

# **A UNIFIED APPROACH FOR PREDICTING TAXI DEMAND USING L-CNN**

**at**

**Sathyabama Institute of Science and Technology  
(DEEMED TO BE UNIVERSITY)**

Submitted in partial fulfillment of the requirements for the award  
of Bachelor of Engineering Degree in Computer Science and  
Engineering

**By**

**PARIDHI. J (Reg. No. 37110544)**

**NIJESH. S (Reg. No. 37110509)**



**DEPARTMENT OF COMPUTER SCIENCE AND  
ENGINEERING SCHOOL OF COMPUTING  
SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY  
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,  
CHENNAI – 600119, TAMILNADU**

**APRIL 2021**



# **SATHYABAMA**

**INSTITUTE OF SCIENCE AND TECHNOLOGY  
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade “A” by NAAC**

**(Established under Section 3 of UGC Act, 1956)**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI– 600119**

**[www.sathyabamauniversity.ac.in](http://www.sathyabamauniversity.ac.in)**



---

## **SCHOOL OF COMPUTING**

### **BONAFIDE CERTIFICATE**

This is to certify that this Project Report is the bonafide work of **PARIDHI. J (Reg. No. 37110544) and NIJESH. S (Reg. No. 37110509)** who carried out the project entitled **“A UNIFIED APPROACH FOR PREDICTING TAXI DEMAND USING L-CNN”** under our supervision from Dec 2020 to Mar 2021.

**Internal Guide**

**Dr. S.Ashok Kumar, M.E., Ph.D.,**

**Head of the Department**

**Dr. S.VIGNESHWARI, M.E., Ph.D.,**

---

**Submitted for Viva voce Examination held on \_\_\_\_\_**

**Internal Examiner**

**External Examiner**

## **DECLARATION**

**We PARIDHI. J (Reg. No. 37110544) and NIJESH. S (Reg. No. 37110509), hereby declare that the Professional Training Report on "AUNIFIED APPROACH FOR PREDICTING TAXI DEMAND USING L-CNN "**

done under the guidance of **Dr. S.Ashok Kumar, M.E., Ph.D.**, at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

**DATE:**

**PLACE: CHENNAI**

**SIGNATURE OF THE CANDIDATE**

## ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Sathyabama Institute Of Science and Technology Management** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. SASIKALA, M.E., Ph.D., Dean, School of Computing** and **Dr. S. VIGNESHWARI, M.E., Ph.D., Head of the Department, Department of Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. S.ASHOK KUMAR, M.E., Ph.D.,** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department of **Computer Science and Engineering** who were helpful in many ways For the Completion of the project.

## **ABSTRACT**

The taxi administration industry is blasting for the most recent few years and it is relied upon to develop in the not so distant future. Cab drivers need to choose where to pause for travelers as they can get somebody when conceivable. Travelers likewise favor a snappy taxi administration at whatever point required. The control focal point of the taxi administration chooses the bustling territory to be concentrated. In the current framework, here and there the cabs were dispersed over the bigger territory missing the time based occupied zone like Air terminal, Business territory, school zone, Train stations and so on. Powerful taxi assignment can support the the drivers and travelers to limit the hold up time to locate one another. In the proposed framework, the future interest can be anticipated utilizing Recurrent Neural Networks based model that can be prepared with given chronicled information. It can serve more clients in a brief time frame by sorting out the accessibility of taxi. The informational collection incorporates GPS area and different properties of the taxi like slanted edge, pickup point and so forth. This model is utilized to foresee the interest for a specific time in various territories of the city.

## TABLE OF CONTENT

| CHAPTER NO | TITLE                              | PAGE NO |
|------------|------------------------------------|---------|
|            | ABSTRACT                           | V       |
|            | LIST OF FIGURES                    | Vii     |
|            | LIST OF ABBREVIATIONS              | Viii    |
| 1          | INTRODUCTION                       | 1       |
| 2          | LITERATURE REVIEW                  | 3       |
| 3          | METHODOLOGY                        | 10      |
|            | 3.1 SYSTEM REQUIREMENTS            | 10      |
|            | 3.1.1 HARDWARE REQUIREMENTS        | 10      |
|            | 3.1.2 SOFTWARE REQUIREMENTS        | 10      |
|            | 3.2 SYSTEM ANALYSIS                | 10      |
|            | 3.2.1 EXISTING SYSTEM              | 10      |
|            | 3.2.2 PROPOSED SYSTEM              | 11      |
|            | 3.3 MODUES                         | 13      |
|            | 3.3.1 DATASET PREPROCESSING        | 13      |
|            | 3.3.2 CONVOLUTIONAL NEURAL NETWORK | 14      |
|            | 3.3.3 PREDICTION                   | 19      |
|            | 3.3.4 RESULT PRESENTATION          | 20      |
|            | 3.4 ARCHITECTURE DIAGRAM           | 20      |
|            | 3.5 ATTRIBUTES                     | 21      |
| 4          | RESULT AND PERFORMANCE ANALYSIS    | 23      |
| 5          | CONCLUSION AND FUTURE WORK         | 25      |
|            | REFERENCES                         | 26      |
|            | APPENDIX                           | 29      |
|            | A SOURCE CODE                      | 29      |
|            | B PAPER WORK                       | 34      |

## LIST OF FIGURES

| FIGURE NO | FIGURE NAME                                | PAGE NO |
|-----------|--------------------------------------------|---------|
| 3.1       | Constructing a weight filter in LCNN       | 16      |
| 3.2       | Look-up Based Convolutional neural network | 17      |
| 3.3       | Architecture Diagram                       | 20      |
| 4.1       | Data Preprocessing                         | 23      |
| 4.2       | Output of Taxi demand prediction           | 24      |

## LIST OF ABBREVIATIONS

|      |                                            |
|------|--------------------------------------------|
| CNN  | Convolutional neural network               |
| CSV  | Comma seperated values                     |
| LCNN | Look-up based convolutional neural network |
| LSTM | Long Short Term Memory                     |
| RNN  | Recurrent neural network                   |
| RAM  | Random access memory                       |
| UML  | Unified modeling languages                 |

# **CHAPTER 1**

## **INTRODUCTION**

Cab drivers need to choose where to wait tight for the travelers with the goal that they can pick somebody rapidly. So also, travelers additionally need to discover their taxis rapidly. Dispatching the taxi productively help both the clients and drivers. Compelling dispatching of taxi assists with diminishing sitting tight time for clients, also as drivers. A driver won't have enough information about where to hold up to get explorers quickly. A taxi center can form and send the required number of cabs to the zone subject to the recorded data. The valid data uses Global Positioning System (GPS) and foresee future intrigue. In Tokyo, today this system is lessening the holding on schedule for the customers, quickly respond to the sudden contrast in solicitations and it vanquishes any deterrent between the experienced drivers and tenderfoot drivers. These favorable circumstances grant the taxi organization to achieve most prominent favorable position. A steady taxi demand desire is proposed here besides, at this moment, data is used to foresee the future enthusiasm for taxis in a particular spot at a explicit time. A segment of the persistent goals consolidate supervising task force of taxi to swarmed zone, convincing use of advantages for decrease delaying time, server more customers in a brief time period by sifting through open taxi. Our system uses GPS region also, various properties of the taxi like pickup point, drop point, etc to anticipate taxi demand. A model is readied using a discontinuous neural framework. The discontinuous neural frameworks are used in talk affirmation programming. The discontinuous neural frameworks are used for progressive data. It is being used in Google's voice search and Apple Siri singular partners. This figuring is the achievement of significant learning in past quite a while. Discontinuous neural frameworks produce desires result for the progressive data. Taxi request expectation is a period arrangement investigation issue. It is a feed forward work neural system and the data moves starting with one system then onto the next organize and from the info layer to the yield layer through the shrouded layer. The distinction between the typical neural system and the intermittent neural organize is that in the intermittent neural system, the data pushes through the circle. In the framework, the repetitive neural system is utilized also, Python language is favored in

light of the fact that it has a huge assortment of machine learning libraries. The informational collection might contain void qualities, negative qualities or mistake. Informational collection is cleaned in the preprocessing. The preprocessing techniques include expelling records which are not finished. When the cleaned informational collection is accessible it is set up to be sustained to the AI calculation. Intermittent Neural Networks take the past hub yield or concealed states as data sources. RNNs are helpful as their middle of the road esteems (state) can store data about past contributions for a period interim. The fundamental element of a Recurrent Neural System (RNN) is that the system have at any rate one criticism association, so the initiations can stream round in a circle shrewd way. That empowers the systems to learn arrangements and to do worldly preparing, e.g., perform arrangement acknowledgment/multiplication or worldly affiliation/forecast. Intermittent neural system models can have numerous variated structures. One normal the sort that comprises of a standard Multi-Layer Perceptron (MLP) in addition to included bends. These can venture the incredible non-direct mapping capacities of the MLP, and furthermore have some type of memory. Since one can consider intermittent arranges as far as their properties as dynamical frameworks, it is expected to get some information about their security, perceptible and comparability. The service industry is booming for the last couple of years and it is expected to grow in the near future. One of the important natures of the business is the serve the customer. To effectively utilize the resource at hand is the key factor. Businesses are using advanced technology to achieve this. To predict the high demand need of pickup location for taxi services based on their previous history

## CHAPTER 2

### LITERATURE SURVEY

**Project Title 1** : A Long Short-Term Memory Recurrent Neural Network Framework for Network Traffic Matrix Prediction

**Year of Publish** 2017

**Author Name** : Abdelhadi Azzouni and Guy Pujolle

**Abstract** :

Network Traffic Matrix (TM) prediction is defined as the problem of estimating future network traffic from the previous and achieved network traffic data. It is widely used in network planning, resource management and network security. Long Short-Term Memory (LSTM) is a specific recurrent neural network (RNN) architecture that is well-suited to learn from experience to classify, process and predict time series with time lags of unknown size. LSTMs have been shown to model temporal sequences and their long-range dependencies more accurately than conventional RNNs. In this paper, we propose a LSTM RNN framework for predicting Traffic Matrix (TM) in large networks. By validating our framework on real-world data from GEANT network, we show that our LSTM models converge quickly and give state of the art TM prediction performance for relatively small sized models.

