Wearable Context-Aware Sensing

for Human Behavior and Health Monitoring

Joseph Neathery (2587357) - Team Leader

Myra Zaidi (2635406)

Manpreet Chatha (2683926)

Faculty Advisor: Dr. Wenbing Zhao

Department of Electrical Engineering and Computer Science

Cleveland State University

Table of Contents

I. Abstract

Our goal was to develop an Android application to be used in conjunction with the Microsoft Band 2 that would utilize real-time data to give users a better picture of their health. The Microsoft Band 2 collects data via a variety of sensors. This sensors include an accelerometer, gyroscope, distance detector, heart rate monitor, pedometer, thermometer, UV monitor, calorie counter, galvanic skin response monitor, inter-heart rate monitor, ambient light monitor, barometer, and altimeter. All these sensors collected data based on the reading the Band 2 receives from the user's body.

We implemented machine learning and algorithms to determine what kind of activities a user performs, based on real-time data. We presented the user with a prediction of their activity, as well as a report of their overall health. We were able to successfully develop tests to ensure that our application was working properly so that users could get an accurate picture of their overall health.

II. Introduction and Background

The overall purpose and importance of this project was to promote healthy living using a graphically appealing, easy to use, cutting edge “smart” Android application in conjunction with the Microsoft Band 2. The idea behind this project was for a user to create an account with our application and have their health data collected and stored via a Microsoft Band 2. The user needed to wear the Band 2 consistently and for an extended amount of time in order for our application to collect enough data to create an accurate visualization of the user’s health. This was done by collecting the user’s sensor readings at set intervals, storing that data, and using machine learning and algorithms to analyze the data. A visualization of this data would then be displayed for the user on the application. As stated previously, the user has to allow for a sufficient amount of data to be collected before any analyzation and thus visualization could occur. Once the required amount of data for the selected time frame (one day, one week, or one month) was satisfied, analyzation could occur and a visualization/report could be displayed for the user to see. This visualization and report would be updated based on newly analyzed collected data from the user as they continue to wear the Band 2.

The Microsoft Band 2 comes with a wide array of sensors for readings to be taken from:

- Optical Heart Rate Monitor
- Three-Axis Accelerometer
- Three-Axis Gyroscope
- Galvanic Skin Response Sensors
- Skin Temperature Sensor
- Capacitive Sensor

In addition to these sensors, the Band 2 also reads the following: calories burnt, steps taken, and flights of stairs ascended/descended. We decided to focus on keeping track of that data as those are most useful and common for users of fitness devices. The application has three main functions that serve the user: activity classifier, graphs of stored health data, and health report of stored data. Details on how all this data is used will be discussed further on in this report.

III. Project Objectives

1. Develop an Android application to be used in conjunction with the Microsoft Band 2 that utilizes real-time data to give users a better picture of their health.
2. Implement machine learning and algorithms to determine what kind of activity a user is performing, based on real-time data.
3. Store collected user health data in a reliable and understandable database for analyzation at set time intervals.
4. Successfully develop tests to ensure that our application is working properly so users will be able to get an accurate picture of their overall health.

IV. Methodology

A. Design Report

Our project architecture was designed with three distinct layers comprising it. Those layers include: the client, SQLite, and an Amazon EC2 WAMP server. The client layer is the Java application that runs on the physical Android device that the user interacts with. This layer is the one that interacts and gets readings from the Microsoft Band 2. The next layer is the SQLite layer, which is a SQL-style database that resides on the Android device. This was implemented so that data could be stored locally to allow the user to use the application while not connected to the internet. The third layer was an Amazon EC2 instance running a WAMP server on it. An Amazon EC2 instance can be viewed like “cloud storage” that the application utilizes to store data off the local device. The Amazon EC2 runs a WAMP server which stands for Windows, Apache, MySQL, and PHP. The EC2 instance runs the Windows Server 2012 operating system and utilizes Apache and PHP to interact with the MySQL database. The MySQL and SQLite databases have the same overall database architecture. Each Microsoft Band 2 sensor has its own table with the table fields:

- Unique User ID
- Band 2 Sensor Reading Values
- Timestamp of Each Reading

For example, the heart rate table will contain the following fields:

- UserId
- HR (heart rate reading taken from the Band 2, stored as float value)

- Quality (string value taken from the Band 2 that indicates the quality of the reading taken)
- Timestamp

The one major difference between the MySQL database architecture and the SQLite database architecture is that the MySQL database contains an extra table on it that stores a user's login information. The user is required to be connected to the internet on initial login, so that the user's credentials can be verified with that of what is stored on the database from when the user registered. The SQLite database does not contain this private information because it would require that every device running the application have every user's login information stored on the local SQLite database instance. Utilizing a singular database table that stores this information on the "cloud" allows for better security and a simpler application architecture.

