Understanding network dynamics of a bike-sharing system to enhance inventory self-balancing

CS5834 Intro to Urban Computing - Project

Alexander Rodriguez
Department of Computer Science
Virginia Tech

Anika Tabassum
Department of Computer Science
Virginia Tech

Bumsik Kim
Department of Civil and Environmental Engineering
Virginia Tech

ABSTRACT
How does the network dynamics from a bike-share system provide insights to enhance its inventory self-balance? In this work, we model station demand and route flow across time and weather considerations. These models are used to make long term and real-time decisions about incentives for riders. Such incentives aim to improve inventory self-balancing, which increments service satisfaction and company’s profit. Therefore, we propose using a visualization to support decision making related to these incentives across different times of the day.

KEYWORDS
Bike-sharing system, urban computing, dynamics visualization

1 INTRODUCTION
Since the bike-sharing system launched in the U.S., the system has grown dramatically. According to the National Association of City Transportation Officials (NACTO), in 2010, the total number of bike-sharing trip was 320,000, however, in 2017, the number of bike-sharing trip was 35 millions.

Figure 1: Annual growth of Bike share ridership in U.S.

Especially, the number of bikes in 2016 was 42,500 while the number of bikes in 2017 was 100,000 due to the introducing ‘dockless’ bike sharing system. The dockless system does not require a conventional parking spot, instead, they used smartphone applications to inform the location of available bikes. In other words, the users in the dockless system can park a bike wherever they want. However, still the system is in an initial stage and most of the cities have adopted traditional bike-sharing systems with bike stations. The dockless bike-sharing system also has pros and cons, however, this study will focus on the traditional station-based bike-sharing system.

Every bike-sharing stations have different demand by time of the day. Sometimes, some of the stations are full of the bike so the users cannot park a bike. On the other hand, during the peak hour, some of the stations have no available bike. To keep it balanced, the operator uses force-balancing which means bike moving truck.

The balancing is important for efficient usage of the asset of a station. In the short term, the imbalanced bike inventory takes chances of using a bike to potential users. At the very worst, In the long term, the users give up their expectation to the bike sharing system and no one wants to use the bike-sharing system. In this study, we defined self-balancing as a trip by user and force-balancing as a trip by operators truck. And we assume self-balancing is the ideal case because self-balancing requires no additional cost.
One policy alternative to improve self-balancing is by incentivizing routing from low demand stations to high demand stations. For instance, in Figure 2, we can note that trips from D to A could be incentivized by splitting in incentives from D to B and B to A. This routing technique would be helpful when there are not many people going from D to A.

![Figure 2: An example of self-balancing incentive](image)

The self-balancing and incentive could reduce the management of system operator as well as enhance the reliability of bike sharing system itself. If the bike-sharing system runs by self-balancing, the operator does not need to dispatch force-balance vehicle and the operator could save cost. Next, the users start to think the bike-sharing system is reliable if they can borrow a bike whenever, wherever they want. In long term view, the more self-balancing could change people’s behavior and could increase bike mode share.

### 2 PROBLEM DEFINITION

#### 2.1 Problem statement

As noted before, imbalanced bike inventory disturbs a potential user from using the system. Currently, bike-sharing systems require force-balanced interventions which are costly and do not fully satisfy demand. The trip data from bike-sharing system shows potential to bring insights to take action, as for example, give incentives to users of specific routes, so that they transport bikes to where the users need them to be.

#### 2.2 Expected benefits

First of all, we can expect that self-balancing could reduce the operational cost of bike-sharing system. From the literature, the bike distribution cost is one of the major operational cost. However, more self-balance might reduces the moving truck and therefore, the operator could save the cost. Next, the users can trust the bike-sharing system if they can hire a bike whenever, wherever they want. In long term view, the more self-balancing increases bike share and reduces congestion.

### 3 RELATED WORK

#### 3.1 Balancing Bike-sharing Systems

Many researchers have been tried to solve balance problem. [8] pointed out that the major operational cost is rebalancing cost. To reduce the cost, they tried to use two methods: determining service level requirements at each bike sharing station, and designing near optimal vehicle routes to rebalance the inventory. In this study, they utilized Markov-chain and clustering to suggest proper routes. [5] introduces the dynamic public bike-sharing balancing problem to attenuate that there is more demand in certain stations than others. This problem routes bikes to balance the network. Its main contribution is a mathematical formulation for the optimization. While we share a similar goal with this work, we opt not to do a optimization and instead we want to use a visualization. One last study, [10], suggests incentives to the user who have willing to repositioning bike by using crowd-sourcing mechanism and surveys. The paper revealed that the self-balancing works with dynamic incentive were accepted 60 percents of participants.