**Project Title 2** : DeepPool: Distributed Model-free Algorithm for Ride-sharing using Deep Reinforcement Learning

**Year of Publish** 2019

**Author Name** : Abubakr Alabbasi, Arnob Ghosh, and Vaneet Aggarwal

**Abstract** :

The success of modern ride-sharing platforms crucially depends on the profit of the ride-sharing fleet operating companies, and how efficiently the resources are managed. Further, ride-sharing allows sharing costs and, hence, reduces the congestion and emission by making better use of vehicle capacities. In this work, we develop a distributed model-free, DeepPool, that uses deep Q-network (DQN) techniques to learn optimal dispatch policies by interacting with the environment. Further, DeepPool efficiently incorporates travel demand statistics and deep learning models to manage dispatching vehicles for improved ride sharing services. Using real-world dataset of taxi trip records in New York City, DeepPool performs better than other strategies, proposed in the literature, that do not consider ride sharing or do not dispatch the vehicles to regions where the

future demand is anticipated. Finally, DeepPool can adapt rapidly to dynamic environments since it is implemented in a distributed manner in which each vehicle solves its own DQN individually without coordination

**Project Title 3** : Recommendation Systems for Tourism Based on Social Networks: A Survey

**Year of Publish** 2019

**Author Name** : Alan Menk \_ Laura Sebastia \_ Rebeca Ferreira

**Abstract** :

Nowadays, recommender systems are present in many daily activities such as online shopping, browsing social networks, etc. Given the rising demand for reinvigoration of the tourist industry through information technology, recommenders have been included into tourism websites such as Expedia, Booking or Tripadvisor, among others. Furthermore, the amount of scientific papers related to recommender systems for tourism is on solid and continuous growth since 2004. Much of this growth is due to social networks that, besides to offer researchers the possibility of using a great mass of available and constantly updated data, they also enable the recommendation systems to become more personalised, effective and natural. This paper reviews and analyses many research publications focusing on tourism recommender systems that use social networks in their projects. We detail their main characteristics, like which social networks are exploited, which data is extracted, the applied recommendation techniques, the methods of evaluation, etc. Through a comprehensive literature review, we aim to collaborate with the future recommender systems, by giving some clear classifications and descriptions of the current tourism recommender systems.

**Project Title 4** : A study and analysis of recommendation systems for location-based social network (LBSN) with big data

**Year of Publish** 2016

**Author Name** : Murale Narayanan [a,\\*](#), Aswani Kumar Cherukuri [b](#)

**Abstract** :

Recommender systems play an important role in our day-to-day life. A recommender system automatically suggests an item to a user that he/she might be interested in. Smallscale datasets are used to provide recommendations based on location, but in real time, the volume of data is

large. We have selected Foursquare dataset to study the need for big data in recommendation systems for location-based social network (LBSN). A few quality parameters like parallel processing and multimodal interface have been selected to study the need for big data in recommender systems. This paper provides a study and analysis of quality parameters of recommendation systems for LBSN with big data.

**Project Title 5** : An Effective Taxi Recommender System Based on a Spatiotemporal Factor Analysis Model

**Year of Publish** 2014

**Author Name** : Yu-Ling Hsueh, Ren-Hung Hwang, and Yu-Ting Chen

**Abstract** :

The taxi fleet management system based on GPS has become an important tool for efficient taxi business. It can be used not only for the sake of fleet management, but also to provide useful information for taxi drivers to earn more profit by mining the historical GPS trajectories. In this paper, we propose a taxi recommender system for next cruising location which could be a value added module of the fleet management system. In the literature, three factors have been considered in different works to provide the similar objective, which are distance between the current location and the recommended location, waiting time for next passengers, and expected fare for the trip. In this paper,

in addition to these factors, we consider one more factor based on drivers experience which is the most likely location to pick up passengers given the current passenger drop off location. A location-to-location graph model, referred to as OFF-ON model, is adopted to capture the relation between the passenger get-off location and the next passenger get-on location. We also adopted a ON-OFF model to estimate the expected fare for a trip started at a recommended location. A real world dataset from CRAWDAD is used to evaluate the proposed system. A simulator is developed which simulates cruising behavior of taxis in the dataset and one virtual taxi which cruises based on our recommender system. Our simulation results indicate that although the statistics of historical data may be different from real-time passenger requests, our proposed recommender system is still effective on recommending better profitable cruising location.

**Project Title 6** : CallCab: A Unified Recommendation System for  
Carpooling and Regular Taxicab Services

**Year of Publish** 2017

**Author Name** : Desheng Zhang and Tian He, Yunhuai Liu

**Abstract** :

Carpooling taxicab services hold the promise of providing additional transportation supply, especially in extreme weather or rush hour when regular taxicab services are insufficient. Although many recommendation systems about regular taxicab services have been proposed recently, little research, if any, has been done to assist passengers to find a successful taxicab ride with carpooling. In this paper, we present the first systematic work to design a unified recommendation system for both regular and carpooling services, called CallCab, based on a data driven approach. In response to a passenger's request, CallCab aims to recommend either (i) a vacant taxicab for a regular service with no detour, or (ii) an occupied taxicab heading to the similar direction for a carpooling service with less detour, yet without assuming any knowledge of destinations of passengers already on occupied taxicabs. To analyze these unknown destinations of occupied taxicabs, CallCab generates and refines taxicab trip distributions based on GPS datasets and context information collected in the existing taxicab infrastructure. To improve CallCab's efficiency to process such a big dataset, we augment the efficient MapReduce model with a Measure phase tailored for our application. We evaluate CallCab with a real world dataset of 14, 000 taxicabs, and results show that compared to ground truth, CallCab can reduce 64% of the total mileage to deliver all passengers and 63% of passenger's waiting time.

**Project Title 7** : An effective taxi recommender system based on a spatio-temporal  
factor analysis model

**Year of Publish** 2015

**Author Name** : Ren-Hung Hwang, Yu-Ling Hsueh

**Abstract** :

The taxi fleet management systems based on GPS have become an important tool for taxi businesses. Such systems can be used not only for fleet management, but also to provide useful information for taxi drivers to increase their profits by mining historical GPS trajectories. In this paper, we propose a taxi recommender system for determining the next cruising location, which could be a value-added module in fleet management systems. In the literature, three factors have

been considered in different studies to address a similar objective: distance between the current location and the recommended location, waiting time for the next passengers, and expected fare for the trip. In this paper, in addition to these factors, we consider one key factor based on driver experience: what is the most likely location to pick up passengers, given the current passenger drop off location. A location-to-location graph model, referred to as an OFF–ON model, is adopted to capture the relation between the passenger drop-off location and the next passenger get-on location. We also adopt an ON–OFF model to estimate the expected fare for a trip that begins at a recommended location. A real-world dataset from CRAWDAD is used to evaluate the proposed system. A simulator that simulates the cruising behavior of taxies in the dataset and a virtual taxi that cruises based on our recommender system is developed. Our simulation results indicate that although the statistics of the historical data may be different from real-time passenger requests, our recommender system is still effective in terms of recommending more profitable cruising locations

**Project Title 8** : Online Learning of a Dirichlet Process Mixture of Generalized Dirichlet Distributions for Simultaneous Clustering and Localized Feature Selection

**Year of Publish** 2012

**Author Name** : Wentao Fan, Nizar Bouguila

**Abstract** :

Online algorithms allow data instances to be processed in a sequential way, which is important for large-scale and real-time applications. In this paper, we propose a novel online clustering approach based on a Dirichlet process mixture of generalized Dirichlet (GD) distributions, which can be considered as an extension of the  $\text{\_nite}$  GD mixture model to the  $\text{in\_nite}$  case. Our approach is built on nonparametric Bayesian analysis where the determination of the number of clusters is sidestepped by assuming an  $\text{in\_nite}$  number of mixture components. Moreover, an unsupervised localized feature selection scheme is

integrated with the proposed nonparametric framework to improve the clustering performance. By learning the proposed model in an online manner using a variational approach, all the involved parameters and features saliencies are estimated simultaneously and effectively in closed forms. The proposed online  $\text{in\_nite}$  mixture model is validated through both synthetic data sets and two challenging real-world applications namely text document clustering and online human face detection.

**Project Title 9** : Article Interest Aware Location-Based Recommender System Using Geo-Tagged Social Media

**Year of Publish** 2016

**Author Name** : Basma Al Banna \*, Mahmoud Sakr, Sherin Moussa and Ibrahim Moawad

**Abstract** :

Advances in location acquisition and mobile technologies led to the addition of the location dimension to Social Networks (SNs) and to the emergence of a newer class called Location-Based Social Networks (LBSNs). While LBSNs are richer in their model and functions than SNs, they fail so far to attract as many users as SNs. On the other hand, SNs have large amounts of geo-tagged media that are under-utilized. In this paper, we propose an Interest-Aware Location Based Recommender system (IALBR), which combines the advantages of both LBSNs and SNs, in order to provide interest-aware location-based recommendations. This recommender system is proposed as an extension to LBSNs. It is novel in: (1) utilizing the geo-content in both LBSNs and SNs; (2) ranking the recommendations based on a novel scoring method that maps to the user interests. It also works for passive users who are not active content contributors to the LBSN. This feature is critical to increase the number of LBSN users. Moreover, it helps with reducing the cold start problem, which is

a common problem facing the new users of recommender systems who get random unsatisfying recommendations. This is due to the lack of user interest awareness, which is reliant on user history in most of the recommenders. We evaluated our system with a large-scale real dataset collected from foursquare with respect to precision, recall and the f-measure. We also compared the results

with a ground truth system using metrics like the normalized discounted cumulative gain and the mean absolute error. The results confirm that the proposed IALBR generates more efficient recommendations than baselines in terms of interest awareness.

**Project Title 10** : **ANALYSIS OF REAL-TIME BASED RECOMMENDATION SYSTEM FOR TAXI RIDESHARING: REVIEW**

**Year of Publish** 2019

**Author Name** : Ms. Sneha Dinesh Jawanjal1

**Abstract** :

Taxi is an important transportation mode, that has been introduced to help the people reach their destination with convenience and ease. Taxi delivering millions of passengers to different locations in urban areas. Now a days, taxi demands are usually much higher than the number of taxis of major cities, resulting in that many people spend a long time on roadsides before getting a taxi. Increasing the number of taxis is an obvious solution. But it brings some adverse effects, e.g., causing additional traffic on the road surface and more energy consumption, and decreasing taxi drivers income. The Transportation Problem is one of the subclass problems in travelling. The sensors installed in each vehicle are providing new opportunities, which in return deliver information for real time decision making. Intelligent transportation systems for taxi dispatching and time-saving route finding are already exploring this sensing data. The present implementation has passengers request a taxi when they depart; then they wait until they are being picked up and delivered to their destination. Eventually, taxicabs will drive to the next request allowed to them, or remain idle until the next request arrives.

