of two days than we pick $48 * 4$ i.e. 192 rows continuously to form a sequence as there are four observations per hour. For sake of simplicity we assume that we have four observation per hours across all weather stations else need to handle differently. It is also important to note that we take only humidity, temperature, and precipitation as a feature for each time stamp. Picking as sequence of observations (192 for two days) only form the input data going in the model while the target is formed similarly starting from next observation for next one day (96), or two days (192), or any other size as preferred. As such accurate prediction by model can help farmers take preemptive actions to save their farmed crops from frost, rain etc.

4. Technologies used

To solve this task we use Python as a programming language along with other supported libraries such as Pandas, Numpy, matplotlib etc. For implementing sequential model we have used the deep learning framework Pytorch. SInce the amount of data is not big enough thus the model is trained locally on computer. Nevertheless, the model provide support for training on GPU if needed.

[1]Machine Learning and Data Mining (MLDM), University of Jean Monnet, Saint-Etienne, France. Correspondence to: Prem Prakash <prem4708@gmail.com>, Ishwar Purushotham <ishwar.purushotham@gmail.com>.

5. Methodology

We use sequence-to-sequence (Sutskever et al., 2014) modelling for out task which takes a sequence of length n as $X_t \cdots X_{t+n}$ and produces an output of sequence length k as $Y_{t+n+1} \cdots Y_{t+n+k}$. Input sequence goes through encoder as given in Figure 1 whose sequence are encoded and this is used to produce a sequence of outputs using decoder as given in Figure 1. In the Figure 1 we can observe that the first input in decoder is a linear transformed of the output from last entry in the input sequence and starting hidden state of decoder is hidden state output from last position of input sequence. As this allows the first input sequence of decoder to be not assigned zero.

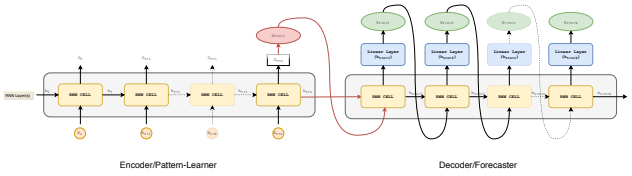

Figure 1. Sequence-to-Sequence RNN model.

In the Figure 1 each of RNN cel can be from vanilla form (Figure 2)

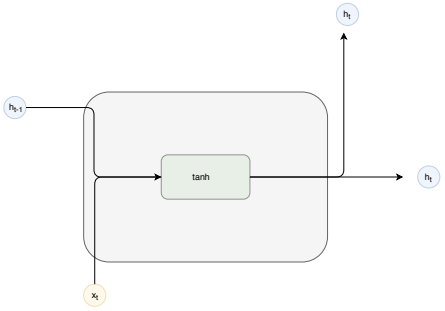

Figure 2. Vanilla RNN

to Gated Recurrent Unit (GRU) as given in Figure 3 which works best for out problem. Nevertheless, one can choose any among various RNN architecture available some of the most used ones are LSTM and GRU.

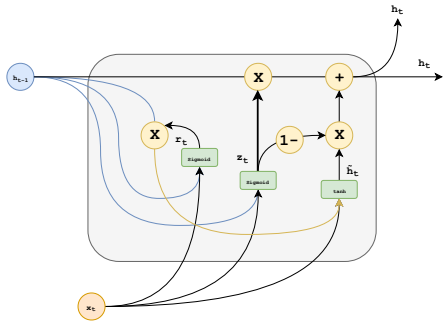

Figure 3. GRU architecture.

GRU (Chung et al., 2014; gru, 2017) can be understood from its multi-gating structure. These gates help keep around memories to capture the dependencies in sequential data. One of the intuitions why GRU is able to capture long-term dependencies can be understood from its gating structure and operation it performs. For example, sigmoid clamp down the data between zero and one which helps in how much of data from last hidden state (h_{t-1}) should be passed on for next time step (h_t). This is shown by reset gate (r_t) whose value affects new memory generated. For instance, if reset gate sets some units (neurons) to zeros then for those units only the new memory ($\tilde{h}_t$) content from current time-step data is kept. The update gate (z_t) in the equation does the job of combining the old memory (h_{t1}) and generated new memory content ($\tilde{h}_t$) which is given by the final memory (h_t). For example, if the update gate output is all ones then, in that case, the final memory will be only from previous time steps and new memory content will be ignored. All these operations help capture sequential information in the data. Thus can learn pattern in last couple of days and how that pattern is able to say something about future timesteps.

6. Experiments

In this only the working of architecture is checked which takes sequence as input and outputs sequence. It is important to note that input and output sequence length can vary. To compute the error between forecast and target we use mean square loss (MSE) as we are treating this with regression problem since the task is to forecast temperature, rain etc. Figure 4 presents the training of data constructed on 48 hours of past data and predicting for next 12 hours. This figure provide a step towards and confidence to further investigate the architecture.

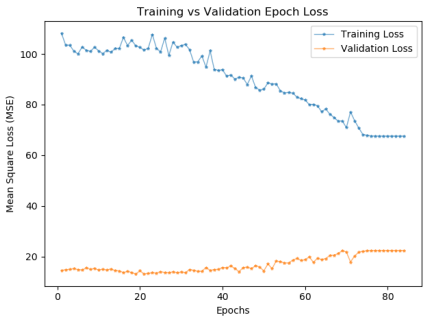

Figure 4. Training vs Validation MSE loss

7. Conclusion

In this hackathon we undertook the task of weather forecast from observed weather condition using sequence-to-sequence learning. The preliminary experiment seems to be promising and is capable of producing better result in presence of sufficient data.

8. Future Works

This project can be further extended in presence of more data. For better training the model we need data for at least two years of observed data. Moreover, the forcast data from multiple sources can also be used if needed and fruitful to further improve the generalization of model. Apart from this the sequence-to-sequence architecture can be also tweaked to compare performances.

Acknowledgements

We would like to thank Hackathon AI ISEP for providing this opportunity to participate from 29^{th}-31^{st} march at Innovation Hub, Garage ISep, ssy-les-Moulineaux, Paris.

References

Lecture 9: Machine Translation and Advanced Recurrent LSTMs and GRUs. https://www.youtube.com/watch?v=QuELiw8tbx8, 2017.

Chung, J., Gülçehre, Ç., Cho, K., and Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014. URL http://arxiv.org/abs/1412.3555.

Sutskever, I., Vinyals, O., and V Le, Q. Sequence to sequence learning with neural networks. pp. 10, 09 2014.