Boston University
Electrical & Computer Engineering
EC464 Capstone Senior Design Project

Final Testing Test Report

5G Network Performance Testing - Sky Seer

Team #16
5G Drone Team Red

Team Members:

Ryan Mondonedo, mondor28@bu.edu
Peter Wallace, peterjw@bu.edu
Jong Whee Jeon, crm04140@bu.edu
Dohun Ji, kaahsh38@bu.edu
Jungin Chang, changju@bu.edu

Setup and Materials

The setup that was carried out was consistent with our test plan. Our Android phone was connected to AT&T's 5G network and we ran our application to collect upload speed, download speed, latitude, longitude, and gps altitude. Since our second prototype testing, we have made additional progress on the web application and machine learning model and added an additional altitude zeroing feature to the mobile application.

The physical materials we needed were our Android phone and laptops. The phone had our Android app installed on it and the laptops were needed to run our web app for controlling data collection and visualizing results, viewing our database, running the machine learning model, and showing the video of our drone.

The software materials were our Firebase Firestore database, PyCharm for running our Python machine learning code, our Android app, and our web app. As mentioned in our second prototype testing, the GPS altitude that the Android Location library gives is the altitude in relation to the WGS84 ellipsoid, a mathematical representation of the surface of the Earth that is inconsistent in some places with the mean sea-level altitude, which is exact. This is due to the inconsistency in the Earth's curvature, and can be corrected by using an aviation-grade altimeter that uses barometric pressure. This is not feasible for our project due to budget constraints, runtime constraints, and hardware constraints (we are limited to our phone's low quality barometer and a small drone). As a result, even before the test, we knew that the measured altitude would not necessarily be accurate (if fetched at all given the lab's location in the basement of Photonics).

Measurements Taken

The measurements we took are as follows:

- Does the drone fly?
    - Results: yes, the drone works successfully (as seen in the video of the drone flying)
- Does the phone stay secured to the drone on the phone mount?
    - Results: yes, the phone does not fall off when the drone is flying (as seen in video of drone)
- Can the network speed be measured by the phone?
    - Results: yes, upload and download speeds could be collected. We measured the upload and download speed at 1219560 bits/sec and 3081880 bits/sec, respectively
- Can GPS coordinates and altitude be measured by the phone for every test run?
    - Results: yes, new GPS coordinates (latitude and longitude) and altitude was measured for every test run (GPS altitude is not entirely accurate inside of a building)

- Can the data be pushed to the database?
- Results: yes, data was pushed into our database (number 31)
  {‘id’,’download_speed’,’upload_speed’,’lat’,’lon’,’alt’}

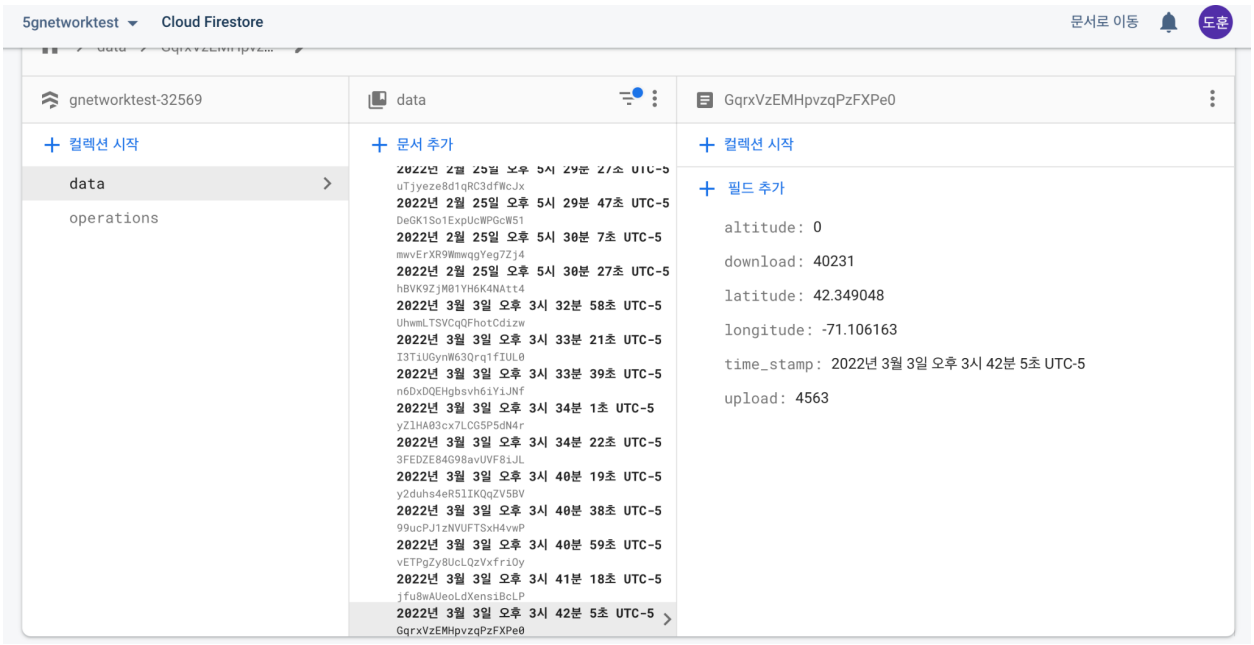

Fig. 1. Image of data in database.

