Middleware Network and Distribution System

PROJECT REPORT - Detection and Prediction System Integrated on Scale

Group Members: Xuan Shi, Yuxuan Hao, Yuanzhe Li

1 Introduction

Our distributed system project is Weather prediction and Visualization based on SCALE system. Safe Community Awareness and Alerting Network (SCALE) is a novel networking technology concept[1]. Scale system provides us a middleware platform to post and gather many weather data such as temperature and humidity. One SCALE box is implemented by us with DHT11 temperature and humidity sensor and raspberry pi 3B. It could be deployed at anywhere to detect weather data and send temperature and humidity data to SCALE broker. Then, our middleware system will predict weather with SCALEąś weather data fetched from SCALE broker and air pressure data fetched from weather website, then send new predicted weather data to broker. Prediction mode is built with LSTM, a special RNN structure. One-year weather history data is crawled off and stored in AWS database as the training data. The input data is the last 24-hour weather data, containing temperature, pressure, humidity and condition. Finally, after prediction via our prediction middleware, predicted data is sent to SCALE broker which could be visualized and shown in SCALE dashboard and stored into our AWS database for future use.

2 Related Efforts

2.1 Scale System

Safe Community Awareness and Alerting Network (SCALE) is a novel networking technology concept[1]. SCALE aims to improve the safety of residents and it leverages the pervasive Internet of Things to extend a smarter and safe home with a low cost. SCALE contains many kinds of novel technologies such as networking, commodity senor devices, cloud services and middleware abstractions.

FlexSCALE is a sensor box which can support many different sensors and network adapters. In such sensor box, Raspberry Pi are used as the compute unit to control other sensors via I/O ports and pins, and Debian Linux system run on Raspberry Pi. In this project, one SCALE client is implemented with raspberry pi 3B and temperature and humidity sensor. Data is collected by SCALE device and sent to SCALE broker.

Data in Motion Exchange (DIME) system is proposed to act as an open communications hub for IoT that simplifies the development and deployment processes[2]. In the DIME system, JSON is used to format data, MQTT is used as machine to machine protocol, which allows multiple servers to collect data from DIME and multiple clients to send it without requiring any configuration.

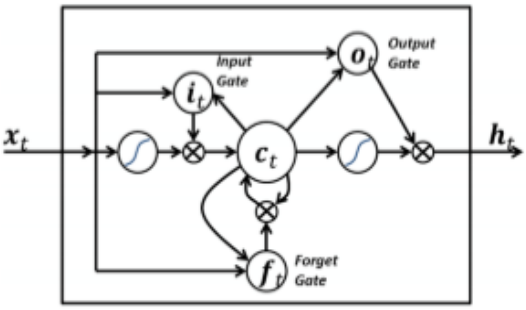
An asynchronous event-driven Python is deployed on the IBM BlueMix platform, receives sensed data through Eclipse Pahoąś MQTT client and routes it to the appropriate eventdetection function. Celery task queue manager is used to handle distributed system event. Djangoąś ORM is used to abstract the PostgreSQL database tables.

The dashboard, hosted on BlueMix, is built on top of software designed by BioBright. Dashboard is written using Node.js at the backend, Javascript and Twitter Bootstrap in the front-end. Data in broker is posted on Dashboard. Temperature and humidity data on our SCALE device could be easily viewed in Dashboard in real-time.

2.2 Some Prediction Algorithms

In choosing prediction algorithm, we did survey on three methods, which are Black-box modeling[3], Clustering-based feature selection for black-box[4], and LSTM.

LSTM represents for ąřlong short-term memoryąś, which is proposed by S. Hochreiter and J. Schmidhuber in 1997.[5] The term long short-term refers to the fact that LSTM is a model for the short-term memory which can last for a long period of time. A simple LSTM unit containing intput, output and gates is illustrated as the following figure. For general-purpose sequence modeling, LSTM as a special RNN structure has shown stable and powerful performance. This is very suitable to classify, process and predict time series. Therefore, this can be applied to time series forecasting, where classical linear methods can be difficult to adapt to multivariate or multiple input forecasting problems.



