Journal of Critical Reviews

ISSN- 2394-5125 Vol 7, Issue 8, 2020

A USER FRIENDLY NAVIGATION SYSTEM FOR PACING URBANIZATION

K. Vijayakumar[1], S. Pushparaj[2], J. Pradeep[3], V. Rohith[4]

[1,2]Assistant Professor, [3]Associate Professor, [4]UG Student
[1,2,3,4]Department of Electronics and Communication Engineering,
Sri Manakula Vinayagar Engineering College, Puducherry, India
Email: [3]pradeepj@smvec.ac.in

Received: 18.05.2020 *Revised: 15.06.2020* *Accepted: 04.07.2020*

Abstract
Navigation through online map has become an immutable one for various people ranging from travelers to any peer to peer ride sharing services. The dynamic urbanization makes the existing navigation applications to stand behind the need of the user. The gap identified can be reduced by regularly updating the map. The paper presents a method to update the map based on features extracted from the images which were uploaded to the cloud server by the users. The four features extracted from the images are category of place, written content, latitude and longitude. The CNN model is used for prediction and the predicted image is plotted in the map using latitude and longitude as X axis and Y axis. A cloud server is used to store the uploaded images, run the prediction model and update the single map which can be accessed by all the users. Google drive, jupyter notebook and google spreadsheet were used in the prototype for storage, feature extraction and mapping process of the images.

Keywords--Machine learning; classes365 CNN model; Optical Character Recognition; Jupyter notebook; Google spreadsheet

DOI: http://dx.doi.org/10.31838/jcr.07.08.249

INTRODUCTION

Cities and towns are prone to changes drastically because of increased populous. Urbanization leads to construction or relocation of many shops, malls, theatres, hospitals and religious places. This newly and relocated buildings made the travelling difficult for travellers, door delivery services and many taxi firms. This gap is bridged by offline and online location service provides. This map has to be regularly updated for the ease of populous. The proper updated map helps the application users to reach the destination quicker. The map update will be so dependable if it is updated by the effective people. This paper explains about a prototype to update the map by the trespassers, delivery service persons and ride service drivers.

Many taxi and bike firms are involved in online ride sharing services. The drivers of those vehicles are solely depending only on the navigation device which is connected to the cloud server. The map which they are using will be optimum if the source of map was gathered by the crowd source. If the map is updated by the crowd source then the dataset collected and processed will be in huge numbers. Machine learning is used to reduce the complexity in processing these dataset.

A manifest report in 2008 stated that over 75% of smartphone owners using navigation applications regularly. Also it stated over 36% uses this application prior to the travel and 34% in en route. About 87% use this application for driving direction the most.

The prototype presented here is a novel method which collects the image from the user of application whereas the other methods use the text from the user to update the map. The uploaded image consists of place where the picture is captured as well as the location information because of the GPS module. Global Positioning System embeds the location of picture captured into the picture itself. The latitude and longitude information along with the time stamp is embedded inside the picture which can be extracted using gps photo in python programming. The extracted features are stored in spreadsheet for further processing of the map. Computer vision is needed to automate the process of feature extraction from the uploaded image.

The uploaded images are convoluted by R-CNN model. This model classifies the image by segmenting the image into 2000 sub-segments. Then sub-segments are classified and combined to form a larger segment. RCNN retrains the pre-trained convolutional neural network based on number of classes to be detected in the image. All the regions are reshaped as per input CNN and then every region is processed by the convolutional neural network. This extracts the features of the regions and this data is used to classify the regions into different classes. Since the map updating is not an immediate turnaround process, the time delay of 40 to 50 seconds taken for processing is not the issue with this model.

Optical Character Recognition falling under the computer vision task is a useful model to extract the characters from the image by various attributes. The attributes like text density, structure of text, fonts, character type, location and artifacts are being used to extract the character from images. License plates, street signs are some of the examples of application areas of the optical character recognition module in the machine learning. Torchvision is used to import huge dataset and process it with its library functions without any hassle

LITERATURE SURVEY

Mohammad Nazmul Hasan and Md. Sharif Hossen [1] designed and developed a real-time bus tracking system using GPS tracking technology which needs only a smartphone and a real-time server. Their application consists of two fundamental concepts: first it collects the real-time location information of buses via GPS technology and secondly updates the location information in the database server. The bus-side, server-side and client-side modules provide all the expected functions. Since this application does not need any external hardware except a smartphone which is available to anyone in the world, the overall cost is very low or no cost needed for tracking the bus location. It provides nearly accurate data in real time that makes possible for the user to track the buses.