After discussing the overall architecture of the project and application, we can now discuss the data collection workflow. That workflow goes as follows:

- User registers an account on the application
    - Inputted data is stored on the "cloud" database.
    - This data is pulled and checked for validation when a user attempts login. On success, the user's unique ID is returned the client side of the application.
- User logs in to the application.
    - User can now use the application to collect data either online or offline, with no data loss.
- User starts data collection service, which begins the sensor readings collection.
    - Every second the Band 2 passes on readings from each of its sensors.

    - These readings are stored in the corresponding tables on the SQLite database.
    - Every five minutes, the application checks for an internet connection. If one is found, the data stored in the SQLite database is pushed to the MySQL database and the SQLite database is emptied to save storage space on the device.
    - If not internet connection is found, the process is repeated in another five minutes.
- This process carries on until the user stops data collection or logs out.

The application is also equipped with a "Graph" and a "Report" functionality. A Graph and Report are generated based on Band 2 readings that have been stored on the MySQL database. They can be generated using data from the past one day, one week, or one month. The user must be connected to the internet for these functionalities to work. When the user selects the time frame they want, that target date and the user id are sent to the EC2 instance for querying. The queried data is then returned to the application for manipulation and display. A generated report will display the number of calories burnt, average heart rate, and total number of steps taken for the selected time frame. A generated graph will display a user's heart rate versus the number of steps taken in a graphically appealing line graph over the course of the selected time frame. We were able to create this graph using a third-party API called GraphView. Please see appendices A and B for a visual representation of the application architecture and user interface.

Our application is also used to alert a user if there is a heart rate irregularity. According to the American Heart Association, a person can calculate their "estimated maximum heart rate" by subtracting their age from 220 [1]. Our application does this for the user when they log in. When the data collection service is started, the user's heart rate reading is compared to their estimated maximum heart rate. If the user's heart rate reading is greater than that of the estimated maximum heart rate, the user is alerted to seek medical help. This all occurs in real time, as this heart rate check occurs every second while the data collection service is running.

One of the main project goals that was set before us was to make our application "context-aware." This meant that our application should be "smart" enough to detect what the user was doing. To accomplish this task, we implemented a custom TensorFlow model to predict the user's activity based on the Band 2's Accelerometer and Gyroscope readings [6]. TensorFlow is Google's open-source machine learning API that is used in a variety of applications. Machine learning is usually applied to solving two kinds of problems: regression and classification. Since we want to identify or "classify" what kind of activity is being performed by the user, we needed to use a classification type of model. Let us now establish some background on the details of our TensorFlow model.

To understand some of the technicalities of our model, we first need to define what a neural network and a recurrent neural network are. A neural network, which is also known as deep learning, is a special type of machine learning where a computer solves a problem by analyzing a set of training examples [4]. A recurrent neural network works like a chain of standard neural networks, each having a separate input and output, while passing along information to the next chain in the network. This is useful when patterns need to be recognized and remembered in a large dataset. Our model utilizes a special kind of recurrent neural network called a Long Short Term Memory network. This has a similar structure to a standard recurrent neural network, but instead of having a single neural network layer, it has four that all interact with each other in a special way (see appendix C for a visualization). LSTMs combat the problem of long term dependency, which is when learned behaviour from farther back in the chain is needed in the current node [5].

TensorFlow models usually need to be trained using a large amount of data and time to produce the most accurate results. In order to get the most accurate model, we utilized a large dataset of over 150,00 rows from the UC Irvine Machine Learning Data Repository [2,3]. Each row of data contains the following fields:

- X-Acceleration
- Y-Acceleration
- Z-Acceleration
- X-Angular Acceleration
- Y-Angular Acceleration
- Z-Angular Acceleration
- Labeled Activity

80% of these rows were used for training the model, while the remaining 20% was used for testing and verifying the model's accuracy. The model was trained over 50 epochs, with one epoch being equal to one complete pass through the entire dataset.