## CHAPTER 3

### METHODOLOGY

#### SYSTEM REQUIREMENTS

##### *Hardware Requirements*

- Hard disk : 500 GB and above.
- Processor : i3 and above.
- Ram : 4GB and above.

##### *Software Requirements*

- Operating System : Windows 7 and above (64-bit)
- Python : 3.6

#### SYSTEM ANALYSIS

##### *Existing System:*

TAXI drivers need to decide where to wait for passengers in order to pick up someone as soon as possible. Passengers also prefer to quickly find a taxi whenever they are ready for pickup. The control center of the taxi service decides the busy area to be concentrated. Sometimes the taxi were scattered across the larger area missing the time based busy area like Airport, Business area, school area, Train stations etc.,.

*Recurrent Neural Networks:* The sequential nature of taxi demand data leads us to the choice of a model that can handle time series data. Recurrent neural networks (RNNs) are one of the most popular models that can process sequential data very well. The idea behind RNNs is to store relevant parts of the input and use this information while predicting the output in the future. Unlike feed-forward neural networks that only predict the output based on the current input, RNNs contain memory in which some important information from the past inputs can be stored. For instance, when we train RNNs on a language modeling task in which we generate a text one character at each timestep, it is better to store what characters the network has predicted in the previous time steps since the next character is dependent on the previous predictions. RNNs

are called recurrent because they perform the same computation on every element of a sequence, with the output being conditioned on the previous computations. A typical RNN structure is given in Fig. 2. As we can see, the RNN processes input  $x$ , stores hidden state  $h$  and outputs  $y$  at each time-step  $t$ . A loop allows information to be passed over from one step to the next. All  $W$ s are the shared weights among different time-steps. For training these weights, we unroll the network for finite number of time-steps as shown in Fig. 3. When the network is unrolled, it is more clear why it is being used for sequence learning and how the information is being passed to the future. The computation at each time-step can be formulated as follows:

- $x_t$  is the input at time-step  $t$ .
- $h_t$  is the hidden state at time-step  $t$ . It is calculated based on the previous hidden state and the current input with the application of a non-linearity. In most RNN implementations, this non-linearity is a hyperbolic tangent:

$h_t = \tanh(Wx_t + Whh_{t-1} + bh)$ . -  $y_t$  is the output at time step  $t$ . We can decide how it looks like according to the task. For example, in predicting next word in a sentence, the output  $y_t$  can be a probability distribution over a vocabulary. All parameters  $Wx$ ,  $Whh$  and  $Why$  are shared among each unrolled time-step. So the network is actually performing the same computation at each time-step, but with different inputs  $x_t$ . This greatly reduces the total number of parameters in the network and avoids over-fitting on smaller datasets. Hidden state  $h_t$  is the main feature of RNNs. It works as the network memory which captures useful information about what happened in all the previous time steps. Currently, the most commonly used type of RNNs are Long Short Term Memory networks (LSTMs). LSTMs are a special kind of RNN, capable of learning long-term dependencies due to their gating mechanism. They were introduced by Hochreiter & Schmidhuber [9], and were refined and popularized by many people in the following year.

### ***Proposed System:***

Effective taxi dispatching can help both drivers and passengers to minimize the wait-time to find each other. Drivers do not have enough information about where passengers and other taxis are and intend to go. Therefore, a taxi center can organize the taxi fleet and efficiently distribute them according to the demand from the entire city. To build such a taxi center, an intelligent system that can predict

the future demand throughout the city is required. Our system uses GPS location and other properties of the taxi like drop point, pickup point etc. to predict the future demand. A convolutional neural network based model is trained with given history data. This model is used to predict the demand in different areas of the city. With recent advancements in virtual reality, augmented reality, and smart wearable devices, the need for getting the state of the art deep learning algorithms onto these resource constrained compute platforms increases. Porting state of the art deep learning algorithms to resource constrained compute platforms is extremely challenging. We introduce LCNN, a lookup-based convolutional neural network that encodes convolutions by few lookups to a dictionary that is trained to cover the space of weights in CNNs. Training LCNN involves jointly learning a dictionary and a small set of linear combinations. The size of the dictionary naturally traces a spectrum of trade-offs between efficiency and accuracy.

LCNN enables efficient inference; our experimental results on ImageNet challenge show that LCNN can offer  $3.2\times$  speedup while achieving 55.1% top-1 accuracy using AlexNet architecture. Our fastest LCNN offers  $37.6\times$  speed up over AlexNet while maintaining 44.3% top-1 accuracy. LCNN not only offers dramatic speed ups at inference, but it also enables efficient training. On-device training of deep learning methods requires algorithms that can handle few-shot and few-iteration constraints. LCNN can simply deal with these problems because our dictionaries are architecture agnostic and transferable across layers and architectures, enabling us to only learn few linear combination weights. Our future work involves exploring low-precision dictionaries as well as compact data structures for the dictionaries.

## MODULES

1. Dataset pre-processing
2. Convolutional Neural Network
3. Prediction
4. result presentation

### *Dataset pre-processing*

We like to use Python language because it has vast collection of machine learning libraries. The dataset might contain empty values, negative values or error. Dataset is cleaned in the pre-processing. The preprocessing methods involve of removing records which is not complete. Once the clean dataset is available we have to prepare it to feed to the machine learning algorithm. Data sets are not perfect. Sometimes they end up with invalid, corrupt, or missing values. For the project I was working on, I could not have any values that were null or empty. This How-To will walk you through writing a simple Python script to see if your data set has null or empty values, and if so, it will propose two options for how to modify your data.

This is the point at which we get into the part of data science that I like to call "data intuition", by which I mean "really looking at your data and trying to figure out why it is the way it is and how that will affect your analysis". It can be a frustrating part of data science, especially if you're newer to the field and don't have a lot of experience.

### **Is this value missing because it wasn't recorded or because it doesn't exist?**

If a value is missing because it doesn't exist (like the height of the oldest child of someone who doesn't have any children) then it doesn't make sense to try and guess what it might be. These values you probably do want to keep as NaN. On the other hand, if a value is missing because it wasn't recorded, then you can try to guess what it might have been based on the other values in that column and row.

Data in real world are rarely clean and homogeneous. Data can either be missing during data extraction or collection. Missing values need to be handled because they reduce quality for any of our performance metric. It can also lead to wrong prediction or classification and can also cause a high bias for any given model being used.

## ***Convolutional Neural Network***

The network input is the current taxi demand and other relevant information while the output is the demand in the next time-step. The reason we use a convolutional neural network is that it can be trained to store all the relevant information in a sequence to predict particular outcomes in the future. In addition, taxi demand prediction is a time series forecasting problem in which an intelligent sequence analysis model is required. We divide the entire city into small areas. It is desired to predict taxi demand in small areas so that the drivers know exactly where to go. We train our system with dataset and create the model for future prediction. With recent advancements in virtual reality, augmented reality, and smart wearable devices, the need for getting the state of the art deep learning algorithms onto these resource constrained compute platforms increases. Porting state of the art deep learning algorithms to resource constrained compute platforms is extremely challenging. We introduce LCNN, a lookup-based convolutional neural network that encodes convolutions by few lookups to a dictionary that is trained to cover the space of weights in CNNs. Training LCNN involves jointly learning a dictionary and a small set of linear combinations. The size of the dictionary naturally traces a spectrum of trade-offs between efficiency and accuracy.

### **ALGORITHM**

#### **LCNN:**

We introduce LCNN, a lookup-based convolutional neural network that encodes convolutions by few lookups to a dictionary that is trained to cover the space of weights in CNNs. Training LCNN involves jointly learning a dictionary and a small set of linear combinations. The size of the dictionary naturally traces a spectrum of trade-offs between efficiency and accuracy. Our experimental results using AlexNet on ImageNet challenge show that LCNN can offer  $3.2\times$  speedup while achieving 55.1% top-1 accuracy. Our fastest LCNN offers  $37.6\times$  speed up over CNN while maintaining 44.3% top-1 accuracy. In the ResNet-18, the most accurate LCNN offers  $5\times$  speedup with 62.2% accuracy and the fastest LCNN offers  $29.2\times$  speedup with 51.8% accuracy. In addition, LCNN enables efficient training; almost all the work in efficient deep learning have focused on efficient inference on resource constrained platforms. Training on these platforms is even more challenging and requires addressing two

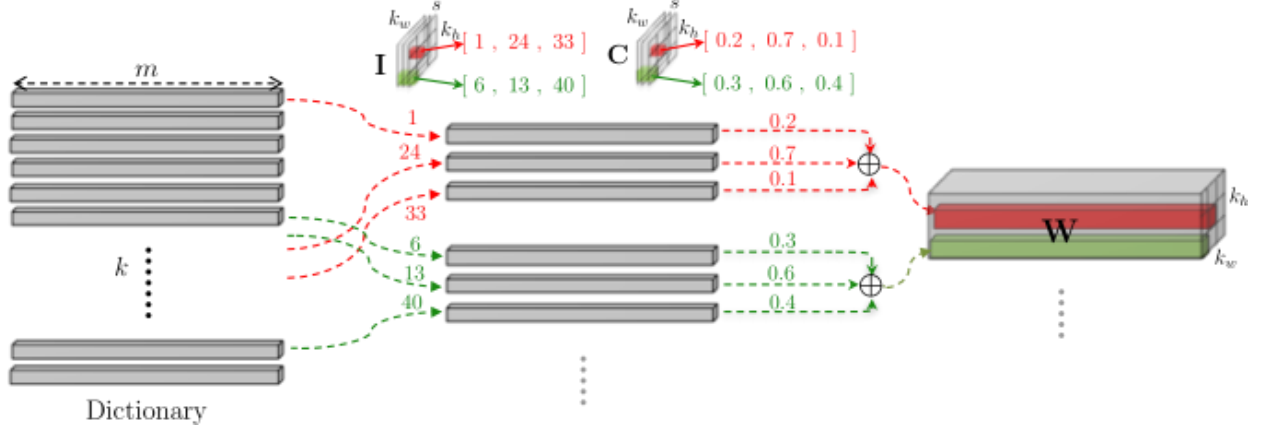
major problems: i. few-shot learning: the settings of on-device training dictates that there won't be enough training examples for new categories. In fact, most training needs to be done with very few training examples; ii. few-iteration learning: the constraints in computation and power require the training to be light and quick. This imposes hard constraints on the number of iterations in training. LCNN offers solutions for both of these problems in deep on-device training.