Heba Aly, Anas Basalamah, and Moustafa Youssef [2] presented Map++: a system for automatically enriching digital maps via a crowd-sensing approach based on standard cell phones. They presented the Map++ architecture as well as the features and classifiers that can accurately detects the different road features such as tunnels, bridges, crosswalks, stairs, and footbridges from the user traces. They implemented their system using commodity mobile phones from different manufacturers running the Android operating system and evaluated it in multiple different cities. Their results show that they can detect the map semantics accurately with 4% false positive and 8% false negative rates for in vehicle traces and 3% false positive and 4% false negative rates for pedestrian traces. In addition, Map++ has a significantly lower energy profile compared to systems that are based on GPS.

Chu Cao, Zhidan Liu, Mo Li, Wenqiang Wang and Zheng Qin [3] presented VitalAlley for discovering and verifying the missing walkways from the NSE mobility data. VitalAlley proposes an ellipse model and a novel weighting scheme to estimate walkable areas, and identify representative walkways from each walkable area through a two-phase clustering method. The GSV is also involved to automatically verify the new-found walkways. Experimental results using the large-scale NSE mobility data demonstrate the performances of VitalAlley that finds 736 walkways (totaling 161 km in distance) for the OSM Singapore road map.

Guan-Yu Chen, Chin-Chu Liu and Cheng-Chung Yao [4] developed a forecast system for early tsunami warning .It is built by integrating the tsunami elevation, the propagation model that calculates the offshore sea surface, and the near shore module that interpolates the inundated distance to the shoreline. This system is implemented by a Matlab graphical user interface (GUI). In this interface system, the rupture arguments such as epicenter location, focal depth, and fault angles are used to compute the initial tsunami wave elevation. These arguments can be retrieved from the website of the Geological Survey department. For nonlinear coastal elevation, a bell-shaped waveform is used to generate the perdiction map, which is then transformed into a Google map format and demonstrated via this GPS system.

He Li and Lai Zhijian [5] developed a mobile navigation system realizing data query and local positon. This system can apply to the popular java phones. The test results on Nokia shows that the navigation system solution is valid and feasible. It helps to provide a fesible alternative to android based navigation systems.

Haoyu Wang, Mohammad Hadian, and Xiaohui Liang [6] proposed an efficient privacy-preserving roadmap data update scheme for AVs to efficiently update the necessary ORD from the server. In this scheme, they define the concept of privacy sensitivity for each road segment and based on this concept, minimize the communication overhead between the AV and the server while preserving the privacy sensitivity of the target segments. Specifically, the privacy sensitivity has three properties, i.e. hotspot sensitivity, proximity sensitivity and route sensitivity. An efficient algorithm based on k-anonymity has been employed to choose the final update set. Their experimental results confirm the relationship between the calculated anonymization parameter k and the trip sensitivity. The results also confirm compliance of our scheme with kanonymity requirements. In our future work, we will explore more privacy-sensitivity features to be added to improve our scheme

Yanmin Zhu, Yin Wang, George Forman and Hong Wei [7] shows the feasibility of a prototype data mining application that detects missing intersections, turn restrictions and road closures, in order to automate the updating of the digital map, which has become a cornerstone resource for so many applications. While in our research we used historic data from a fleet of over 10000 taxis, we envision future streaming implementations that scavenge the data by products of a variety of fleet management deployments as well as mobile smartphones, given sufficiently anonymized, delayed and/or opt-in data sources. With ample volume and variety of available GPS streams, changes to the map can be detected and corrected in ever shorter time scales than demonstrated here. Rather than detecting a closure in 2 weeks, future systems should be able to use these data mining techniques to detect changes in real-time: minutes for busy roads and perhaps hours or days for seldom used roads. By having the most current maps, navigation aids can avoid directing drivers on circuitous routes and can avoid directing drivers to make illegal turns.