After training the model and ensuring accuracy, we implemented this model in our application. The model takes the following fields as input:

- Band 2 X-Acceleration
- Band 2 Y-Acceleration
- Band 2 Z-Acceleration
- Band 2 X-Angular Acceleration
- Band 2 Y-Angular Acceleration
- Band 2 Z-Angular Acceleration

To ensure the highest accuracy, the application collects five sets of readings and passes them to the model. This means that activity classification only occurs once every five seconds. Upon receiving the input dataset, the model returns a "certainty" percentage for each of the following five potential activities:

- Standing Still
- Lying Down
- Walking
- Climbing Stairs
- Jogging

The activity with the highest "certainty" percentage is then displayed to the user as their predicted activity based on their real time Band 2 readings.

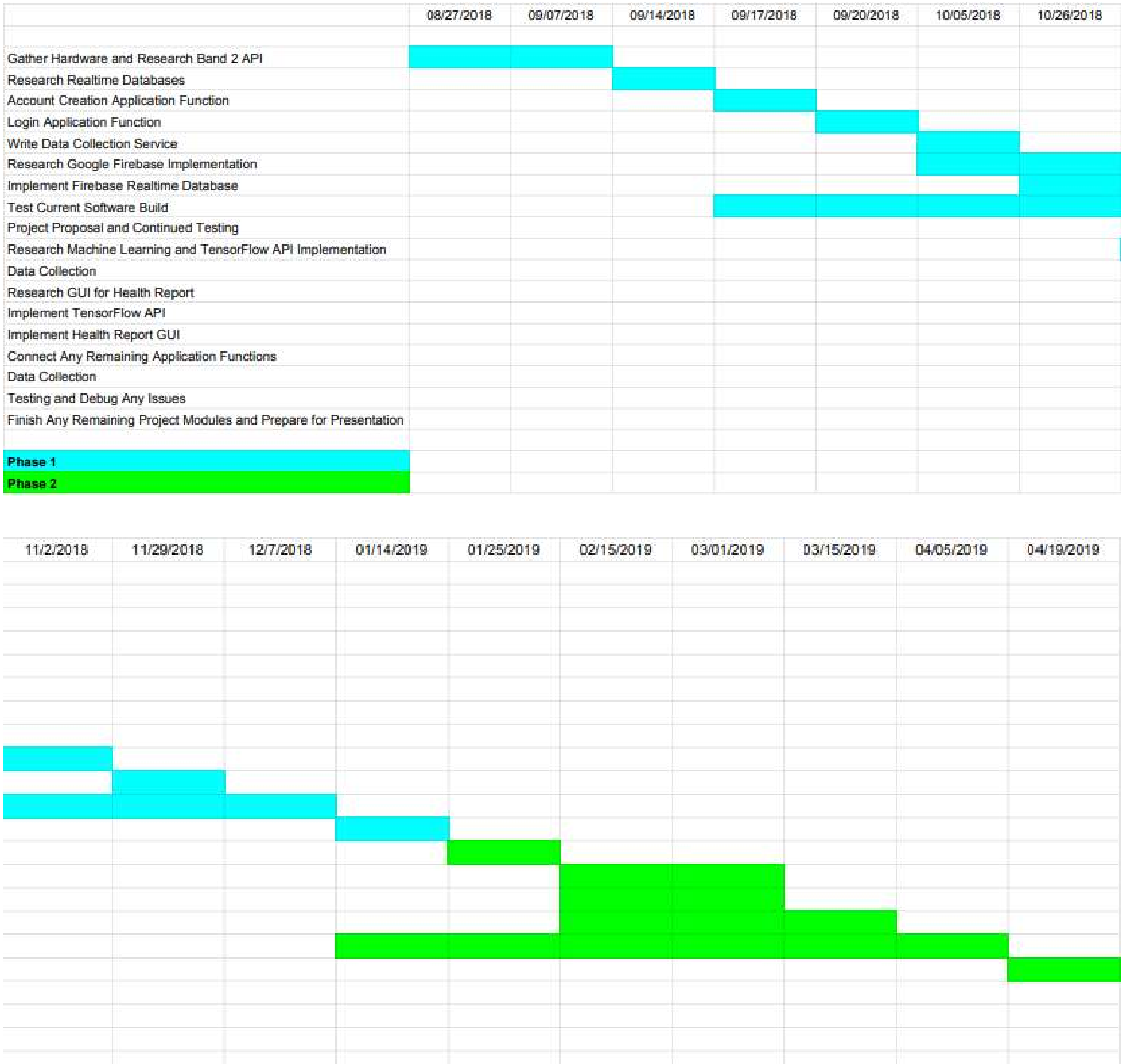
B. Synthesis of Project Results and Discussions