#### 3.2 Machine Learning in Bike-sharing Systems

In literature, there have been some applications of machine learning in bike-sharing systems. For example, [7] worked to estimate the traffic in a bike-sharing system by including demand prediction and trip duration with regression models that consider meteorological conditions and important events in the city. In our work, we partially adopt this idea and include meteorological conditions as we think they should be highly predictive for demand forecasting. [12] worked in a similar problem to ours in a bike-sharing system where they wanted to predict the trip destination and duration. For the trip destination inference they used MART (Multiple Additive Regression Trees) and the estimation of the trip duration they used Lasso. To build these models they used features from the user (gender, age), the departure time, and the bike stations location. In contrast to them, we don’t have personal information about users. In addition, this paper suggests using Manhattan distance in estimating travel distances in a city.

#### 3.3 Visualization of Urban Dynamics

[6] created a visualization tool for taxi trips in a city called TaxiViz. In their case study, they examine how trip frequency vary for each day of the week for each neighborhood. Their work recognizes the importance of considering transportation hubs to understand the flow of taxis around the city, as for example, airports and major train stations. In addition, it exploits relationships among origin-destination and displays
statistics about this. This work is a good example of what we could continue for future work.

3.4 Time-series Forecasting

Different algorithm have been proposed for time series and demand forecasting in past years. Zhang et al. [11] proposed a hybrid approach for forecasting using Auto Regressive Moving Average (ARIMA) and Artificial Neural Network (ANN). Shumway et al. proposed EM algorithm of machine learning for time-series forecasting and [9] smoothing time-series.


LSTM is a variant of Recurrent neural network which is very popular now-a-days for classifying and forecasting time-series data [2]. This model is designed to solve exploding and vanishing gradient problem of deep learning. LSTM network is handled to design prediction for sequenced data and long term dependencies . An LSTM unit is composed of a cell, input gate, output gate and forget gate. Figure 3.4 shows a how a lstm unit is designed. Each input gate decides to take new input or from previous hidden layer, forget gate decides whether to delete the output or save it for later and output gate decides which to output for prediction.

4 DATA DESCRIPTION

4.1 Description of target area

Currently, there are seven cities in the U.S. which open their bike-sharing trip data to the public. Among the seven cities, we selected greater Boston area as our target area. Because, the bike-sharing system in Boston (Bluebikes) is one of the biggest bike-sharing systems and the bike trip data is detailed than the other cities. Additionally, the Bluebikes have been released their real-time bike inventory data in General Bikeshare Feed Specification (GBFS) Format. Therefore, the authors decided the target area as greater Boston area because of the extensibility.

4.2 Bike sharing data

Since 2015, the bike trip data of Bluebikes has been collected and published every month. The bike trip data contains trip duration in seconds, trip start time and end time, trip start station and end station, bike ID, membership type, age, and gender. Every station data (start station and end station) has its coordinates and name. Figure 4 shows the sample bike trip data.

Each row represents each trip so the number of row represents the number of total trips. This research uses one year data from November 2017 to October 2018 to reflect seasonal effect.

4.3 Descriptive statistics of the data

Here is the general description of the bike trip data. The number of the total trips during the analysis period (one year) is 1,724,961 trips. During the period, the number of bike stations has been changed slightly due to the new construction and demolition. In this study, we used August 2018 data as our reference data so as of August 2018, the number of the station is 228. The average trip duration is revealed as 26.11 minutes. One notable thing is that there are a lot of subscribing users than the single trip users. Around 80 percent of trips were occurred by subscribers and only 20 percent of trips were one-time user. Based on the information, we assumed that the Bluebikes is mostly used by locals for commuting purposes.

The trip distribution plot supports this assumption. In Figure 5, there are two obvious peak patterns exist during the weekday. However, on Weekend, the two peaks were not existing. It means that the trip is mostly used for commuting because most of the trips on a weekday were focused on AM peak and PM peak. During this peak period, we can expect that most of the bikes will move from the residential area or near the transit station to the CBD area. And in these two peak time period, some of the stations will have no bike whereas some of the stations have no empty spot.