Suo Li, Haipeng Wang and Feng Xu [8] proposed an automatic framework that translates the optical image into land cover is proposed based on FCN, which achieves a reasonable result on Google Earth's image and NLCD 2011 dataset. The model's overall test accuracy is 70.8% for 8 classes. There are three primary applications for this work. Firstly, the model can obtain the general land cover from optical images, which indicates that the timely land cover can be obtained by feeding our model with the recent collected optical images, which can fill the time gap between the long update cycles of the land cover database. Secondly, this model can also be used to produce the land cover in areas lacking of data. By feeding this framework the optical image observed in different time, the difference between predictions indicates the change of the land cover. Thirdly, the processing speed is very fast, 100 square kilometers land cover can be generated within 100s. The proposed method can also be used for a number of other applications, e.g. land utilizing and forest monitoring. Furthermore, it could be used in GIS simulation by generating the modeling environment. Further developments will explore to integrate the multi-source land cover data and other GIS data such as CLC30 and open street map to improve the performance of the model.

Briana M. Sullivan [9] has successfully demonstrated how to integrate the NOAA nautical charts, the NOAA chart catalog, critical chart corrections from the LNM and the Coast Pilot in a georeferenced web-based environment. He proposed a project that offers integration of NOAA chart catalogs and NOAA nautical charts, the ability to filter charts depending on viewport, critical chart corrections from OCS as a dynamic/interactive table.

Chi-Chun Pan Prasenjit Mitrat [10] presented an architecture and an implementation of a system that can be used to automatically extract information from vast amounts of textual data and present the extracted information visually to end-users. They used FEMA National Situation Updates as their test bed to demonstrate the usefulness of their system. Their system, FEMARepViz utilizes an entity-relationship extractor, FactXtractor, to extract named entities and links entities with syntactic and semantic relations. Named entity graphs can be created to visualize entity relations. Furthermore, they use GeoTagger to resolve geographical ambiguity and create geo-temporal visualizations with FEMARepViz and Google Earth. Since there is no computational approach that can achieve human-level accuracy for complex information extraction tasks, visualization could be a solution to bridge the gap between fully automated extraction and search systems and manual data processing.

Thunyasit Pholprasit, Suporn Pongnumkul, Chalermpol Saiprasert, Sarinthon Mangkorn-ngam and Lalida Jaritsup Dhurakij [11] proposed Live Bus Track: a small scale real-time high frequency location update bus tracking system. Based on the use of smartphones as tracking device, the proposed approach has the ability to track vehicles automatically in real-time through our tailored made application. Studies conducted to evaluate the proposed model shows that results have shown that our system is power efficient, uses reasonable amount of data

transmission, provides reliable location updates, were understood by the users and satisfied our participants. All of these results suggest that our system is feasible to be implemented in a small scale vehicle tracking service that requires high frequency location update.

Yuuichi Teranishi, Junzo Kamahara and Shinji Shimojo [12] proposed a new map-based collaboration environment 'MapWiki' for ubiquitous content distribution. They have implemented a MapWiki system on Google Maps called 'GMapWiki' and evaluated its feasibility and usefulness. This environment can be used for a user interface of the word-of-mouth content authoring tool, a visualization tool of the sensor informations, a browser of a group status, and so on. GMapWiki is designed and implemented for practical use. However, in the current system architecture, there is a possibility of system overload caused by access concentration since all contents are held on one server

Hiromitsu Sugawara, Nobuyoshi Sato, Tsuyoshi Takayama and Yoshitoshi Murata [13] proposed an algorithm to estimate the number of lanes based on many GPS trajectories in their proposing Rapid Road Map Survey System Using GPS Trajectories as Collective Intelligence. The proposed algorithm can distinguish the number of lanes for a direction.

R. Immanuel Rajkumar, P.E. Sankaranarayanan and G. Sundari [14] proposed a system that uses Arduino which would provide a solution for a big organization like Indian railways to enable monitoring train's position in one place by using internet. The locations of all individual vehicles are mapped using their name and MAC address, in the Google map wherein each point in the Google map plotted provides the current GPS information about the trains.