After much research and hard work, our team was able to deliver successful project results. We were able to successfully develop, train, test, and deploy a TensorFlow model with approximately 94% accuracy. This model was later implemented in our application in order to create a functioning activity classifier. This activity classifier has overwhelming success in classifying the five different activities based on synthetic test data (see appendices D and E for a visual representation of the accuracy of the model). We were also able to create successful and insightful "Graph" and "Report" functions that give the user a better visualization of their health. Overall, we were able to develop a functioning, cohesive, and useful Android application that can be utilized for a variety of purposes and expanded upon in the future.

C. Project Deliverables

- A TensorFlow model that predicts user activity based on Band 2 readings with roughly 94% accuracy.
- Activity Prediction using this model in our Android application to make it "context-aware."
- A functioning, intuitive, graphically appealing, and "context-aware" Android application with "Graph" and "Report" functionalities that give users a better visualization of their health.

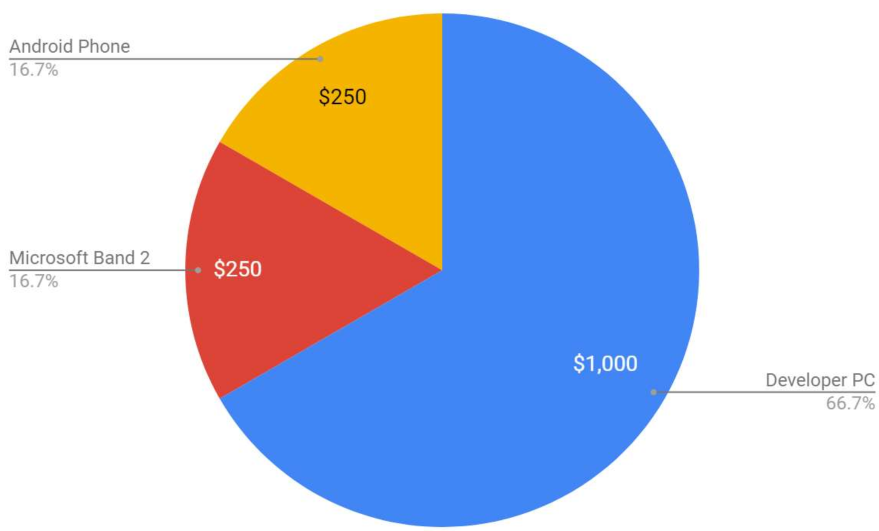
D. Project Timetable



E. Budget and Budget Description

We were provided with two pieces of hardware, which were a Microsoft Band 2 and an Android phone. Team members did not have to purchase these pieces of hardware, as they were provided by Dr. Zhao.

Item	Estimated Cost
Microsoft Band 2	$250
Android Phone	$250
Developer PC	$1000
Amazon EC2	Free (In the Free Tier)



F. Problems and Challenges

We recommend using different hardware instead of the Microsoft Band 2, as it sometimes provided inaccurate readings. The sensors made it difficult to collect correct data, which is why we believe Microsoft made the decision to discontinue these bands. For future projects involving this hardware, we recommend building the sensors to ensure more accurate readings. This application could also be worked on further to include more features such as different reporting methods, more data analysis, and adding even more activities to classify. Reaching out to one of the local health institutions to sponsor further development of this project is also recommended. We certainly believe that this project improved our ability to work with a team of people for an extended period of time. We also learned Machine Learning using TensorFlow and how to graph real-time data for an application using Android Studio.