Few-shot learning, the problem of learning novel categories from few examples (sometimes even one example), have been extensively studied in machine learning and computer vision. The topic is, however, relatively new for deep learning, where the main challenge is to avoid over fitting. The numbers of parameters are significantly higher than what can be learned from few examples. LCNN, by virtue of having fewer parameters to learn (only around 7% of parameters of typical networks), offers a simple solution to this challenge. Our dictionary can be learned offline from training data where enough training examples per category exists. When facing new categories, all we need to learn is the set of sparse reconstruction weights. Our experimental evaluations show significant gain in few-shot learning; 6.3% in one training example per category.

Few-iteration learning is the problem of getting highest possible accuracy in few iterations that a resource constrained platform can offer. In a typical CNN, training often involves hundreds of thousands of iterations. This number is even higher for recent deeper architectures. LCNN offers a solution: dictionaries in LCNN are architecture agnostic and can be transferred across architectures or layers. This allows us to train a dictionary using a shallow network and transfer it to a deeper one. As before, all we need to learn are the few reconstruction weights; dictionaries don't need to be trained again. Our experimental evaluations on ImageNet challenge show that using LCNN we can train an 18-layer ResNet with a pre-trained dictionary from a 10-layer ResNet and achieve 16.2% higher top-1 accuracy on 10K iterations.

A convolutional layer in a CNN consists of four parts: 1) the input tensor  $X \in \mathbb{R}^{m \times w \times h}$ ; where  $m$ ,  $w$  and  $h$  are the number of input channels, the width and the height, respectively, 2) a set of  $n$  weight filters, where each filter is a tensor  $W \in \mathbb{R}^{m \times k_w \times k_h}$ , where  $k_w$  and  $k_h$  are the width and the height of the filter, 3) a scalar bias term  $b \in \mathbb{R}$  for each filter, and 4) the output

tensor  $Y \in \mathbb{R}^{n \times w \times h}$ ; where each channel  $Y[i, :, :] \in \mathbb{R}^{w \times h}$  is computed by  $W * X + b$ . Here  $*$  denotes the discrete convolution operation<sup>11</sup>The  $(:)$  notation is borrowed from NumPy for selecting all entries in a dimension.



**Figure 3.1: constructing a weight filter in LCNN.**

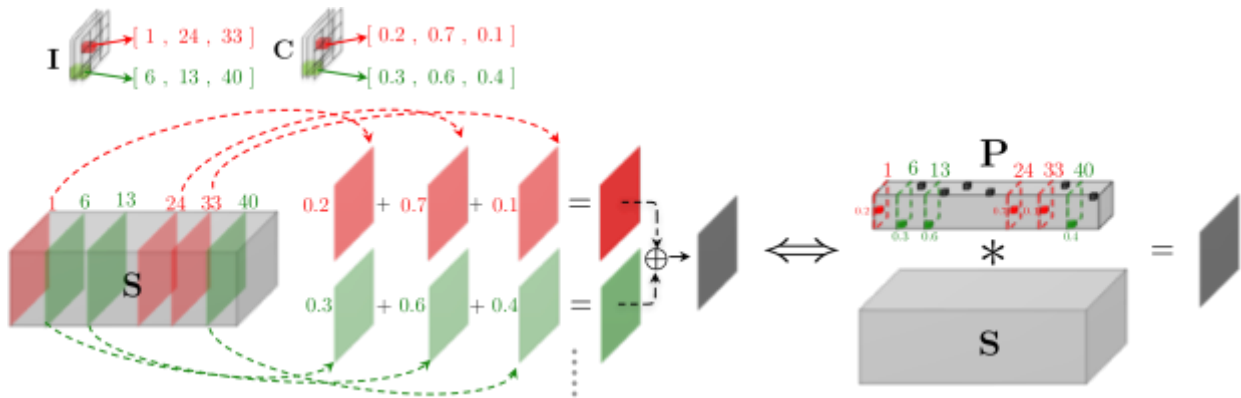
A vector in the weight filter (the long colorful cube in the gray tensor  $W$ ) is formed by a linear combination of few vectors, which are looked up from the dictionary  $D$ . Lookup indices and their coefficients are stored in tensors  $I$  and  $C$ .

For each layer, we define a matrix  $D \in \mathbb{R}^{k \times m}$  as the shared dictionary of vectors. This is illustrated in figure 3.1, on the left side. This matrix contains  $k$  row vectors of length  $m$ . The size of the dictionary,  $k$ , might vary for different layers of the network, but it should always be smaller than  $nkwkh$ , the total number of vectors in all weight filters of a layer. Along with the dictionary  $D$ , we have a tensor for lookup indices  $I \in \mathbb{N}^{s \times k_w \times k_h \leq k}$ , and a tensor for lookup coefficients  $C \in \mathbb{R}^{s \times k_w \times k_h}$  for each layer. For a pair  $(r, c)$ ,  $I[:, r, c]$  is a vector of length  $s$  whose entries are indices of the rows of the dictionary, which form the linear components of  $W[:, r, c]$ . The entries of the vector  $C[:, r, c]$  specify the linear coefficients with which the components should be combined to make  $W[:, r, c]$  (illustrated by a long colorful cube inside the gray cub in Figure 1-right). We set  $s$ , the number of components in a weight filter vector, to be a small number. The weight tensor can be constructed as follows:

$$W[:, r, c] = \sum_{t=1}^s C[t, r, c] \cdot D[I[t, r, c], :] \forall r, c \quad (1)$$

This procedure is illustrated in Figure 1. In LCNN, instead of storing the weight tensors  $W$  for

convolutional layers, we store  $D$ ,  $I$  and  $C$ , the building blocks of the weight tensors. As a result, we can reduce the number of parameters in a convolutional layer by reducing  $k$ , the dictionary size, and  $s$ , the number of components in the linear combinations. In the next section, we will discuss how LCNN uses this representation to speed up the inference.



**Figure 3.2: Look-up based convolutional neural network**

$S$  is the output of convolving the dictionary with the input tensor. The left side of this figure illustrates the inference time forward pass. The convolution between the input and a weight filter is carried out by lookups over the channels of  $S$  and a few linear combinations. Direct learning of tensors  $I$  and  $C$  reduces to an intractable discrete optimization. The right side of this figure shows an equivalent computation for training based on sparse convolutions. Parameters  $P$  can be trained using SGD. The tiny cubes in  $P$  denote the non-zero entries.

### PSEUDO CODE:

Step 1: Import all the necessary packages.

Step2: Read the csv file.

```
df=pd.read_csv('E:/Python Taxi_Full_Code/clean_data/FinalData_for_Models.csv')
```

Step 3: Use html code for layouts.

```
layout = dict(autosize=True, height=500, font=dict(color='#CCCCCC'),
titlefont=dict(color='#CCCCCC', size='14')
```

Step 4: Give the title name.

```
app.layout = html.Div([html.Div([html.H1('Estimating Taxi Demand at chennai International
Airport', className='ten columns', style = {'font-weight':'bold'}),]
```

Step 5: Adding parameters:

a. Date:

```
html.H6('Date', className='one columns'), html.Div([dcc.DatePickerSingle(id='my-date-picker-
single', min_date_allowed=dt.datetime(2014, 1, 1), max_date_allowed=dt.datetime(2020, 12,
31), initial_visible_month=dt.datetime(2017, 12, 11)
```

b. Time:

```
html.Div([html.P('Select Hour:'), dcc.Slider(id='hour', min=0, max=23, marks={i: '{:}'.format(i) for
i in range(0, 24)}, value=10),], className='five columns', style={'margin-top': '20', 'margin-
bottom': '30px', 'position':'relative'}
```

c. Precipitation:

```
html.Div(html.P('Filter by Precipitation:'), dcc.Dropdown(id='precipitation', options=[{'label':
'None', 'value': 0}, {'label': 'Light', 'value': 0.04}, {'label': 'Medium', 'value': 0.13}, {'label': 'Heavy',
'value': 0.35}],value=0
```

d. Weather:

```
html.Div([html.P('Filter by Weather Category:'), dcc.Dropdown(id='weather', options=[{'label':
'Clear', 'value': 'clear'}, {'label': 'Clouds', 'value': 'clouds'}, {'label': 'Fog', 'value': 'fog'}, {'label':
'Rain', 'value': 'rain'}, {'label': 'Snow', 'value': 'snow'}, {'label': 'Thunderstorm', 'value':
'thunderstorm'}],value=['clear'], multi=True))]
```

6. Getting the output in graph format.

```
html.Div([dcc.Graph(id='predict_graph')],className='seven columns', style={'margin-top':
'20'}),],className='row '),className='ten columns offset-by-one' )
```

LCCN enables efficient inference; our experimental results on ImageNet challenge show that LCNN can offer 3.2× speedup while achieving 55.1% top-1 accuracy using AlexNet architecture. Our fastest LCNN offers 37.6× speed up over AlexNet while maintaining 44.3% top-1 accuracy. LCNN not only offers dramatic speed ups at inference, but it also enables efficient training. On-device training of deep learning methods requires algorithms that can handle few-shot and few-iteration constraints. LCNN can simply deal with these problems because our dictionaries are architecture agnostic and transferable across layers and architectures, enabling us to only learn few linear combination weights. Our future work involves exploring low-precision dictionaries as well as compact data structures for the dictionaries.

### ***Prediction***

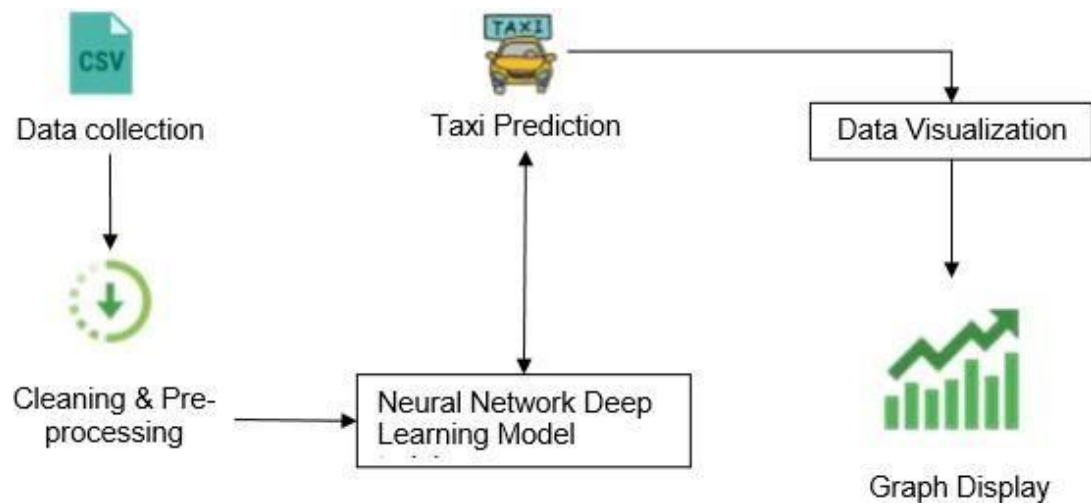
Prediction” refers to the output of an algorithm after it has been trained on a historical dataset and applied to new data when forecasting the likelihood of a particular outcome, such as whether or not a customer will churn in 30 days. The algorithm will generate probable values for an unknown variable for each record in the new data, allowing the model builder to identify what that value will most likely be.