Mark Smith [15] proposed a map based project is to integrate all the sources of traffic communication channel together in one place and decipher all the recent incident data so the results are cross verified a number of times thus reducing the number of false reports and incidents. This project is implemented for apple devices using C-programming, allows the user to view traffic reports for large scale cities with the capabilities for the addition of many more locations.

EXISTING SYSTEM

The map provided to the people has to be precise because the tendency of people to rely on the smartphone for navigation is increasing as steady rate. Also the imprecise map will have a definite economic penalty.

The popular navigation applications provide the access to users to update the location in map in a traditional way of entering text of the place along with the GPS information. Once the user uploaded text to the cloud server, the text is flagged at the map using the GPS location gathered from the smartphone. Map creators are usually provided with three options.

(i) Add or edit places with photos and opening hours
(ii) Add or update addresses and pinpoint their exact location
(iii) Add or update roads, cycle paths and walkways

From the above mentioned availabilities of users to update the map, the idea of updating the roads, cycle or walkways are automated by crowdatlas application. The crowdatlas consider the existing map available from the server and embeds the new road inference by the map matching process.

The functionality of this mapping is explained in either the matched segments phase or the unmatched segments phase as shown in the Fig.1 and Fig.2. The map output depends on the geometric refinement along with the on road traffic density and opening or closing of paths.

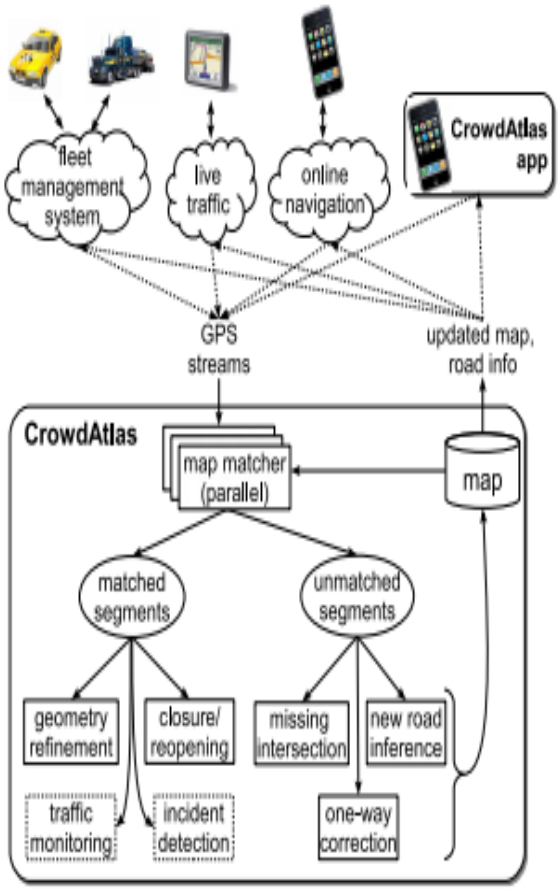

Figure 1. CrowdAtlas Block Diagram

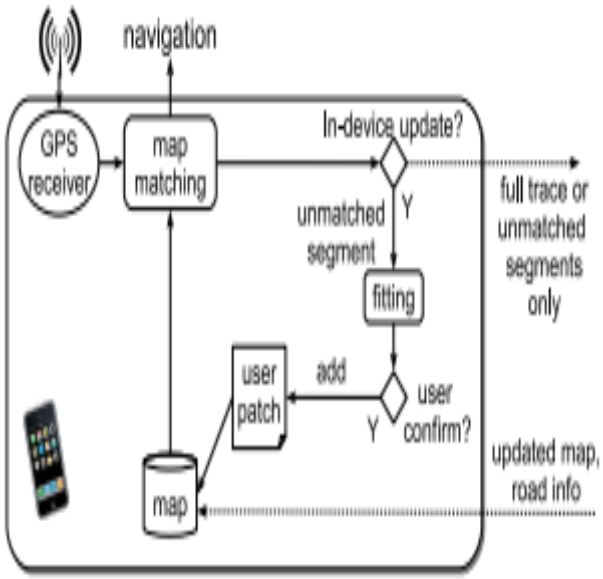

Figure 2. CrowdAtlas Flow Chart

It was created by collecting data from 80 taxis for 8 days at Beijing city. The sampling rate for interval is taken as 10 seconds as shown in figure 3. The extracted data is compared with the existing map to find out the unmatched segments after the map matching seconds as shown in Fig.3