V. Conclusion

Through research, we learned how to apply machine learning to a useful application that can allow users to have a better grasp on their everyday health. This machine learning was made useful in the form of an activity classifier in order to make our application "context-aware." We were also able to implement a "Graph" and "Report" function to give users a better visualization of their health. Some possible applications of our work include everyday use, monitoring of a patient, and research to see how different health readings relate to each other. While this application is useful, there definitely is room to build upon it in the future. Despite the ability to build upon the application further, we met the project objectives that were set forth from the beginning.

VI. References

[1] American Heart Association. (2015, January 4). Know Your Target Heart Rates for Exercise, Losing Weight and Health. Retrieved April 23, 2019, from https://www.heart.org/en/healthy-living/fitness/fitness-basics/target-heart-rates

[2] Banos, O., Garcia, R., Holgado, J. A., Damas, M., Pomares, H., Rojas, I., Saez, A., Villalonga, C. mHealthDroid: a novel framework for agile development of mobile health applications. Proceedings of the 6th International Work-conference on Ambient Assisted Living an Active Ageing (IWAAL 2014), Belfast, Northern Ireland, December 2-5, (2014). https://archive.ics.uci.edu/ml/datasets/MHEALTH+Dataset#

[3] Banos, O., Villalonga, C., Garcia, R., Saez, A., Damas, M., Holgado, J. A., Lee, S., Pomares, H., Rojas, I. Design, implementation and validation of a novel open framework for agile development of mobile health applications. BioMedical Engineering OnLine, vol. 14, no. S2:S6, pp. 1-20 (2015). https://archive.ics.uci.edu/ml/datasets/MHEALTH+Dataset#

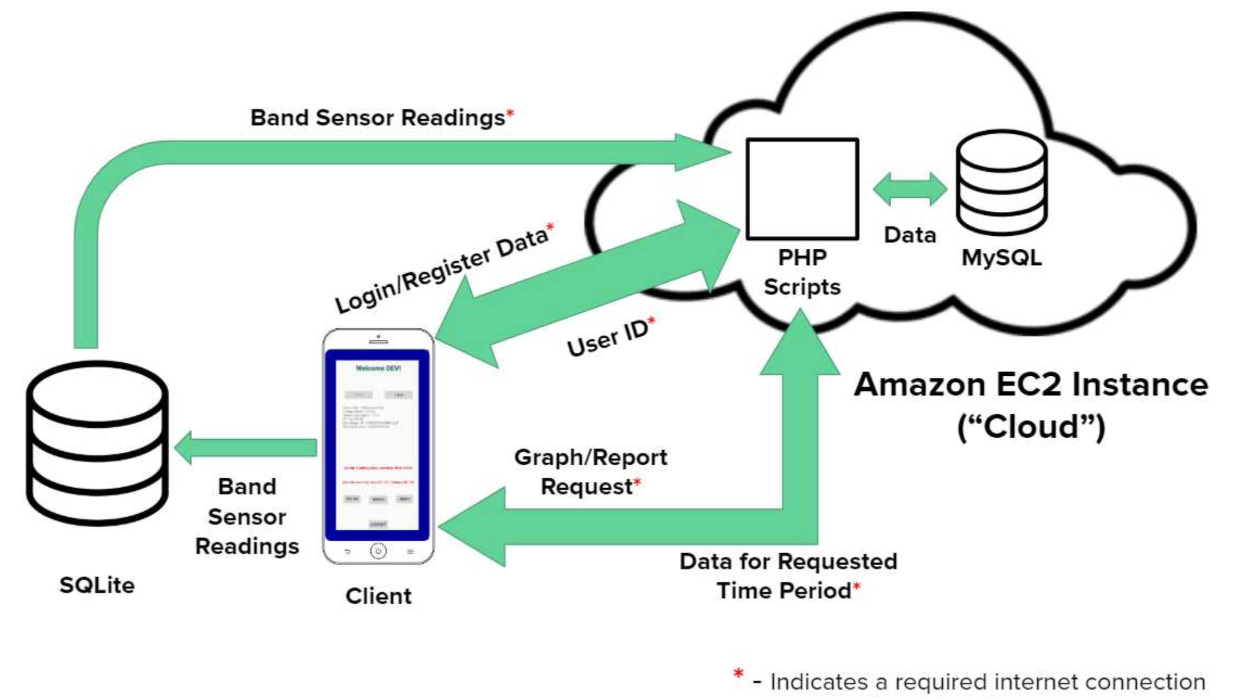
[4] Hardesty, L. (2017, April 14). Explained: Neural networks. Retrieved April 23, 2019, from http://news.mit.edu/2017/explained-neural-networks-deep-learning-0414