The word “prediction” can be misleading. In some cases, it really does mean that you are predicting a future outcome, such as when you’re using machine learning to determine the next best action in a marketing campaign. Other times, though, the “prediction” has to do with, for example, whether or not a transaction that already occurred was fraudulent. In that case, the transaction already happened, but you’re making an educated guess about whether or not it was legitimate, allowing you to take the appropriate action. Machine learning model predictions allow businesses to make highly accurate guesses as to the likely outcomes of a question based on historical data, which can be about all kinds of things – customer churn likelihood, possible fraudulent activity, and more. These provide the business with insights that result in tangible business value.

## ***Result presentation***

A graph is plotted for the future prediction for the next time slot and the area to be crowded. This machine learning model predicts the future demand area in a city based on CNN and the drivers were taken to wait in the area where the system identified as demand area.

## **ARCHITECTURE DIAGRAM**



***Fig 3.3 Architecture Diagram***

**Data Collection:** Data collection is the process of gathering and measuring information on targeted variables in an established system, which then enables one to answer relevant questions and evaluate outcomes. The collected data is saved in the csv format where csv stand for comma separated values.

**Cleaning and Preprocessing: Data cleaning or cleansing** is the process of detecting and correcting (or removing) corrupt or inaccurate records from a record set, table, or database and refers to identifying incomplete, incorrect, inaccurate or irrelevant parts of the data and then replacing, modifying, or deleting the dirty or coarse data

**Data visualization:** Data visualization is the graphical representation of information and data. By using visual elements like charts, graphs, and maps, data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data

## ATTRIBUTES:

**a) Date and Time:** The date format is required to find the demand of taxi in that date at particular time. The time format is taken in 24 hour format so that we can predict the accurate demand.

**Date:**

| Date       | No of pickups |
|------------|---------------|
| 10/11/2018 | 273           |
| 5/10/2019  | 284           |
| 3/6/2020   | 225           |

**Time:**

| Time | No of pickups |
|------|---------------|
| 4    | 117           |
| 14   | 554           |
| 8    | 303           |

**b) precipitation and weather:** This is required to measure demand at certain climatic conditions. There are three categories, such as light, medium and high. Weather is classified as clear, rain, clouds, fog, snow, and thunderstorm.

**Precipitation:**

| No of Pickups | light | medium | high |
|---------------|-------|--------|------|
| 496           | yes   | -      | -    |
| 473           | -     | yes    | -    |
| 461           | -     | -      | yes  |

**Weather:**

| No of pickups | Weather      |
|---------------|--------------|
| 519           | clear        |
| 525           | clouds       |
| 523           | fog          |
| 524           | rain         |
| 518           | Snow         |
| 499           | Thunderstorm |

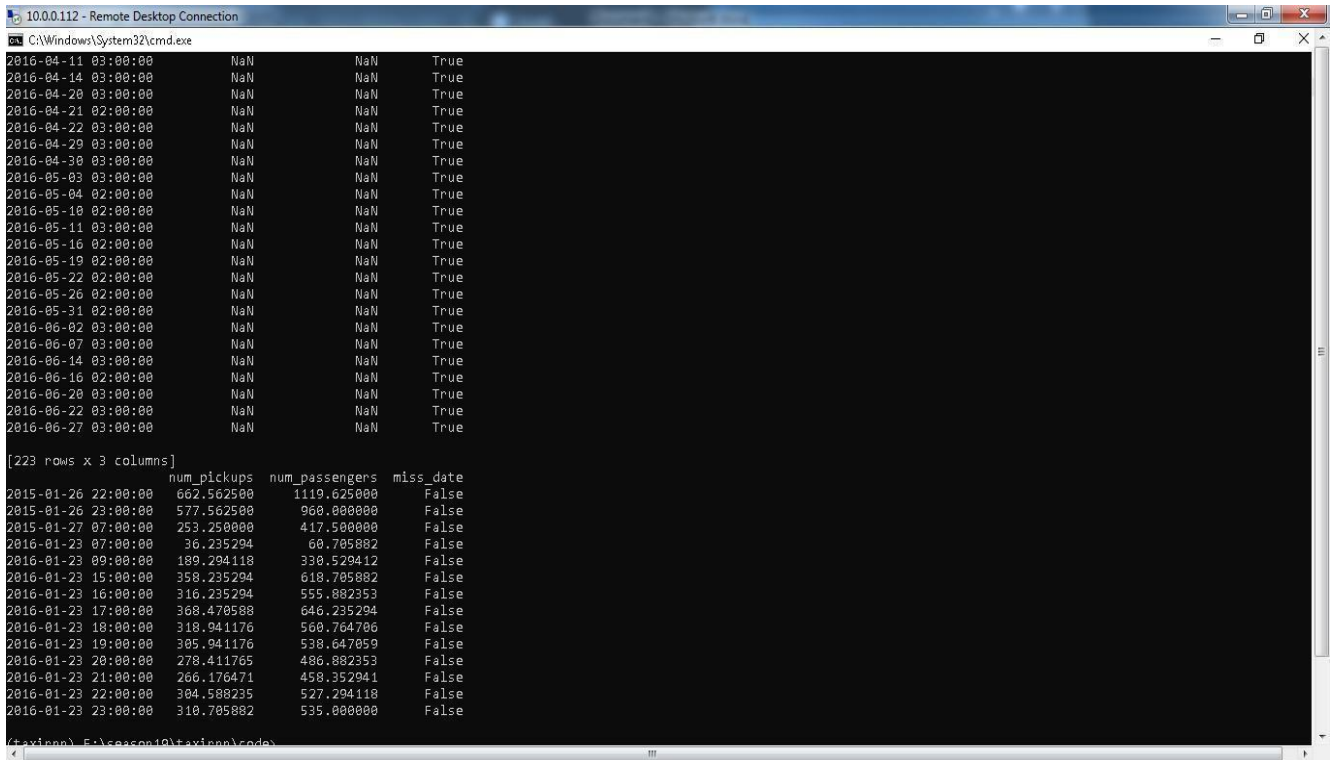
**c) Temperature:** In this project temperature is taken input as Fahrenheit (deg F). To predict the accurate demand.

**Temperature:**

| Temperature | No of pickups |
|-------------|---------------|
| 60          | 316           |
| 56          | 321           |
| 90          | 298           |

## CHAPTER 4

### RESULT AND PERFORMANCE ANALYSIS



The screenshot shows a Remote Desktop Connection window titled "10.0.0.112 - Remote Desktop Connection". The command prompt window displays the following output:

```
C:\Windows\System32\cmd.exe

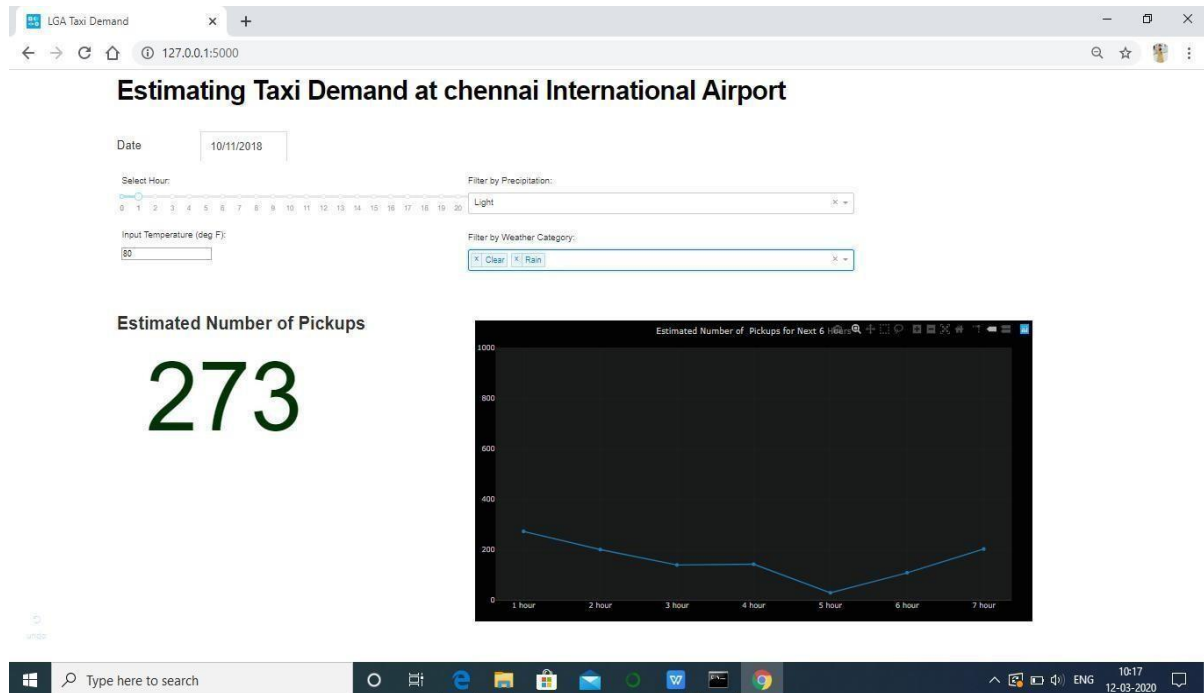
2016-04-11 03:00:00      NaN      NaN      True
2016-04-14 03:00:00      NaN      NaN      True
2016-04-20 03:00:00      NaN      NaN      True
2016-04-21 02:00:00      NaN      NaN      True
2016-04-22 03:00:00      NaN      NaN      True
2016-04-29 03:00:00      NaN      NaN      True
2016-04-30 03:00:00      NaN      NaN      True
2016-05-03 03:00:00      NaN      NaN      True
2016-05-04 02:00:00      NaN      NaN      True
2016-05-10 02:00:00      NaN      NaN      True
2016-05-11 03:00:00      NaN      NaN      True
2016-05-16 02:00:00      NaN      NaN      True
2016-05-19 02:00:00      NaN      NaN      True
2016-05-22 02:00:00      NaN      NaN      True
2016-05-26 02:00:00      NaN      NaN      True
2016-05-31 02:00:00      NaN      NaN      True
2016-06-02 03:00:00      NaN      NaN      True
2016-06-07 03:00:00      NaN      NaN      True
2016-06-14 03:00:00      NaN      NaN      True
2016-06-16 02:00:00      NaN      NaN      True
2016-06-20 03:00:00      NaN      NaN      True
2016-06-22 03:00:00      NaN      NaN      True
2016-06-27 03:00:00      NaN      NaN      True

[223 rows x 3 columns]
   num_pickups  num_passengers  miss_date
2015-01-26 22:00:00    662.562500    1119.625000    False
2015-01-26 23:00:00    577.562500    960.000000    False
2015-01-27 07:00:00    253.250000    417.500000    False
2016-01-23 07:00:00     36.235294     60.705882    False
2016-01-23 09:00:00    189.294118    330.529412    False
2016-01-23 15:00:00    358.235294    618.705882    False
2016-01-23 16:00:00    316.235294    555.802353    False
2016-01-23 17:00:00    368.470588    646.235294    False
2016-01-23 18:00:00    318.941176    560.764706    False
2016-01-23 19:00:00    305.941176    530.647059    False
2016-01-23 20:00:00    278.411765    486.802353    False
2016-01-23 21:00:00    266.176471    450.352941    False
2016-01-23 22:00:00    304.588235    527.204118    False
2016-01-23 23:00:00    310.705882    535.000000    False

(taxiapp).5>season10(taxiapp).code>
```