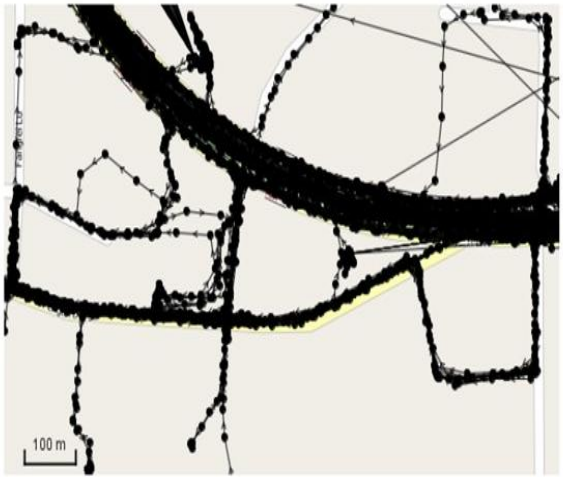

Figure 3. Data collected from 80 taxis

Figure 4. Mapping unmatched segments in the existing map

By matching the data over a week, the new clusters were found out in the map, which are different from the available route. Those new clusters are considered for the new road interference to be updated in the map as shown in Fig.4

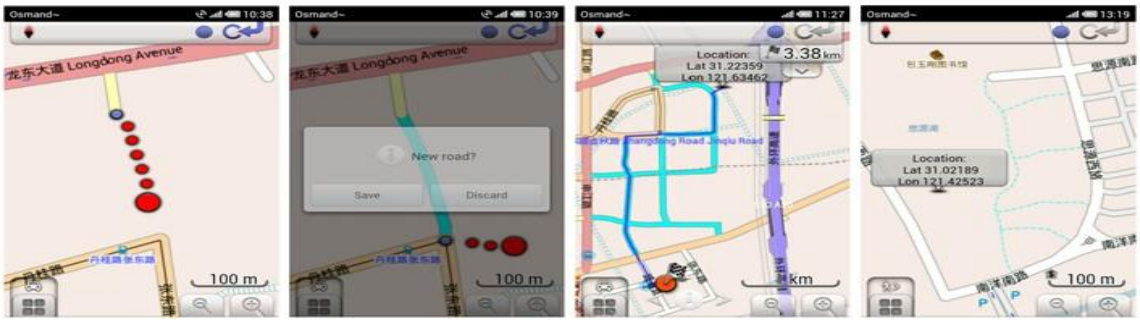

Figure 5. Implementation of CrowdAtlas map

The new road instance is created by the map matching of road, GPS sample and the candidate match. This is termed as clustering based map interpretation algorithm. In this the distance between two unmatched segments are taken as handoff and it is used to form the cluster by combining the newly unmatched segment with the existing segment within the radius of handoff. The cluster is formed by the center line fitting. The cluster formed in this way is considered as a new road and it is made to join with the existing anchor road. This process is continued until no more unmatched segment is mined. This method updates the new road but collection of data for updating the map is tedious process.

PROPOSED SYSTEM

The proposed system describes about the way of updating the location in the map by uploading an image to the server. The machine learning is used to predict the category of image and map it on plane using GPS information. The components needed for the system and its process flow is explained as follow.

The proposed system has a smooth web application interface. The web application is a compound of four major components as listed as Frontend, Backend, Database and Storage.

1. Frontend

The frontend is synonymous with UI (User Interface) but it's not so in case of this update model. The only part of the frontend architecture is the edge devices like smartphone, or cameras used to crowd source the images. GPS needs to be active at all times on smartphones for EXIF data to be generated along with captured images. Frontend is the only part available for interaction with the user aiding in collection of update datasets. None of the other components come in contact the users keeping its minimalistic interaction design intact.

