**INTERNSHIP REPORT**

**Title: SMART NETWORK SYSTEMS (Parking Prediction)**

|  |  |
| --- | --- |
| Period: | 1/06/2022 – 1/09/2022 |
| Host MRG: | CYENS |
| Supervisor: | Prof. Vasos Vassiliou |
| Intern: | Murtala Ibrahim Kankarofi |
| Main theme: |  |

|  |
| --- |
| 1. TABLE OF CONTENTS   * Acknowledgement * Context * Aim and objectives * Methodology and work  1. Nicosia Municipality Dataset 2. Kaggle Dataset  * Problems and Issues * Results and Conclusion |
| 2. ACKNOWLEDGEMENT  I am writing to express my gratitude to Prof Vasos Vassiliou and Dr Iacovos Ioannou for giving me this amazing opportunity to learn and do something new, which makes me proud of what I have achieved. They have worked tirelessly with me through this journey. I started with no knowledge of object-oriented programming, but they coached me and pushed me to be from zero to hero. No words can express my appreciation towards you, yet I still say Thank you for everything. |
| 3. Context  Smart parking enables citizens to explore, exploit and enjoy the urban environment. With a system that will predict the parking availability and use the LSTM DNN model to be implemented via REST services, the project consists entirely of two different datasets, which are from Nicosia Municipality and Kaggle, and was used in designing the smart parking system. This intelligence derived from data gathering from city infrastructure has helped to make a model that manages our streets and experiences in them. The advantages of smart parking systems are the following: it plays a vital role in developing urban environments for car owners and others, including environmental merits. Within this report is the full detail on how this was created/implemented. |
| 4. Aim and objectives  Parking space has been and is still a significant issue that we face every day as we aim to develop methods to curb the problem. This research aims to develop a way that will ease the difficulties it has in our daily lives.  The objective of this research is to assist in developing an urban environment where drivers find safety and ease in moving around and reaching their destination within a convenient time. Also, it will add an impact on the environmental aspects due to carbon emission, assist law enforcement, reduce traffic congestion, operational costs and so on. |