- Can 5G network performance test operations can be remotely controlled on the web application?
  - Results: yes, user can start or stop tests on flask web application remotely

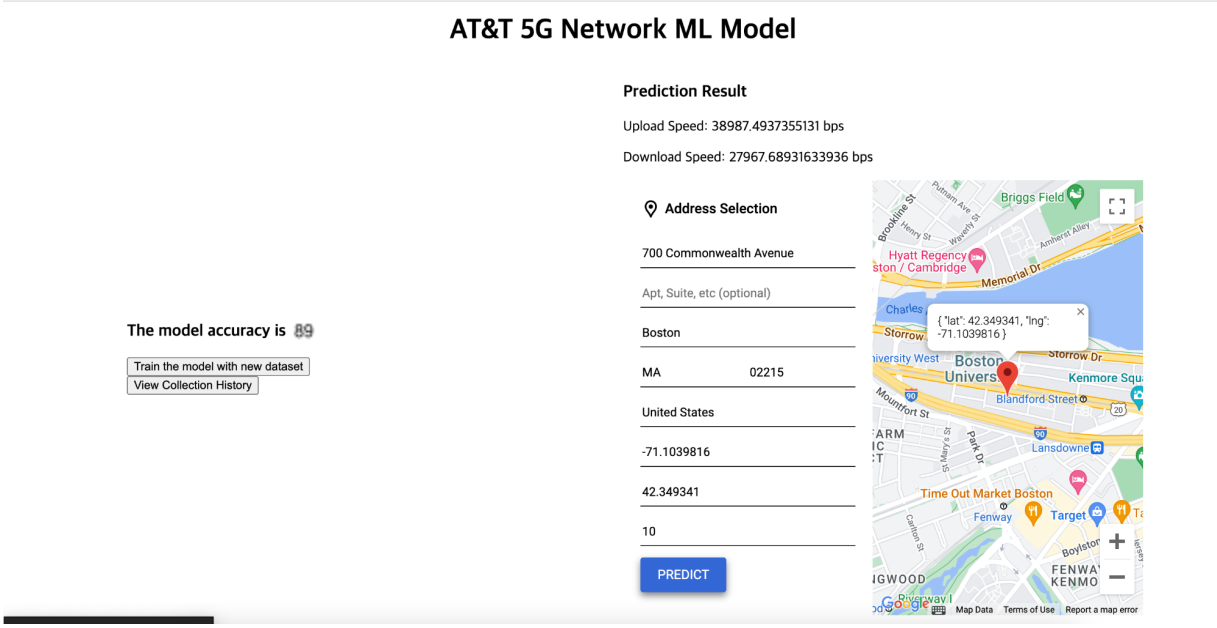

Fig. 2. Image of web application interface.

- Can the data be imported into the machine learning model?
    - Results: yes, we were able to pull data from Firebase to run the machine learning model locally on one of our team member's laptop
- Can the machine learning model calculate predictions based on the data?
    - Results: yes, we used a multi-out regression model from SciKit Learn on python, with a measured accuracy against test labels of 89% (disclaimer: this was achieved by clustering data by zip code using geopy API, and then separating predictions by altitude with 50 meters increments. Noisy data was normalized by being replaced by mean)

```
Sample predicdtion for latitude:  42.350872  lontitude:  -71.125286  altitude:  -8.944273
Actual Upload speed(Mbps):  42 Download speed(Mbps):  54
Predicted Upload speed:  60.19 Predicted Download speed:  68.09


Model Acurracy:  0.89
Upload Speed MSE:2.5218
Download Speed MSE:6.0499
```

Fig. 3. Image of machine learning model statistics.