**Fig 4.1 Data pre-processing**

The above screenshot illustrates the replacing the null values and missing values with the mean values in the pre-processing.



**Fig: 4.2 Output of Taxi Demand prediction**

The above screenshot represents the output for predicting taxi demand. Few more parameters also exist to predict the accurate demand such as precipitation, weather, temperature at particular date and time. The output is in the graph format which predicts the next six hours of data.

## **CHAPTER 5**

### **CONCLUSION AND FUTURE WORK**

#### **CONCLUSION**

The proposed framework is a consecutive learning model with repetitive neural system for anticipating the taxi request in various regions in the city. Gaining from the previous chronicled information, the interest expectation is accomplished for the area. Three Years information of the Airport is utilized to train our model. This model gives the forecast of taxi interest for hourly premise and a specific time. This work can be reached out later on by including more info, for example, occasions, celebrations and so forth. Taxicabs can be sorted out and send dependent on the forecast of the model. What's more, it can spare such a lot of gas that is at present being spent by cabs to discover travelers.

#### **FUTURE WORK**

This approach meets no parallel in the literature due to its test bed; the models have been tested in a streaming environment, while the state of the art mainly presents offline experimental setups. This model will be used as a feature for a recommendation system, which will produce smart live recommendations to the taxi drivers about which taxi stand they should head to after a drop-off. This decision support framework will also address other features such as distance or live traffic conditions. We believe that the deployment of such a system in a taxi fleet will contribute to increasing its competitiveness facing other taxi fleets in a Scenario-1 network (e.g., such as the network that was studied, where the average waiting time to pick up a passenger at a taxi stand is three times higher than the average service duration) by improving the distribution of the vacant vehicles throughout the stands.

## REFERENCES

- [1] D. J. Zawack and G. L. Thompson, "A dynamic space-time network flow model for city traffic congestion," *Transp. Sci.*, vol. 21, no. 3, pp. 153–162, 1987.
- [2] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," in *The Handbook of Brain Theory and Neural Networks*, vol. 3361. Cambridge, MA, USA: MIT Press, no. 10, 1995, p. 1995.
- [3] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [4] J. Lee, I. Shin, and G.-L. Park, "Analysis of the passenger pick-up pattern for taxi location recommendation," in *Proc. 4th Int. Conf. Netw. Comput. Adv. Inf. Manag.*, vol. 1, 2008, pp. 199–204.
- [5] Y. Yue, Y. Zhuang, Q. Li, and Q. Mao, "Mining time-dependent attractive areas and movement patterns from taxi trajectory data," in *Proc. IEEE 17th Int. Conf. Geoinformat.*, 2009, pp. 1–6.
- [6] J. Yuan et al., "T-Drive: Driving directions based on taxi trajectories," in *Proc. 18th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, 2010, pp. 99–108.
- [7] L. Liu, C. Andris, and C. Ratti, "Uncovering cabdrivers' behavior patterns from their digital traces," *Comput. Environ. Urban Syst.*, vol. 34, no. 6, pp. 541–548, 2010.
- [8] S. Phithakkitnukoon, M. Veloso, C. Bento, A. Biderman, and C. Ratti, "Taxi-aware map: Identifying and predicting vacant taxis in the city," in *Proc. Int. Joint Conf. Ambient Intell.*, 2010, pp. 86–95.
- [9] Z. Deng and M. Ji, "Spatiotemporal structure of taxi services in Shanghai: Using exploratory spatial data analysis," in *Proc. IEEE 19th Int. Conf. Geoinformat.*, 2011, pp. 1–5.
- [10] B. Li et al., "Hunting or waiting? Discovering passenger-finding strategies from a large-scale real-world taxi dataset," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops*, 2011,

pp. 63–68.

- [11] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, “Translating embeddings for modeling multi-relational data,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2787–2795.
- [12] M. Qu, H. Zhu, J. Liu, G. Liu, and H. Xiong, “A cost-effective recommender system for taxi drivers,” in *Proc. 20th ACM SIGKDD Int. Conf.* N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie, “T-finder: A recommender system for finding passengers and vacant taxis,” *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 10, pp. 2390–2403, Oct. 2013.
- [13] K. Cho et al., “Learning phrase representations using RNN encoder decoder for statistical machine translation,” 2014, arXiv:1406.1078.
- [14] R.-H. Hwang, Y.-L. Hsueh, and Y.-T. Chen, “An effective taxi recommender system based on a spatio-temporal factor analysis model,” *Inf. Sci.*, vol. 314, pp. 28–40, 2015.
- [15] A. Canziani and E. Culurciello, “Cortexnet: A generic network family for robust visual temporal representations,” 2017, arXiv:1706.02735.
- [16] P. Chen, H. Lv, S. Gao, Q. Niu, and S. Xia, “A real-time taxicab recommendation system using big trajectories data,” *Wireless Commun. Mobile Comput.*, vol. 2017, 2017, Art. no. 5414930.
- [17] R. Pamula and R. Chakraborty, “Taxi recommender system using ridesharing service,” in *Proc. IEEE 4th Int. Conf. Adv. Comput. Commun. Syst.*, 2017, pp. 1–6.
- [18] Z. Ma, J.-H. Xue, A. Leijon, Z.-H. Tan, Z. Yang, and J. Guo, “Decorrelation of neutral vector variables: Theory and applications,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 1, pp. 129–143, Jan. 2018.

- [19] Z.Ma, Y. Lai,W. B. Kleijn, Y.-Z. Song, L.Wang, and J. Guo, "Variational Bayesian learning for Dirichlet process mixture of inverted Dirichlet distributions in non- Gaussian image feature modeling," IEEE Trans. Neural Netw. Learn. Syst., vol. 30, no. 2, pp. 449–463, Feb. 2019.
- [20] Z. Ma, H. Yu, W. Chen, and J. Guo, "Short utterance based speech language identification in intelligent vehicles with time-scale modifications and deep bottleneck features," IEEE Trans. Veh. Technol., vol. 68, no. 1, pp. 121–128, Jan. 2019.

## APPENDIX

### A. SOURCE CODE

```
import os
import pickle
import copy
import datetime as dt
from datetime import timedelta
import pandas as pd
from flask import Flask
import dash
from dash.dependencies import Input, Output, State
import dash_core_components as dcc
import dash_html_components as html
import pandas as pd
import numpy as np
from pandas.tseries.holiday import USFederalHolidayCalendar as calendar
from sklearn.externals import joblib

df = pd.read_csv('E:/Python/Taxi_Full_Code/clean_data/FinalData_for_Models.csv')
loaded_model = joblib.load("model_rfgbtxgb_la.pkl")
hourly_biases = pd.read_csv('E:/Python/Taxi_Full_Code/clean_data/biases_per_hour.csv', index_col = 0)
hourly_biases = hourly_biases.values
app = dash.Dash()
app.css.append_css({'external_url': 'https://cdn.rawgit.com/plotly/dash-app-stylesheets/2d266c578d2a6e8850ebce48fdb52759b2aef506/style-sheet-oil-and-gas.css'})
app.title = "LGA Taxi Demand"
layout = dict(
    autosize=True,
    height=500,
    font=dict(color='#CCCCCC'),
    titlefont=dict(color='#CCCCCC', size='14'),
```

```

margin=dict(
    l=35,
    r=35,
    b=35,
    t=45
),
hovermode="closest",
plot_bgcolor="#191A1A",
paper_bgcolor="#020202",
legend=dict(font=dict(size=10), orientation='h'),
title='Satellite Overview',
)

```

```

app.layout = html.Div(
    [
        html.Div(
            [
                html.H1(
                    'Estimating Taxi Demand at chennai International Airport',
                    className='ten columns', style = {'font-weight':'bold'}
                ),
            ],
            className='row', style={'margin-bottom':'20px', 'color' : '#000000'}
        ),
        html.Div(
            [

                html.H6('Date', className='one columns'),
                html.Div([dcc.DatePickerSingle(
                    id='my-date-picker-single',
                    min_date_allowed=dt.datetime(2014, 1, 1),
                    max_date_allowed=dt.datetime(2020, 12, 31),
                    initial_visible_month=dt.datetime(2017, 12, 11),