4.4 Area of interest

As we mentioned previously, we assumed 228 bike stations as our total. Most of the stations are located in the Boston and Cambridge area (see Figure 6), however, some of the stations are located in rural area. The problem is, the stations in rural area has very few trips a day and it reduces the model prediction power.
Figure 4: Sample bike trip data of Bluebikes

Figure 5: Average demand of bikes of stations with time based on Bluebikes trip data

Figure 6: The Bluebikes station distribution

Therefore, we selected high demand stations that requires self-balancing. First, we draw maps that shows top 30 origins and top 30 destinations. Figure 7 and Figure 8 shows the location of top 30 stations. According to the figures, the all top 30 stations are in the downtown of Boston and Cambridge area. Therefore, we decided to use these area’s bike trip data for the study.

Figure 7: The top 30 origin stations

Figure 8: The top 30 destination stations

Here is the rule of area selection. First, the target area should cover all top 30 origins and destinations. Next, the
area is selected by census level because the same census level shows similar characteristics. Based on these criteria, we picked census tract that starts their ID as 010, 020, 030, 070, 352, and 353. Figure 9 shows the selected station. 85 stations are in these area of interest, and we used trip data between these 85 stations for the analysis.

Figure 9: The selected 85 stations

4.5 Other data
To enhance the model prediction power, we used additional data as our input variables. The authors believe that the most important determinant of biking is weather condition such as rain and temperature. So we collected one-year weather data, includes average temperature, precipitation, snowfall, and wind speed, from the National Oceanic and Atmospheric Administration (NOAA) and used it as our input variables. Also, we used weekday, weekend, and holiday variables to reflect daily variation. Lastly, we utilized real bike inventory data to see how many bikes exist by each station.

5 METHODOLOGY
The methodology to understand the dynamics of the system comprises three parts: detecting force-balanced trips, demand prediction, and destination prediction.

5.1 Detecting force-balanced trip
To detect if a trip or bike needs to be force-balanced is to necessary to model the decision making of the system. In the following Figure 3 row 3, we observe that a user starts a trip with a bikeID W00005 from one station (31087) and ends at a different station (51047). In row 4, the bikeld W00005 started trip from station 31047 to 31083 and it seems reasonable because it explains the bike W00005 stayed at station 31047 from the end of previous trip. However, in row 5, the bikeld W00005 started a trip from station 31048 that is supposed to be the end station (31083) from the previous trip. In this case, the bike W00005 was force-balanced from station 31083 to station 31048 between 17:56 and 20:29.

In this study, we have to know whether this bike needs to be force-balanced to other station or not and if it needs to be force-balanced then at which hour. For this, we need to know the demand of the station at a particular time from historical data as well as the number of force-balanced trip occurs at a particular hour of that station.

5.2 Demand Forecasting
Demand forecasting or to forecast the demand of bikes of a particular station of a particular time period of a day can be ease the force-balanced decision. We adopted a multivariate time-series forecasting approach to forecast demand of bikes of a station. We use LSTM sequential model for each station to forecast demand. We discuss the approach in following:

5.2.1 Data Preprocessing: From Figure 5 we divide the 24-hour time into four time periods: off peak (8pm-5am), AM peak (5am-10am), moderate (10am-3pm) and PM peak (3pm-8pm). For each day, We calculate demand of each station at these four time periods. We assume that, along with time demand of a station also varies with weather and day of a week, i.e., demand of a station is different for sunny, rainy, weekdays and holidays. For this, for each trip data, we combine the weather variables and holiday variable of that day. Next, among all the stations of one year trip data we select 85 high demanding stations as discussed in Sec 4.4. After data preprocessing for each station a row of the dataset has the features shown in Table 1. The size of the dataset after preprocessing is 0.15M for 85 high demanding stations. Each station has on average 1500 index or rows with the following features as shown in Table 1.