Mohamed and Chakerąŕs used LSTM on sequence to sequence weather forecasting.[6] The result shows that LSTM can provide competitive performance compared to tradition linear methods, besides, it can even become a better alternative. Similarly, Xingjian, el have successfully applied LSTM to the chanllenging precipitation nowcasting, which has not been solved even by sophisticated machine learning techniques.[7] In this project, we aim to predict future weather conditions, which can not be detected by sensors directly. Since this is well-suited for time series model, we choose to implement this based on LSTM as well.

3 Implementation

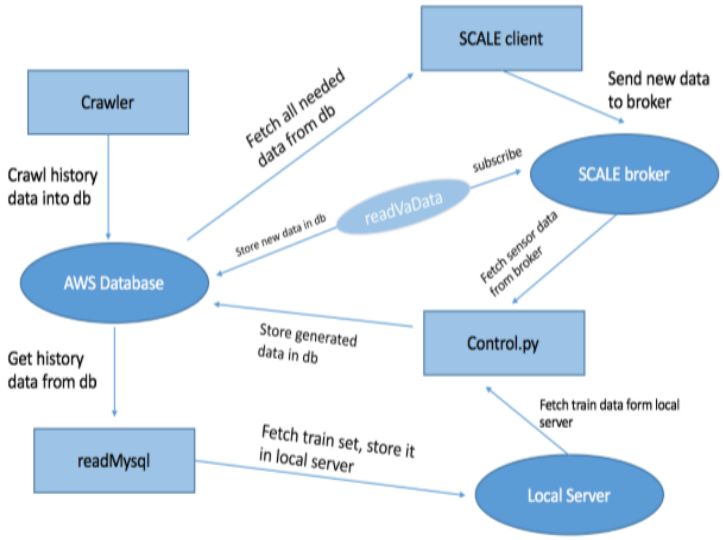
3.1 Structure of the Prediction System

In this project, we integrate the prediction system on SCALE system. Since the sources of data are different, there are totally three ąřserverąś in our system to store data. We will explain why we need these parts in details below. Due to the limitation of our permissions provided by AWS service, we can only use AWS database service, which means we need to store some of our results on local service temporarily. The general structure is as the following.

There are mainly 8 components in this system.

- **Crawler**. This refers to the program crawler.py in our implementation. It crawls one-year history data from public website (WeatherUnderground.com), and then store crawled data in AWS database.
- **AWS database**. Due to the limitation of our permissions, we can only use the database service. But as the time goes, there are new hourly data generated, it will be dangerous or inconvenient to write in database immediately. Thatąŕs why we need a local server to store a single csv file containing daily data.

- **Local server**. The need of local server was explained above. We need it to store single csv file containning daily data, in order to avoid data changing issues in the database.
- **SCALE broker**. This is the middleware of SCALE system. We send generated data to broker from the SCALE client.
- **SCALE client**. We write codes to fetch data from broker and send generated data to broker in the client program.
- **readMysql**. This refers to the program ąřreadMysql.pyąś in our implementation. We read history data daily from AWS database and store it on the local server. Besides, it is controled by the part ąřControląś, explained as the following.
- **readVaData**. Here, we also write a python script ąřreadVaData.pyąś to subscribe the broker to fetch the newest data and write fetched data into the database.
- **Control**. This refers to the program ąřcontrol.pyąś in our implementation, which acts as the master node in this system. Inside the control program, there is a clock, which decides what to do after specific time interval. This program reads history data form local server as the input of model training program; it also reads the last 24-hour data from the database as the input of model prediction program since we use the last 24-hour data to predict the future 6-hour data. The Trainning and Prediction program can only be activated by control.py. For example, the control.py will activate the trainning model once a day, and fit model to predict hourly data once an hour. After predicted data is generated, control program will store this new data into database for future use.



Generally, we use these separated compenents because of the complexity of the data flow patterns. The control program acts as a master node to coordinate between other components, providing a clock for this system. It is like a middleware between different servers. In addition, some other prediction, training, readMysql are responsible for specific functionalities, which are controled by the control component.