```

```

        date=dt.datetime(2017, 12, 11)
    ]], className='two columns'),
],
className='row'
),
html.Div(
    [
        html.P('Select Hour:'),
        dcc.Slider(
            id='hour',
            min=0,
            max=23,
            marks={i: '{}'.format(i) for i in range(0, 24)},
            value=10
        ),
    ], className='five columns',
    style={'margin-top': '20', 'margin-bottom': '30px', 'position':'relative'}
),
html.Div(
    [
        html.Div(
            [
                html.P('Filter by Precipitation:'),
                dcc.Dropdown(
                    id='precipitation',
                    options=[
                        {'label': 'None', 'value': 0},
                        {'label': 'Light', 'value': 0.04},
                        {'label': 'Medium', 'value': 0.13},
                        {'label': 'Heavy', 'value': 0.35}
                    ],
                    value=0
                ),
            ]
        )
    ]
)

```

```

    ],
    className='five columns', style={'margin-left':'600px', 'margin-top': '-
90px', 'display':'block'}
  ),
  html.Div(
    [
      html.P('Input Temperature (deg F):'),
      dcc.Input(id='temp', value=70, type='int'),
    ],
    className='six columns', style={'margin-bottom':'80px'}
  ),
],
className='row', style= {'margin-top': '110px'}
),
html.Div(
  [
    html.P('Filter by Weather Category:'),
    dcc.Dropdown(
      id='weather',
      options=[
        {'label': 'Clear', 'value': 'clear'},
        {'label': 'Clouds', 'value': 'clouds'},
        {'label': 'Fog', 'value': 'fog'},
        {'label': 'Rain', 'value': 'rain'},
        {'label': 'Snow', 'value': 'snow'},
        {'label': 'Thunderstorm', 'value': 'thunderstorm'},
      ],
      value=['clear'],
      multi=True
    )
  ],
  className='five columns', style={'margin-left':'600px', 'margin-top': '-
130px', 'display':'block', 'margin-bottom': '40px'}

```

```

        ),
        html.Div(
            [
                html.P('Estimated Number of Pickups', style={'font-size': '3.0rem',
'margin-bottom': '-400px', 'position': 'absolute', 'font-weight': 'bold'}),
                html.Div(id='Prediction',
                    className='four columns',
                    style={'margin-top': '20', 'font-size': '16rem', 'color': '#003406',
'align': 'center', 'margin-left': '50px'}
                ),
                html.Div(
                    [
                        dcc.Graph(id='predict_graph')
                    ],
                    className='seven columns',
                    style={'margin-top': '20'}
                ),
            ],
            className='row'
        ),
        ],
        className='ten columns offset-by-one'
    )
    @app.callback(
        dash.dependencies.Output('Prediction', 'children'),
        [dash.dependencies.Input('my-date-picker-single', 'date'),
        dash.dependencies.Input('precipitation', 'value'),
        dash.dependencies.Input('weather', 'value'),
        dash.dependencies.Input('temp', 'value'),
        dash.dependencies.Input('hour', 'value')])

```

```

def update_prediction(date, precipitation, weather, temp, hour):

```

# B.PAPER PUBLICATION

## A UNIFIED APPROACH FOR PREDICTING TAXI DEMAND USING L-CNN

[1]AVVARI LAKSHMI, [2]M.DIVYA LAKSHMI, [3]Dr.M.Maheswari

[1] [2] Student, Department of Computer Science and Engineering, Sathyabama Institute of Science & Technology

[3] Assistant Professor, Department of Computer Science and Engineering, Sathyabama Institute of Science & Technology, Chennai.

**Abstract :** The taxi administration industry is blasting for the most recent few years and it is relied upon to develop in the not so distant future. Cab drivers need to choose where to pause for travelers as they can get somebody when conceivable. Travelers likewise favor a snappy taxi administration at whatever point required. The control focal point of the taxi administration chooses the bustling territory to be concentrated. In the current framework, here and there the cabs were dispersed over the bigger territory missing the time based occupied zone like Air terminal, Business territory, school zone, Train stations and so on. Powerful taxi assignment can support the two drivers and travelers to limit the hold up time to locate one another. In the proposed framework, the future interest can be anticipated utilizing Recurrent Neural Networks based model that can be prepared with given chronicled information. It can serve more clients in a brief time frame by sorting out the accessibility of taxi. The informational collection incorporates GPS area and different properties of the taxi like slanted edge, pickup point and so forth. This model is utilized to foresee the interest for a specific time in various territories of the city.

### I. INTRODUCTION

Cab drivers need to choose where to wait tight for the travelers with the goal that they can pick somebody rapidly. So also, travelers additionally need to discover their taxis rapidly. Dispatching the taxi productively help both the clients and drivers. Compelling dispatching of taxi assists with diminishing sitting tight time for clients, also as drivers. A driver won't have enough information about where to hold up to get explorers quickly. A taxi

center can form and send the required number of cabs to the zone subject to the recorded data. The valid data uses Global Positioning System (GPS) and foresee future intrigue. In Tokyo, today this system is lessening the holding on schedule for the customers, quickly respond to the sudden contrast in solicitations and it vanquishes any deterrent between the experienced drivers and tenderfoot drivers. These favorable circumstances grant the taxi organization to achieve most prominent favorable position.

A steady taxi demand desire is proposed here besides, at this moment, data is used to foresee the future enthusiasm for taxis in a particular spot at a explicit time. A segment of the persistent goals consolidate supervising task force of taxi to swarmed zone, convincing use of advantages for decrease delaying time, server more customers in a brief time period by sifting through open taxi. Our system uses GPS region also, various properties of the taxi like pickup point, drop point, etc to anticipate taxi demand. A model is readied using a discontinuous neural framework. The discontinuous neural frameworks are used in talk affirmation programming. The discontinuous neural frameworks are used for progressive data. It is being used in Google's voice search and Apple Siri singular partners. This figuring is the achievement of significant learning in past quite a while. Discontinuous neural frameworks produce desires result for the progressive data. Taxi request expectation is a period arrangement investigation issue. It is a feed forward work neural system and the data moves starting with one system then onto the next organize and from the info layer to the yield layer through the shrouded layer. The distinction between the typical neural system and the intermittent neural organize is that in the intermittent neural system, the data pushes through the circle. In the framework, the repetitive neural system is utilized also, Python language is favored in light of the fact that it

has a huge assortment of machine learning libraries. The informational collection might contain void qualities, negative qualities or mistake.

Informational collection is cleaned in the preprocessing. The preprocessing techniques include expelling records which are not finished. When the cleaned informational collection is accessible it is set up to be sustained to the AI calculation. Intermittent Neural Networks take the past hub yield or concealed states as data sources. RNNs are helpful as their middle of the road esteems (state) can store data about past contributions for a period interim.

The fundamental element of a Recurrent Neural System (RNN) is that the system have at any rate one criticism association, so the initiations can stream round in a circle shrewd way. That empowers the systems to learn arrangements and to do worldly preparing, e.g., perform arrangement acknowledgment/multiplication or worldly affiliation/forecast. Intermittent neural system models can have numerous varied structures. One normal

The sort that comprises of a standard Multi-Layer Perceptron (MLP) in addition to included bends. These can venture the incredible non-direct mapping capacities of the MLP, and furthermore have some type of memory. Since one can consider intermittent arrangements as far as their properties as dynamical frameworks, it is expected to get some information about their security, perceptible and comparability.

## II. RELATED WORK

Fei Miao et al [1] proposed a cutting edge powerful transportation framework that detects information gathered from transportation frameworks that help in investigating the traveler requests. Forecast Methods on taxi passenger request were travel time and voyaging speed as indicated by traffic checking information have been created. In their proposed model Robotic mobility on-request frameworks that limit the quantity of re balancing excursions and best stopping frameworks that assign asset dependent on a driver's installments. These sorts of calculations mean to diminish long mile or to limit clients' holding up time have been created. In spite of the fact that advancement in hearty methodology means to limit the most pessimistic scenario cost under all conceivable irregular parameters, it brings about normal framework exhibitions. For a taxi dispatch framework, it is fundamental to address the pay between the most pessimistic scenario and the normal dispatch costs under questionable interest. Estimations show that under the vigorous dispatch system we structure, the normal request supply proportion irregularity is diminished by 31.7%, furthermore, the normal all out inactive driving separation is diminished by 10.130% or around 20 million miles altogether in one year.

Mohammad Saiedur Rahaman et al [2] characterized the neighborhood ID issue in the nearness of countless heterogeneous logical modules. It systematizes investigate as an issue of less pause time forecast for cab drivers at air terminals and research heterogeneous components identified with time, climate, flight appearances and taxi trips. Cabs are viewed as the least demanding method of transport for move between the air terminal and the city. The line chiefs ceaselessly screen the simultaneous lines identified with cabs and travelers and educate cabbies to join the travelers at the terminal when there is more popularity.

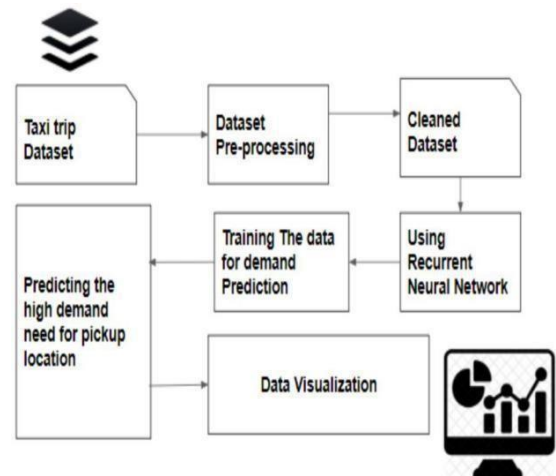
Desheng Zhang et al [3] states that the current framework of information assortment is disconnected and gathered by manual examinations and it might bring about off base information for ongoing investigation. To address this, they have utilized a model called D model, utilizing wandering taxis and utilizing them as constant versatile sensors. By executing this, they can surmise showing up traveler minutes by exploring the consistent data. They utilized 450GB data set of 14,000 taxis for a half year and it accomplishes 83% precision and beats the measurable model by 42%. Traveler request expectation might be stopped by terrible climate, extraordinary occasions or mishaps. They give two pieces of the answer for this. To start with, they mine an enormous data set of verifiable information, comprising of taxi travelers and their related pickups. Second, to address the continuous elements, they think about a great many taxis and use them as constant portable sensors. It is conceivable since they use GPS information in thick urban zones. The front-end framework and the back-end dispatching focus structure a system called meandering sensor organize. The utilized D model watches shrouded settings to derive request dependent on authentic and constant taxi information. D model investigations both the disconnected examination of information what's more, ongoing information gathered from meandering taxicabs or sensor arrange framed by them.