5.2.2 Machine learning model: We use LSTM sequential model for demand forecasting. Our approach is similar to the multivariate timeseries forecasting1 (MTSF). With given input features at time $t$ MTSF forecast features at time $t + k$. In our case with the input features, i.e., weather variables, holiday or not, time period $t$ and given demand at time period $t$ our approach forecasts demand for time $t + 1$. Since, we are using multiple features for forecasting demand along with the present time series value, we processed our input data to work for supervised learning prediction problem. For training, the LSTM model is based on this- we added one column as output which contains demand at $t + 1$, i.e., we consider demand of the same station of the next row in the dataset as our output. For 85 high demanding stations we

---

trained 85 LSTM model to have better output. We split the dataset for each station into 90% training and 10% test data. We provide a LSTM model summary in Table 2.

<table>
<thead>
<tr>
<th>Optimizer</th>
<th>Adam</th>
</tr>
</thead>
<tbody>
<tr>
<td>Epoch</td>
<td>50</td>
</tr>
<tr>
<td>Batch size</td>
<td>5</td>
</tr>
</tbody>
</table>

Table 2: Parameters used in LSTM model

5.2.3 Python packages: We used Python Pandas and Numpy packages for preprocessing the data. For LSTM model we used Keras. Figure 10 shows the code snippet where we design the LSTM network for demand forecasting and Figure 11 contains the part of the code snippet where we convert time series data into supervised learning prediction problem.

5.3 Destination Prediction

We plan to model a destination prediction of a trip originated at a source station. Our goal is to identify number of bikes a station might need to force balance. For example, A station can predict its demand of bikes at a particular hour using Section 5.2 and it already knows its current inventory of bikes. From this, it can easily calculate its shortage or surplus of bikes at a particular hour. Next, if it can predict what are the possibilities of bikes coming in this station within a few hours, it can decide if it requires any force-balanced bikes as well as optimize the number of force-balanced trips a station requires. In the following, we describe how we build a model to predict destination of a trip.

5.3.1 Data preprocessing. In data preprocessing step, for destination prediction, for each station we used two input features- origin station Id and time period of the day. We used the destination station Id as the prediction output of the model. We also calculated the average trip duration of same pair of origin-destination of stations and store that to understand the time of bikes arriving at a particular station. For every trip, we choose the pair of origin-destination Id of stations and convert actual starting trip time into one of four time periods as we discussed in Figure 5. We filter the data by considering only 85 high demanding stations we obtained in Section 4.4 as destination. The size of the dataset after filtering is 0.1M trips.

5.3.2 Machine learning model. For predicting destination from the dataset we use a decision tree classifier. We assume decision tree should work best as we only have two features station Id and time period of day. Figure 12 shows the goal of using decision tree classifier. Each level is considered as one of the features. First level uses the origin station, Each origin station is divided into four time period into second level and third level contains 85 selected stations as destination containing probability value of arriving at this destination for a particular origin and time period.

After obtaining the probability values of each destination station for a particular origin and time period, we use a random generator model to predict the destination station. The random generator model uses the probability weight values to predict destination Id of the station. Figure 13 shows a sample of probability weight values for a particular origin station and time period after modeling the decision tree classifier. We split the data set into 90% training and 10% test data to evaluate the prediction model. Our goal is to predict the number of bikes coming to a particular station.
Understanding network dynamics of a bike-sharing system to enhance inventory self-balancing Conference'17, July 2017, Washington, DC, USA

Figure 12: Decision Tree classifier

at a particular time period, not which bike is coming. We discuss the evaluation of our prediction model in Section 6

Figure 13: Probability values of destination stations for a particular origin and time period.

5.3.3 Computing number of bikes arriving. We predict the destination of each trip using the destination prediction model and random generator. Next for each origin-destination pair we found from the trip data, we calculate the number of bikes coming to each station using the predict result.

5.3.4 Python packages: Pandas for preprocessing data, Scipy for modeling destination prediction. For identifying probability values of each station we use predict_proba of decision tree classifier. Figure 14 and Figure 15 shows the code snippet in Python how we generated the decision tree classifier and random generator to predict destination Id.

5.3.5 Bias on model: We consider only a station Id and time period of a day to model the destination prediction ignoring several other factors like shortest path, user preference etc as we did not have these factors in data.