3.2 Physical Sensors

DHT11 digital temperature and humidity sensor is a composite Sensor contains a calibrated digital signal output of the temperature and humidity. The sensor includes a resistive sense of wet compo-

nents and an NTC temperature measurement devices, and connected with a highperformance 8-bit microcontroller.

DHT11 temperature and humidity sensor is connected to GPIO 17 port of raspberry pi 3B in order to detect temperature and humidity data in the environment around SCALE device. One new class, named myTemperatureSensor, is created based on temperature physical sensor class, which extends a VirtualSensor class in SCALE client. Read_raw method interface is implemented by importing Adafruit_DHT library[8], which is able to help format data. After formatting, temperature and humidity will be sent to broker middleware via MQTT.

3.3 Virtual Sensors

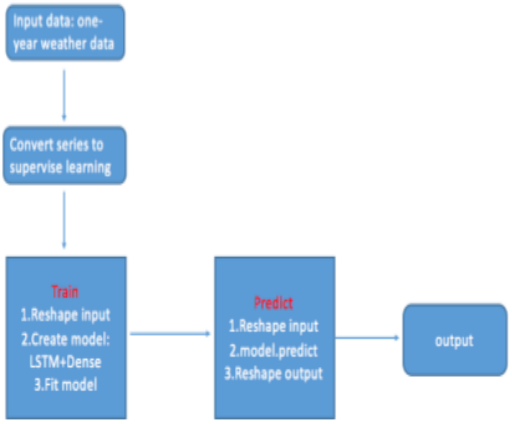
We need to use the virtual sensor for two reasons. The first reason is that after making the prediction, we want to push the prediction results into the broker. So that others can see the result of prediction. The second reason is that we need data for pressure and current weather condition to make the better prediction. However, we do not have sensors that can sense these data currently. Therefore, we need a virtual sensor to do these. To achieve that, we modify the scale client by adding three python files under the folder scale_clientscale_clientsensorsdummy. The three python files are condition_sensor, history_condition_sensor, and pressure_sensor. The condition_sensor reads the prediction results from mysql, and then post the data on the broker. It posts an event called condition, and the data is in the format "hour:data, hour:data, ...". History_condition_sensor collects current weather condition from weather underground website, and posts a current_condition event on the broker. Pressure_sensor collects current air pressure from weather underground website, and posts a pressure event on the broker.

3.4 Prediction

Firstly, we use Keras to implement the neural network, which is an integrated deep learning library based on Tensorflow.

Then, we refer to the algorithm in Zaytar and Amraniąŕs paper, but we simplified the neural network. We apply two layers: one LSTM layer and one Dense layer. The training data is from the history data, which is one-year set stored in the AWS database. The input data is the last 24-hour weather data, containing temperature, pressure, humidity and condition. Since the condition is discrete, we map it to 21 values, and the sequence is based on the relationship between weather conditions. The output is the future 6-hour weather condition data.

To make the structure more straightforward, we illustrate it through the following figure:



3.5 Mysql

We use a Mysql database to help the transmission of data from different elements. So that each element could run in a different platform. Since this Mysql is used in many cases, we need to make sure it is stable and always available to use. Therefore, we use the Mysql database on AWS. And we construct and use the Mysql database in AWS based on the official tutorial[9]. We create three databases in this Mysql, which are conditions, trData, and vaData. Database conditions contains a table called forecast, where the prediction result is stored in the format hour and cond. The hour column represents how many hours is the predicted result after. And the cond column is the predicted result. Database trData stores the training data for the prediction algorithm. We use a crawler program to crawl the historical hourly data in last 3000 days. The data in each day is stored in a table named as t_date, such as t_20180612. In each table, it contains data for hour, temperature, humidity, pressure, and weather condition. Database vaData stores the recent data read from the dashboard. The tables in this database are named as table_date, such as table_20180614. Each table contains data for hour, temperature, humidity, pressure, and weather condition.

4 Evaluation

4.1 Evaluation of Prediction Algorithm

To evaluate our prediction algorithm, we train the model with data in 365 days, and choose 30 days among them to test. The result shows that we have an MSE value of three. This is reasonable because we have 22 possible weather conditions. Each of them is represented with numbers from 0 to 21. And the adjacent numbers represent similar weather conditions. For example, cloudy and partly cloudy are next to each other. Therefore, the MSE of three represents that most results are in the range of [expected-1, expected+1], that means most predictions we make are accurate or almost accurate. Thus, we think we have good prediction results.