2. Backend

Backend specifies the abstract process taking place in the background which processes the user input/data and results in output. This output is stored in database. Jupyter Notebook that is a python interpreter is used to process the images and generate the necessary update data. The jupyter notebook is hosted on Google Colab that provides compute as well as temporary storage for importing or installing python modules. Although the prototype needs a third person to run the predesigned code against data but it can be automated with further improvements.

3. Database

Database is where the processed data/output is stored. Now although we could use a dynamic and complex database, we go with a simple excel sheet as excel is synonymous with advanced Artificial intelligence specifically machine learning, deep leaning and data science fields. Excel sheets are used to store the update data generated by the jupyter notebook. The pandas module is used to read the excel sheets in the jupyter notebook if the need arises. Millions of rows of data can be processed in a very short time.

4. Storage

Storage is used to store input and in some cases the output. In our case the input is unprocessed image file and the output is the excel sheet containing processed data. A web based model must have its storage as a cloud based model that can be accessed from anywhere. Google drive acts a perfect storage solution with a free storage cap enough for all our needs. The Google drive is used to store the model file which is loaded to the jupyter notebook while making the classification once per session or the drive can be directly accessed by verifying credentials and allowing file stream access. Also the images can be stored in

drive and predictions can be done in cluster rather than one image per session. All these parts together make a successfully automated location services architecture. Each section is optimized for efficiency compared to its predecessor. Also extra care is taken to extremely minimize the interaction between edge device users and the services themselves. The edge device is a smartphone used by the user for navigation purpose. The application needs the GPS enabled settings in the edge device for the server to mapping process. The model is built on fundamental crowd sourcing GPS tagged images from users. The images are temporarily stored in Google drive when can be retrieved whenever needed. These images are processed to extract the needed info using a python interpreter powered by jupyter cloud notebook. The output information is stored in excel sheet, which can be later uploaded to My-Maps to update Google maps. The block diagram and flow chart for the proposed model are depicted below in Fig.6 and Fig.7.

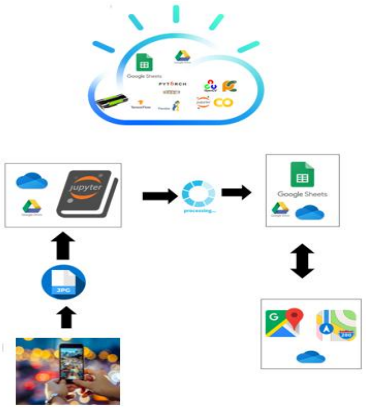

Figure 6. Block diagram of Proposed mapping model

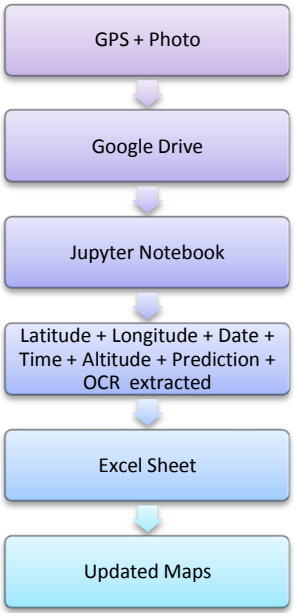

Figure 7. Flow chart of Proposed mapping model

Every block of the flowchart is explained below with its corresponding result produced by the simulation.

A. Edge Devices for Photo Capture

For edge devices to generate EXIF data encrypted with the picture files the first and foremost thing that needs to be active is the device's GPS sensor. Older handsets depend on the SIM card to detect the location .But in the newer handsets the option to activate GPS is the user's choice. Hence if we need a dataset, it is essential that the GPS be active for generation of the encrypted EXIF data which is the foundation of this update model. The normal camera application can be used to take pictures. In fact the generation of valid EXIF data was tested with the newly emerging smartphone brands such as OPPO, Redmi, One Plus and Samsung each of which yielded positive results with nearly the same accuracy. Once the image is taken the duty of the edge devices is done.