Figure 14: Code snippet of building decision tree classifier.

```python
def model_dec(X_train, Y_train, X_test, Y_test):
    X_train = np.asarray(X_train,dtype=float)
    X_test = np.asarray(X_test,dtype=float)
    Y_train = np.asarray(Y_train,dtype=float)
    Y_test = np.asarray(Y_test,dtype=float)
    print('Decision Tree!')
    from sklearn import tree
    clf = tree.DecisionTreeClassifier(max_depth=5, min_samples_split=1)
    clf.fit(X_train, Y_train) # np.unique(Y_train)
    y_pred_test = clf.predict_proba(X_test)
    y_pred_train = clf.predict_proba(X_train)
    print('predict proba')
    print(y_pred_test.shape)
    print(y_pred_train.shape)
```

Figure 15: Code snippet of using random generator to predict destination.

```python
def output_prediction(y_pred_train, y_pred_test, X, sta_list):
    predY = np.concatenate((y_pred_train, y_pred_test), axis=0)
    dataX = predY.shape[0]
    index = np.arange(predY.shape[1])
    print(predY.shape)
    print(index.shape)
    random_index() for obs in range(dataX):
        prob_weight = predY[obs, :]
        print('prob weight')
        random_pred=random.choice(index,prob_weight)
        print(random_pred)
        random_index.append(sta_list[random_pred])
    X['prediction'] = random_index
    X.to_csv('prediction.csv', sep=',', index=False)
```
5.4 Understanding Inventory
From the trip data, we compute the total number of bikes present in the bike sharing system and at the end of a day we track the station ID of each bike. From this, we identify the current inventory of bikes of each station. Now, if we can predict how many bikes are coming to a particular station using Section 5.3.

5.5 Interactive Visualization of Network Dynamics
We used Leaflet maps and D3.js (e.g. Figure 17) to develop a web-based interactive visualization that depict network dynamics. The main purpose of this visualization is to support incentive decision making so that force-balanced trips diminish.

Ideally, this visualization would be updated as often as decisions should be made. For simplicity, we decided to use the four time periods identified in Section 4.3, so that we would place a different incentive policy for each of these periods. However, these periods could be shrunk to 1 hour or less.

Figure 16 contains our interactive visualization tool of the dynamics of the bike-sharing network. It displays the following information:

1. Station ID and name.
2. Bike inventory at station (either from real-time updates or predicted).
3. Expect demand of bikes for the following time period.
4. Inventory/demand balance, which is the difference between inventory and expected demand.
5. Number of bikes expect to arrive to the station in a established time.

Information for the each station is displayed visually in the map so that an overview is available for the user. Colors and sizes provide the user with information scent so that he or she will be able to quickly understand the stations that require more attention.

(1) Expected demand is mapped to circle size. The bigger the more expected demand.
(2) Inventory/demand balance is mapped to circle color. Color interpolation from red to green representing from negative to positive balance, being black close to zero.

Examples of how this visual information is useful for decision making are the following. If there is a green circle (positive balance), there is no need to intervene. However, if there is a red circle, we may want to intervene this station (by incentivizing trips to this station). This decision is aided with the number of bikes of arriving. If there are enough bikes arriving, no or minimal intervention may be needed.

6 RESULTS
We discuss our results and evaluation in the following:

6.1 Demand Forecasting
For demand forecasting, since our objective is to forecast the demand of bikes at a particular time period, we use RMSE for evaluating our forecasting result. For each station, we validate our LSTM model on 10% validation set (the default validation split in Keras). Also, we arrange our test data by selecting a summer date, a winter day, weekday, weekends, and each four time periods of a station. In total we build 85 LSTM model for 85 high demanding stations. Figure 18 shows the comparative plot of loss for training and validation set after fitting history data of station 141. Note, both training and validation loss decreases with each epoch which indicates a better fit of model. We choose loss plot of station id 141 because this is the highest demanding station.

We summarize the result of demand forecasting in the following:

- Dataset size: 0.15M
- Test data size: 10% of the data set, considering different weather, weekends and time periods
- Average RMSE on test data (considering all stations model): 8.39
- Lowest RMSE on test data: 2.48, for station Id 109
- Highest RMSE on test data: 33.09 for station Id 90
- Average RMSE on training data (considering all stations model): 4.19

Note that our demand forecasting model is producing better results. The lower the RMSE, the better the result is.