[5] Olah, C. (2015, August 27). Understanding LSTM Networks. Retrieved April 23, 2019, from https://colah.github.io/posts/2015-08-Understanding-LSTMs/

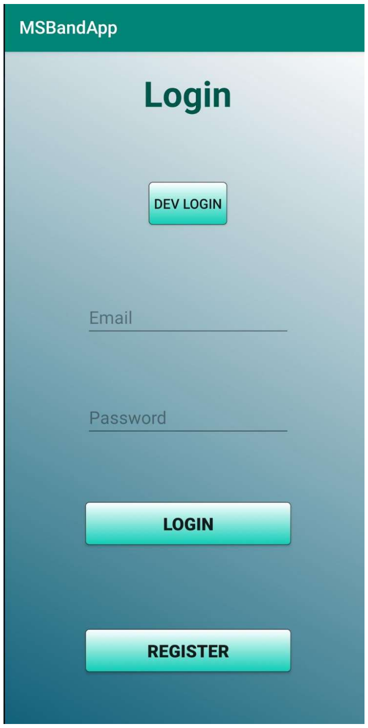
[6] Valkov, V. (2017, June 03). Human Activity Recognition using LSTMs on Android - TensorFlow for Hackers (Part VI). Retrieved April 23, 2019, from https://medium.com/@curiousily/human-activity-recognition-using-lstms-on-android-tensorflow-for-hackers-part-vi-492da5adef64