Luis Damas et al [4] proposed a novel strategy for anticipating the spatial appropriation of taxi-travelers for a brief time frame skyline utilizing spilling input information. To start with, the data was gathered into a histogram time-arrangement. At that point, three timearrangement guaging strategies were consolidated to rise a forecast. Exploratory tests were led utilizing the online informational collection that is transmitted by 441 vehicles of an armada running in the city of Porto, Portugal. The results explained with the goal that the structure proposed can give compelling learning into the spatio-fleeting conveyance of taxi-traveler interest for a skyline of 30 minutes. This paper centers around the continuous decision issue of which is the best taxi administration remain to go to after a traveler drop-off

(i.e., the stand where another traveler can be gotten inside a short length of time). A savvy approach with respect to this defect will improve organize unwavering quality for both organizations and clients; a smart appropriation of vehicles all through stands will limit the normal holding up time to get a traveler, while the separation voyaged will be progressively beneficial. Besides, at whatever point they need a taxi, travelers will likewise encounter a lesser holding up time to get a empty taxi. The significant commitment of this paper confronting this is to assemble expectations on the spatio-worldly dissemination of the taxi-traveler request utilizing just gushing information. Accordingly, the model that is displayed has had the option to foresee the taxi-traveler request at every last one of the 64 taxi represents 30 minutes time span interims. The model has exhibited a more than agreeable execution, effectively foreseeing the 506 874 tried administrations with an collected blunder estimation

### III. System architecture

The genuine word taxi-trip informational index is gathered. The gathered information is pre-handled and the information preprocessing is an information mining method that includes changing crude information into a reasonable arrangement. True information is frequently fragmented, conflicting, or potentially ailing in specific practices or inclines, and is liable to contain numerous mistakes. The significant assignments of the information preprocessing includes information cleaning, information reconciliation, information change, information decrease and so on. The missing qualities are distinguished and it is supplanted with mean qualities. The cleaned dataset is bolstered to the repetitive neural system model and is it prepared utilizing authentic information which comprises of date, time, pickup area, climate and so on and the future interest for the taxi is anticipated. Repetitive Neural Networks (RNN) are a amazing and strong kind of neural systems and have a place with the most encouraging calculations out there at the minute since they are the main ones with an interior memory. The anticipated interest for the taxi is imagined utilizing a chart.



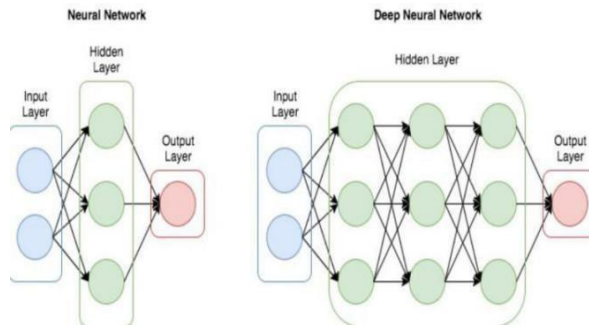
### IV. System implementation

The dataset comprises of three sections and 30,426 sections. The base file esteem and most extreme file esteems are found. The most extreme and least records are utilized to design the dataset. The yield comprises of a diagram comprising of missing qualities in the dataset relating to the given information with time in a one-hour interim in x-pivot and number of pickups in the y-pivot. It additionally comprises of a chart comprising of appointments with time in a onehour interim in x-pivot and number of pickups in the yhub. Once, missing qualities are discovered, the mean an incentive for the sections is found and this worth is supplanted in the spot of missing passages. There are a great deal of choices for supplanting missing qualities however the least complex and the most regular one is supplanting them with the mean of all the passages under the individual segment. Toward the finish of the information preprocessing, a cleaned informational collection with no absent values is found and all the missing qualities in the information set were supplanted by the mean of the particular sections. The cleaned dataset is significant in light of the fact that there might be some insignificant information in the dataset that may cause expectation mistake and causes off base results. The Comma Separated document (CSV) is gathered from the use of the client and perform cleaning and handling, because of a cleaned informational index is accomplished. The cleaned informational index is utilized for the following stage in anticipating the taxi request the repetitive neural organize is applied to anticipate the future taxi request.

#### A. LCNN

We are presenting LCNN, a lookup-based, convolutionary neural network that encodes convolutions to a dictionary trained by just a few lookups to cover the weight space in CNNs. LCNN training involves shared development of a dictionary, and a small set of linear

combinations. Of course the size of the dictionary follows a range of trade-offs between usefulness and accuracy. Use of our experimental material.



LCNN is a fast, compact but accurate model for convolutionary neural networks that enables efficient inference and training. LCNN training involves the combined learning of a vector dictionary and a selection of linear combinations. LCNN is not a model architecture, but any CNN architecture can be transformed to LCNN by replacing all convolutionary and dense layers to lookup-based layers.

#### Date:

| Date       | No of pickups |
|------------|---------------|
| 10/11/2018 | 273           |
| 5/10/2019  | 284           |
| 3/6/2020   | 225           |

#### Time:

| Time | No of pickups |
|------|---------------|
| 4    | 117           |
| 14   | 554           |
| 8    | 303           |

#### Temperature:

| Temperature | No of pickups |
|-------------|---------------|
| 60          | 316           |
| 56          | 321           |
| 90          | 298           |

#### Precipitation:

| No of Pickups | light | medium | High |
|---------------|-------|--------|------|
| 496           | yes   | -      | -    |
| 473           | -     | yes    | -    |
| 461           | -     | -      | yes  |

#### Weather:

| No of pickups | Weather      |
|---------------|--------------|
| 519           | clear        |
| 525           | clouds       |
| 523           | fog          |
| 524           | rain         |
| 518           | Snow         |
| 499           | Thunderstorm |

## V. Conclusion

The proposed framework is a consecutive learning model with repetitive neural system for anticipating the taxi request in various regions in the city. Gaining from the previous chronicled information, the interest expectation is accomplished for the area. Three Years information of the Airport is utilized to train our model. This model gives the forecast of taxi interest for hourly premise and a specific time.

This work can be reached out later on by including more info, for example, occasions, celebrations and so forth. Taxicabs can be sorted out and send dependent on the forecast of the model. What's more, it can spare such a lot of gas that is at present being spent by cabs to discover travelers.

## REFERENCES

- [1] D. J. Zawack and G. L. Thompson, "A dynamic space-time network flow model for city traffic congestion," *Transp. Sci.*, vol. 21, no. 3, pp. 153–162, 1987.
- [2] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," in *The Handbook of Brain Theory and Neural Networks*, vol. 3361. Cambridge, MA, USA: MIT Press, no. 10, 1995, p. 1995.
- [3] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [4] J. Lee, I. Shin, and G.-L. Park, "Analysis of the passenger pick-up pattern for taxi location recommendation," in *Proc. 4th Int. Conf. Netw. Comput. Adv. Inf. Manag.*, vol. 1, 2008, pp. 199–204.

- [5] Y. Yue, Y. Zhuang, Q. Li, and Q. Mao, "Mining time-dependent attractive areas and movement patterns from taxi trajectory data," in Proc. IEEE 17th Int. Conf. Geoinformat., 2009, pp. 1–6.
- [6] J. Yuan et al., "T-Drive: Driving directions based on taxi trajectories," in Proc. 18th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst., 2010, pp. 99–108.
- [7] L. Liu, C. Andris, and C. Ratti, "Uncovering cabdrivers' behavior patterns from their digital traces," Comput. Environ. Urban Syst., vol. 34, no. 6, pp. 541–548, 2010.
- [8] S. Phithakkitnukoon, M. Veloso, C. Bento, A. Biderman, and C. Ratti, "Taxi-aware map: Identifying and predicting vacant taxis in the city," in Proc. Int. Joint Conf. Ambient Intell., 2010, pp. 86–95.
- [9] Z. Deng and M. Ji, "Spatiotemporal structure of taxi services in Shanghai: Using exploratory spatial data analysis," in Proc. IEEE 19th Int. Conf. Geoinformat., 2011, pp. 1–5.
- [10] B. Li et al., "Hunting or waiting? Discovering passenger-finding strategies from a large-scale real-world taxi dataset," in Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops, 2011, pp. 63–68.
- [11] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in Proc. Adv. Neural Inf. Process. Syst., 2013, pp. 2787–2795.
- [12] M. Qu, H. Zhu, J. Liu, G. Liu, and H. Xiong, "A cost-effective recommender system for taxi drivers," in Proc. 20th ACM SIGKDD Int. Conf. N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie, "T-finder: A recommender system for finding passengers and vacant taxis," IEEE Trans. Knowl. Data Eng., vol. 25, no. 10, pp. 2390–2403, Oct. 2013.
- [13] K. Cho et al., "Learning phrase representations using RNN encoder decoder for statistical machine translation," 2014, arXiv:1406.1078.
- [14] R.-H. Hwang, Y.-L. Hsueh, and Y.-T. Chen, "An effective taxi recommender system based on a spatio-temporal factor analysis model," Inf. Sci., vol. 314, pp. 28–40, 2015.
- [15] A. Canziani and E. Culurciello, "Cortexnet: A generic network family for robust visual temporal representations," 2017, arXiv:1706.02735.
- [16] P. Chen, H. Lv, S. Gao, Q. Niu, and S. Xia, "A real-time taxicab recommendation system using big trajectories data," Wireless Commun. Mobile Comput., vol. 2017, 2017, Art. no. 5414930.
- [17] R. Pamula and R. Chakraborty, "Taxi recommender system using ridesharing service," in Proc. IEEE 4th Int. Conf. Adv. Comput. Commun. Syst., 2017, pp. 1–6.
- [18] Z. Ma, J.-H. Xue, A. Leijon, Z.-H. Tan, Z. Yang, and J. Guo, "Decorrelation of neutral vector variables: Theory and applications," IEEE Trans. Neural Netw. Learn. Syst., vol. 29, no. 1, pp. 129–143, Jan. 2018.
- [19] Z. Ma, Y. Lai, W. B. Kleijn, Y.-Z. Song, L. Wang, and J. Guo, "Variational Bayesian learning for Dirichlet process mixture of inverted Dirichlet distributions in non-Gaussian image feature modeling," IEEE Trans. Neural Netw. Learn. Syst., vol. 30, no. 2, pp. 449–463, Feb. 2019.
- [20] Z. Ma, H. Yu, W. Chen, and J. Guo, "Short utterance based speech language identification in intelligent vehicles with time-scale modifications and deep bottleneck features," IEEE Trans. Veh. Technol., vol. 68, no. 1, pp. 121–128, Jan. 2019.