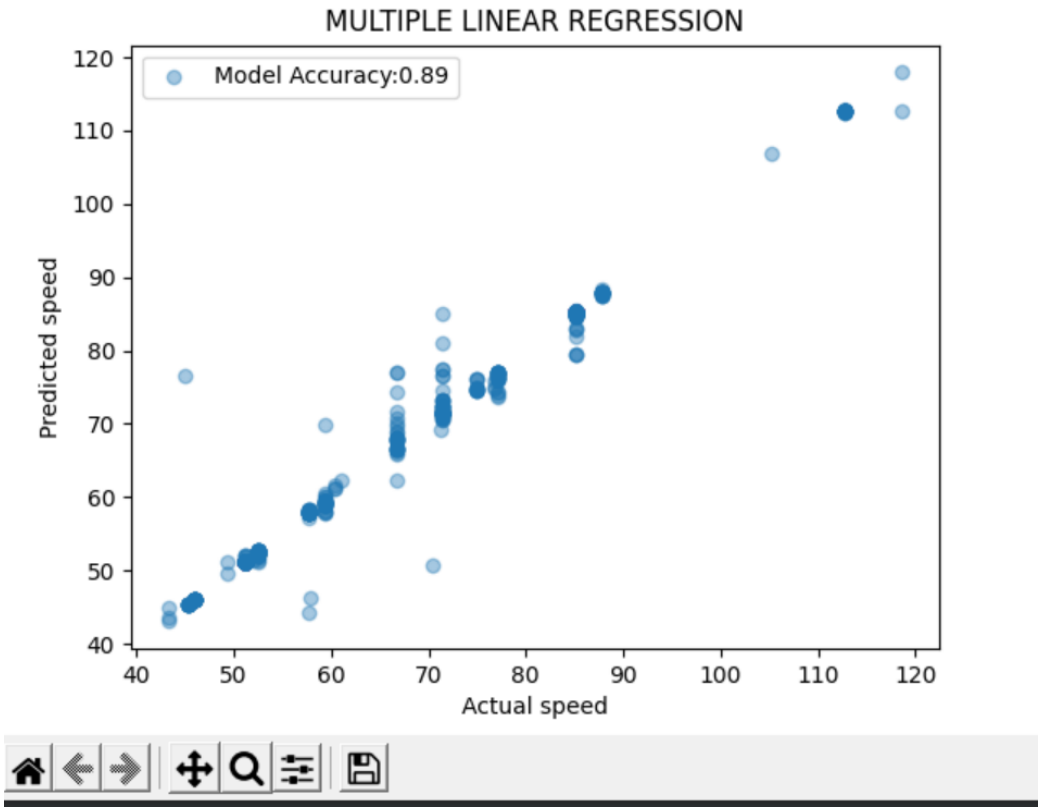

Fig. 4. Plot produced by model, predicted speed vs actual speed through linear regression.

Conclusions

Hardware (drone and 5G phone):

The hardware demonstrations were essentially simple qualitative yes or no tests. As shown in our video, the drone flies successfully. We have been collecting aerial data with the phone on the drone ever since before the second prototype test. Furthermore, we will likely collect even more data before the end of the project. Lastly, there have been no issues with the drone meeting our project requirements other than difficulties dealing with windy conditions when flying.

Also seen in the video and based on the fact that we have been collecting aerial data already, the phone fits well onto the drone. There also have been no issues with the phone connecting to 5G and running our app as necessary for this project.

Data collection app:

We were able to obtain results for upload speed, download speed, longitude, latitude, and altitude in our test. Since the network speed tests were taken in the same place, the collected data had a low variation over different network speed tests. However, as mentioned earlier, the altitude measured in the lab was not necessarily accurate since it was in a building. We have observed accurate altitude measurements in our field tests, using our new method of measuring altitude. The mobile application takes an initial altitude measurement on the ground and sets that equal to the ground altitude, and then on each subsequent data collection iteration (i.e. Ookla speedtest plus GPS query), the set ground altitude is subtracted from the measured GPS altitude to give a semi-accurate altitude in relation to the ground. Since our tests occur in locations with minimal altitude variance (i.e. flat fields or parks), this is an acceptable method of measuring how high the drone is. Every collected data point for each network speed test (latitude, longitude, upload speed, download speed, and altitude) can still be successfully pushed into the database and synchronously updated, as demonstrated in the second prototype test previously. As the mobile application is completely functional, we do not plan on touching it anymore.

Web application:

Our web app worked as expected. First, we were able to remotely control the starting and stopping of network performance tests on the Android application through the web app. Furthermore, the web app is integrated with the machine learning model so that the user can train the model whenever new data is added to the database and predict an upload speed and download speed based on the user input of GPS coordinates and altitude. We also implemented Google Maps so that a user can choose and find particular GPS coordinates via a displayed map, seen in Fig. 2.

Machine Learning:

Instead of sample data we used from the first prototype test, the machine learning model was trained with our own collected data. By exporting data from Firebase, we were able to access data with separated columns: upload, download, latitude, longitude, and altitude. Data preprocessing using dataframe takes place to make it a trainable set for our model.

The result predicted through the collected dataset has an error range of 100kbps, which is less than the standard deviation of the original dataset. We clustered GPS points to zip codes in order to reduce input parameters. Then, we separate altitudes into 50 meter increments and calculate means of speeds at corresponding altitude range. By taking Z-scaling normalization on the dataset, we achieved the model accuracy of 89% with 2.5Mbps and 6.0Mpbs of mean squared error for each. For this current model, we produced a graph showing linear regression between predicted and actual labels, which is seen in Fig. 4.