|  |
| --- |
| 5. Methodology and Work  Due to issues in gathering the data and the model’s need to be accurate, two different datasets were used to train the model independently through the same cycle to ensure a good result.  Nicosia Municipality Dataset:  This is the Dataset provided by the Nicosia municipality with parking space details to create a model. It consists of 4 parking IDs (parking 2, parking 3, parking 4, & parking 5). Each ID consists of id, title, description, spaces, updated-on, and geolocation information. It is supposed to be updated every 5 minutes to get an accurate reading of the parking spaces. This will help us understand the times it is busy and less busy while calculating the probabilities in making our model.  The first method was data extraction. This has been done with complete code shown in the file “data.py”. I imported the request library in python to get the data from the provided URL. The python code with the request library requests to the REST to servers to collect all the information on the page and transfer to you. The information on the web server is in an XML format; However, using python to parse it is best using the Element tree library. Parsing is an essential step for it helps you move through the document faster and efficiently access materials based on your needs. The Element tree arranges the data in logical tree manner which root is the central part. An XML format is divided into root, child, sub-child ETC all connected to the main, which is the root element. Now you create a class which will act as a blueprint for your data extraction. In this XML file, only four parking data was provided, but the code was made in case we have additional parking.  Furthermore, a place to save the data is needed. I used the library MySQL Connector to connect it to MySQL database to save my data. To achieve this, I created a database in MySQL called “parkingdata”. Two tables were also created in which they were named parking and parking details. For the parking table, the data saved from the XML file are ID, title, description, and geolocation with the parking ID as the primary key while for parking details table, the data saved are parking ID (Foreign key from parking table), spaces and updated-on. To connect to MySQL, username and root must be defined.  Finally, the insert method to complete the data saving process into my database. Since the parking table has a primary key of parking ID, a duplicate means an error in the code. So, "insert ignore" is added to the code of insertion. For the parking details table, an auto-increment of Parking Details Id was added and acted as the primary key with the ID being the foreign key. Commit the change, close cursor, and close the MySQL connection. A Cron job was set to run every 5 minutes every day continuously to monitor and save any change or update from the given server. It Worked successfully.  The next step is data cleaning before building the model. The saved data in the MySQL database was exported as a CSV file named "parking\_details.csv". Due to problems faced and will be discussed in the next part of the report, some procedures were taken to feed good data for our model. Only parking ID 2 data was used because it had more accurate information. After reading the CSV file, a slicing method was used to only access parking 2. No missing data was found using is “is null ()” method. The only parameters needed now to feed our model will be updated-on and spaces. All other columns were popped from the table using the pandas pop method. Using Mat plot lib, we can plot a graph showing us visually the evolution of the availability of parking spaces concerning updated-on time frame. A subplot was also made to show the evolution change daily to have a better inspection of detailed changes in spaces. Using the described method, we view the statistical representation of the Dataset, which includes the mean and standard deviation, in which needs to use in data normalization, which is a method of subtracting the mean and dividing by the standard deviation of each feature and are only computed using the training data so that the other models won’t have access to it. Before normalization, we split the data (70%, 20%, and 10%) for training, validating, and testing respectively.  In this category, we will apply data windowing. Data windowing will make a set of predictions based on a window of consecutive data samples. The main features of the input windows are the width, offset, and the features used inputs, labels, or both. This can be used to build a variety of CNN, DNN and RNN models for (single-output and multi-output predictions) and (single-time step and multi-time step predictions). Given a list of consecutive inputs, the split window method will convert them to a window of inputs and a window of labels. Finally, this makes dataset method will take a time series Data Frame and convert it to a TF data Dataset of (input window, label window) pairs using the TF KERAS Utils Timeseries dataset from array function. The Window Generator object holds training, validation, and test data. Add properties for accessing them as TF data Datasets using the make dataset method you defined earlier. Also, add a standard example batch for easy access and plotting. Now, the Window Generator object gives you access to the TF data Dataset objects, so you can easily iterate over the data. The Dataset element spec property tells you the dataset elements structure, data type, and shape. The simplest model you can build on this sort of data is one that predicts a single feature's value—1 time step (one hour) into the future based only on the current conditions. Before building a trainable model, it would be good to have a performance baseline as a point for comparison with the later, more complicated models. The first task is to predict parking availability 1 hour into the future, given the current values of all features. The current values include spaces. So, create a wider Window Generator that generates windows 24 hours of consecutive inputs and labels at a time. The new wide window variable doesn't change the way the model operates. The model still makes predictions one hour into the future based on a single input time step. Here, the time axis acts like the batch axis. Each prediction is made independently with no interaction between time steps. A Recurrent Neural Network (RNN) is a type of neural network well-suited to time series data. RNNs process a time series step-by-step, maintaining an internal state from time-step to time-step and then calling the wide window plot(LSTM model), we have our plot with a prediction over the labels based on the inputs provided.  Kaggle Dataset:  A dataset is given on the Kaggle website, a platform for machine learning and data science projects. This data which is a bit different from the data provided by Nicosia Municipality, has only three parameters: parking ID, spaces and updated-on. There are various ways to get the data to your python workspace, but I used the Kaggle API. First step is to set up your Kaggle API on your working PC (provided on the website) and import the Kaggle library. Authenticate the connection and navigate the website to find the Dataset convenient for your work. In this case, I used the “mypapit/klccparking” and downloaded it. It was saved as “klccparking.zip”. Next is to import the Zip File and extract the zip folder downloaded. The file is provided in a ".txt" format, so after the extraction, it will be saved on your PC as "parking-klcc-2016-2017.txt". Open the file, but you will notice it has no named columns and some information in the data is not numbers where it was supposed to be considered as parking spaces, and it is called OPEN and FULL. According to the Kaggle website, full means the parking has no place to park, with the highest number considered as 5,500 and open is considered NaN. Using the “. replace ()” function, we can edit all the places in our file that has these problems to fix our data. This function was used twice in my work with each edit; the file was saved as output1 and output2. Immediately the output2 file, the file is then converted to a ".csv" file. During the conversion, using the “csv writer”, we add names to our columns, respectively: parking ID, spaces and updated-on. The file is then saved as "parking-klcc-2016-2017.csv".  Upon this completion, you can import all the necessary libraries same as the previous steps in Nicosia Municipality Dataset. However, in this case we have NaN values which we don’t have in the previous Dataset. We use the “.fillna().mean()" to convert all the NaN values to the mean of the column of the Dataset. We have fully eliminated all unnecessary data from our final data before building our model. Finally, the same steps will be followed to create the model for prediction as we did in our previous Dataset to have a prediction. It would be best if you have an LSTM DNN model that will plot a graph with prediction based on the data you give. It will be different from the previous model you created because the data provided to the same model is different. |
| 6. PROBLEMS AND ISSUES   * It took days to update data on the server * Most parking spaces didn’t get updated throughout the project cycle * Delayed response from the server technicians * Unavailable to get the complete data even though formal requests were made * Attempts to find alternatives * Delayed the project |
| 7. Results and Conclusions  In this paper, two different and large datasets used in real-world were proposed and used to predict parking availability using the Long Short-Term Memory (LSTM) model, which is the Nicosia Municipality Dataset and Kaggle Dataset, whereby the parameters used to make the model was solemnly depending on the updated-on and parking spaces. The only weakness in this paper is the error in the Nicosia municipality Dataset, which needs attention in the future. However, a satisfactory model was built and made a prediction which performs better overtime. Feeding good data to your model to get achievable results is of utmost importance.  To understand the pattern of time-series occurrences, we only employ the most fundamental LSTM. More potent and beneficial recurrent neural networks are proposed to address such problems with faster speed and higher accuracy, considering the recent rapid progress of deep learning techniques. As a result, the future strategy should include using various strategies to increase the system's efficiency. |