4.2 Evaluation of Sensors

In the evaluation of sensors, the most important thing we want to make sure is that whether the data are sent successfully. Therefore, we write a test program to check if the number of occurrence in the correct format is the same as expected number of occurrence for each event. To do that, we write a program to modify if any of correct-formatted data are received. Once each of them are received, we write something to a file. This program is run for three hours. Then, we write a program to analyze the result, output values including number of occurrence, average of intervals, and standard deviation of intervals.

	expected #	actural #	expected mean	actual mean	expected std	actual std
condition	560	449	10	12.41	0	8.78
pressure	560	450	10	12.43	0	8.78
forecast	1120	705	5	7.89	0	8.80
temperature	1120	752	5	7.56	0	8.73

From the analysis result, we can see that most events are sent and received successfully. Especially, the low standard deviation ensures that the interval between each two events are very close, so we do not need to worry about that we need to wait for a long time in some situations.

5 Conclusion and Future Works

Weather prediction middleware system is developed in our middleware project.. Scale system provides us a middleware platform to post and gather many weather data such as temperature

and humidity. One SCALE box is implemented by us with DHT11 temperature and humidity sensor and raspberry pi 3B. A new python class, named myTemperatureSensor is developed with Adafruit_DHT library, which formatting raw digital sensor data. SCALE device could be deployed at anywhere to detect weather data and send temperature and humidity data to SCALE broker via MQTT. Then, weather prediction is made by our middleware system from last 24-hour weather data fetched on broker and new predicted data is sent back to broker. Prediction mode is built with LSTM, a special RNN structure. One-year weather history data is crawled off and stored in AWS database as the training data. The input data is the last 24-hour weather data, containing temperature, pressure, humidity and condition. Finally, after prediction via our prediction middleware, predicted data is sent to SCALE broker which could be visualized and shown in SCALE dashboard and stored into our AWS database for future use.

Future research directions could bečž first, enhance data exchanging or flexibility of SCALE system by proposing and developing a multi-protocol distributed broker middleware; second, adopt multiple sensing technologies and human provided input (wearable and mobile sensors, social media) can be integrated into the SCALE platform for improved awareness. Third, SCALE device could be integrated into IC chips, which has a smaller size and stability and safety.

References

[1] Benson, Kyle, et al. "Scale: Safe community awareness and alerting leveraging the internet of things." IEEE Communications Magazine 53.12 (2015): 27-34.

[2] Uddin, Md Yusuf Sarwar, et al. "The scale2 multi-network architecture for IoT-based resilient communities." Smart Computing (SMARTCOMP), 2016 IEEE International Conference on. IEEE, 2016.

[3] Zahra Karevan, Siamak Mehrkanoon, lohan A.K. Suykens, ąřBlack-box modeling for temperature prediction in weather forecastingąś IEEE, 2015

[4] Zahra Karevan, lohan A.K. Suykens, ąśClustering-based feature selection for black-box weather temperature predictionąś, IEEE, 2016

[5] Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural computation, 1997, 9(8): 17351780.

[6] Zaytar M A, El Amrani C E. Sequence to sequence weather forecasting with long short term memory recurrent neural networks[J]. Int J Comput Appl, 2016, 143(11).

[7] Xingjian S H I, Chen Z, Wang H, et al. Convolutional LSTM network: A machine learning approach for precipitation nowcasting[C] Advances in neural information processing systems. 2015: 802-810.

[8] "Raspberry Pi: Measure Humidity and Temperature with DHT11/DHT22." Raspberry Pi Tutorial, Raspberry Pi, tutorials-raspberrypi.com/raspberry-pi-measure-humidity-temperature-dht11-dht22/.

[9] "How to Create a MySQL Database ÍC Amazon Web Services." Amazon, Amazon, aws.amazon.com/getting-started/tutorials/create-mysql-db/?nc1=h_ls.