VII. Appendices

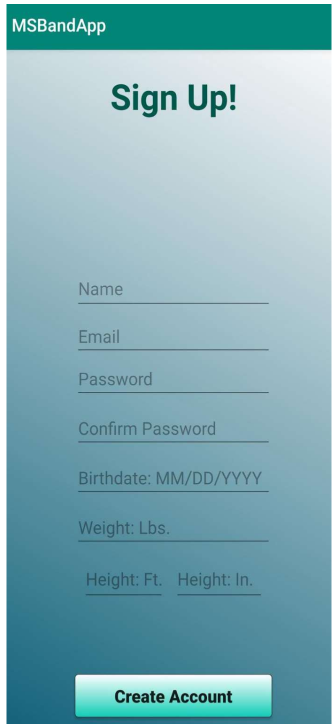
A. Application Architecture



B. User Interface



Login Screen



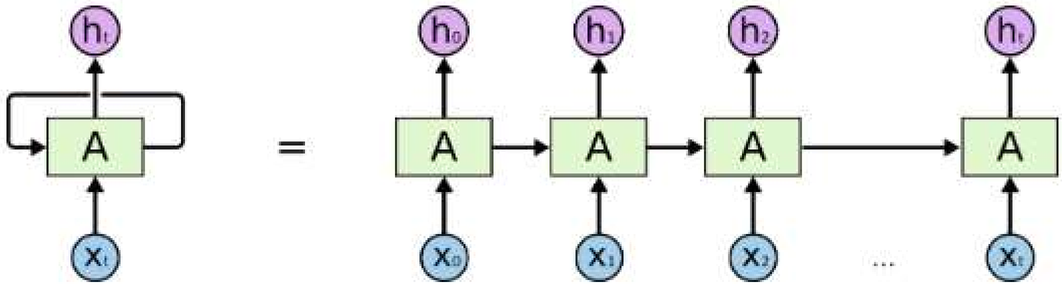
Register Screen

Main Application Interface

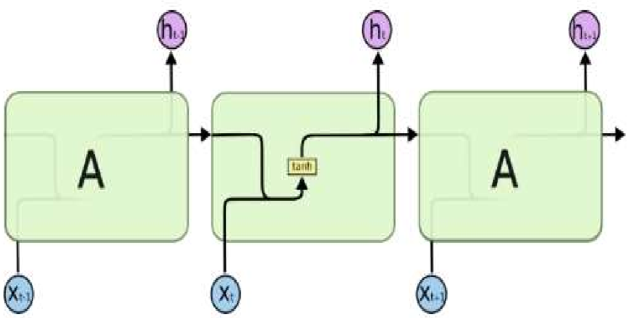
Report Screen

Graph Screen

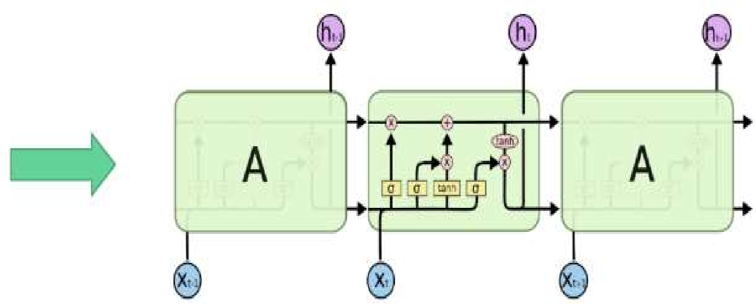
C. RNN and LSTM Visualization



An unrolled recurrent neural network.

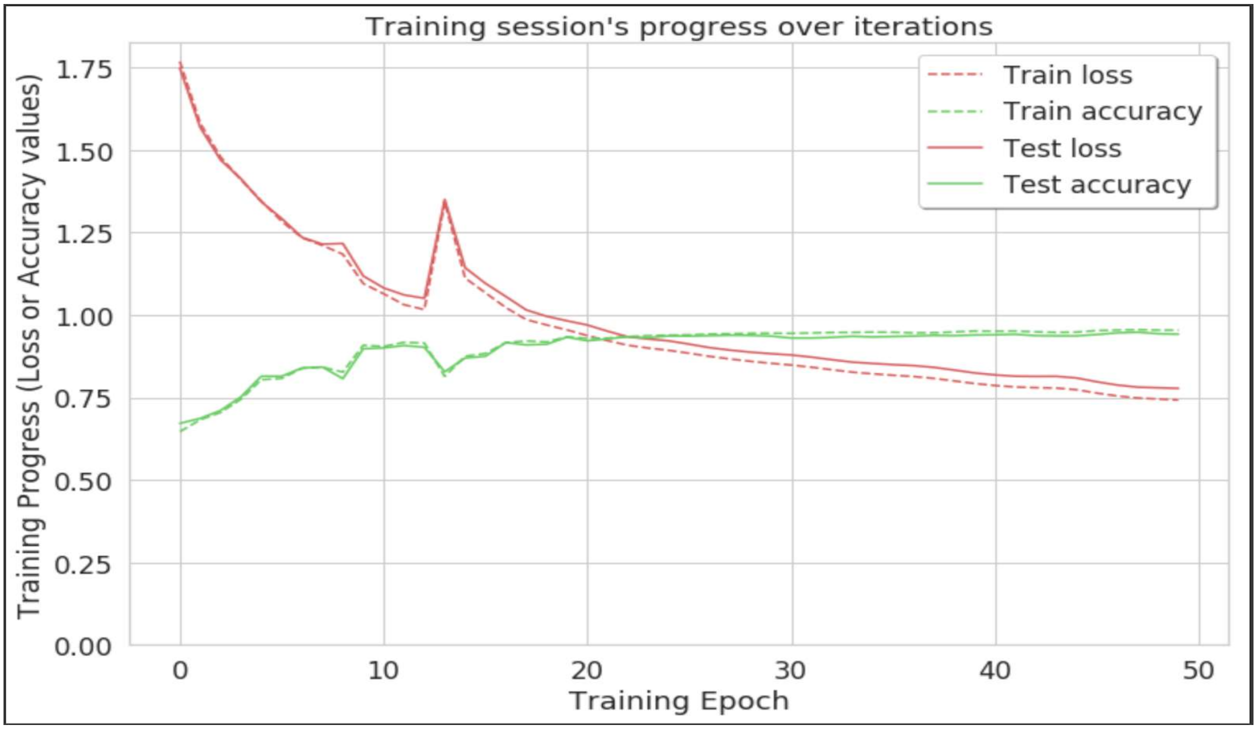


The repeating module in a standard RNN contains a single layer.



The repeating module in an LSTM contains four interacting layers.

D. TensorFlow Model Training Accuracy



E. TensorFlow Model Confusion Matrix