6.2 Destination Prediction
We evaluate our destination prediction model using root mean squared error (RMSE). The lower the value the better is the result. Our aim for building destination prediction model is to predict the number of bikes coming to a station. In Figure 19 left figure, we observe that, none of observed destination Id matches with the predicted destination Id for the same origin and time period. However, note that the number of predicted trips are equal to the number of observed trips for the same origin-destination pair and at the same time. This is because for our predictive model we are using a random generator whose probability distribution for destination station is same as historical data but random generator could not predict the same destination Id as the observed Id at the same trip.
On another note, building a simple classifier without using random generator model yields the same destination Id for all trips, which in turn yields lower accuracy rate, i.e. 5% in our case. In Figure 20 we observe that, in index 1,2,4 for the same origin station Id 4 and time period 1, the simple decision tree classifier predicts destination Id 55 for all the trips, which is not our goal. This is because, for every trip classifier considers only the station having the highest probability score.

We evaluate our results by calculating RMSE of actual number of bikes arriving at a station at a particular time with predicted number of bikes arriving at the same station at that time. Average test RMSE for prediction model is 62.6, which is high. If we notice Figure 21, we observe that for some of the origin-destination (OD) pair, for e.g., 53-67, 178-139 it gives high RMSE result, which in turn produces high RMSE score in average.

### 6.3 Interactive visualization

The resulting interactive visualization is shown in Figure 16. The user is able to zoom and pan across them map, and click on the circles that represent the visualizations to see details about the selected station. In addition, there is a PLAY button to visualize an animation that will show how each circle represents a station varies in size and color, which is mapped to demand and inventory/demand balance. More
Our initial hypothesis was that weather would be predictive for demand, thus, we included it in our models. Figure 22 contains our visualization for two days of different seasons. It is noticeable that when it is Summer, there is more demand of bikes in the overall system than in Winter, which is represented in circle sizes. Additionally, demand hubs change.

7 SOCIETAL IMPACT
In Bike sharing system, efficient self-balancing and force balancing can save a huge operational cost for every station. If a station can predict its demand, inventory of bikes for several hours and knows how many bikes are coming within a few hours, it can easily decide whether force-balancing the bikes would be profitable or not.

8 FUTURE WORK
In future, we expand our work by considering real time data for inventory modeling instead of using historical data. Using real time data we can update the inventory of stations at every hour. Also, we aim to formulate an optimization problem to automate decision of force-balance using demand, inventory and number of arriving bikes. Moreover, we also think to extend our framework for trip data sets other big cities like New York, Chicago, Washington D.C. from BlueBikes trip data.

9 CONCLUSION
To understand the network dynamics of bike sharing system, we first create a interactive visualization showing the stations in a real time map. To make decision making easier, we model demand forecasting for every high demanding stations, predict destinations or number of stations coming to a station along with their arrival time and obtain the inventory of bikes available at each station at a particular time period. Our framework suggests practical information to the operator to give incentive as well as help understand how the bike sharing system varies with time. However, we have some limitations in our framework. First, due to unavailability of enough data we consider only 85 high demanding stations instead of all the stations in system. Second, the RMSE value of our destination prediction model is very high, which makes the model vulnerable. In future we consider other factors like shortest path, user preference etc. to build a better prediction model.

10 WORK DIVISION
The work was divided as in Table 3, being Alex the abbreviated name of Alexander.

REFERENCES
Understanding network dynamics of a bike-sharing system to enhance inventory self-balancing

Figure 21: Reason for getting high RMSE score in destination prediction model

Figure 22: Dynamics visualization for a day of Summer 2018 (left) and a day of Winter 2018 (right). Both in afternoon peak.


<table>
<thead>
<tr>
<th>Work Plan</th>
<th>Member</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data collection &amp; analysis</td>
<td>Bumsik</td>
</tr>
<tr>
<td>Visualization</td>
<td>Alex</td>
</tr>
<tr>
<td>Data pre-processing and cleaning</td>
<td>Anika, Bumsik</td>
</tr>
<tr>
<td>Demand forecasting, destination prediction</td>
<td>Anika</td>
</tr>
<tr>
<td>Writing</td>
<td></td>
</tr>
<tr>
<td>Introduction, problem formulation</td>
<td>Bumsik</td>
</tr>
<tr>
<td>Data description</td>
<td>Bumsik</td>
</tr>
<tr>
<td>Methodology</td>
<td>Alex, Anika</td>
</tr>
<tr>
<td>Related Work</td>
<td>All</td>
</tr>
<tr>
<td>Experiments, results</td>
<td>Alex, Anika</td>
</tr>
</tbody>
</table>

Table 3: Work division of the project