B. Google Drive for Storage

It is not always feasible to launch an update model for every image individually. Hence it is more efficient to collect the images in cluster and run the model on this cluster which boosts efficiency with a minor increase in time interval which is linear with the total load. Here's where the Google drive comes into action. The Google drive provides seamlessly fast access to the data in the cloud the user. Even the data for the model is stored on the Google drive for seamless superfast access. We only need to authorize the drive access once per account. Once the authorization is set up we are good to go for all further transactions.

C. Jupyter notebook for python Interpreter

Google Colab is free for all cloud service with few restrictions in place. But it is more than sufficient for the prototype. We use Google Colab for storage and processing capacity to run a python interpreter as well as host a temporary session for cluster processing. The jupyter notebook run inside the virtual environment of Google Colab.

D. Data Extraction and Visualization Modules

The final three blocks of the flowchart performs image prediction, data extraction and data visualization process. The jupyter notebooks needs to be set up before processing any data. It is to be remembered that each type of information that needs to be extracted needs its own unique initialization, modules and variables and most important part –the code. It is to be noted that the language used for development is python programming language.

1) Prediction Module
2) Date and Time Module
3) GPS Module
4) OCR Module
5) Video to Images Module
6) Spreadsheet Module
7) Data Visualization Module(Non Pictorial)

1) Prediction Module

To run the prediction module it is necessary to import the model file and the input data to temporary session storage of Google Colab. Modules imported for running the prediction against the input include Torch and TorchVision that are backbone of the CNN network as well as a means to load the model for processing. The TorchVision python module consists of models, architectures, datasets and complex as well as simple image transformations for computer vision. Torch is an open-source machine learning library, a scientific computing framework, and a script language supported. It provides a good range of algorithms for deep learning, and uses the scripting language LuaJIT, and an underlying C implementation.

The core package of Torch is torch. It provides a versatile N-dimensional array or Tensor, which supports basic routines for indexing, slicing, transposing, type-casting, resizing, sharing storage and cloning. This object is employed by most other packages and thus forms the core object of the library. The Tensor also supports mathematical operations like max, min, sum, statistical distributions like uniform, normal and multinomial, and BLAS operations like scalar product, matrix-

vector multiplication, matrix-matrix multiplication, matrix-vector product and matrix product.

The nn package also comprised in the Torch module is employed for building neural networks. It's divided into modular objects that share a standard Module interface. Modules also have a forward() and backward() functions that allow them to feedforward and backpropagate, respectively. Modules are often joined together using module composites, like Sequential, Parallel and Concat to make complex task-tailored graphs. Simpler modules like Linear, Tanh and Max structure the basic component modules. This modular interface provides first-order automatic gradient differentiation.

The prediction module takes an image or cluster of image as input and returns the predicted class label along with its accuracy. Not only is the prediction single-class type rather it is a multi-class prediction of up to five classes at a time with ascending order of prediction accuracy. The places365 CNN is used to return the classification model output. The classes365 CNN model is trained on a vast dataset of 400+ classes with a images dataset size of 105 GB. A training dataset having at least 5,000 images and at most 22,000 images was used for each of the classes. The predicted output is shown below in Fig.8.

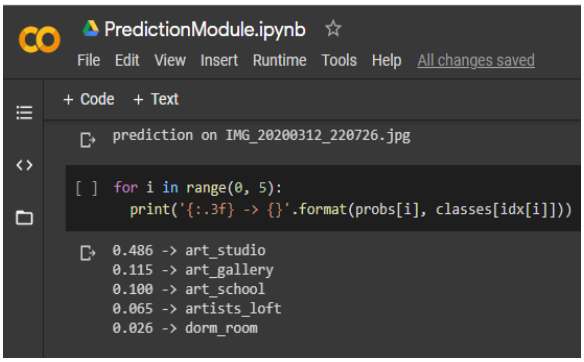

Figure 8. Prediction module result

2) Date and Time Module

The date and time module extract the time stamp as well as the date stamp. The time stamp is in UTC format. This is a metadata common to all the media capture devices irrespective of any internet or GPS. The date and time module takes an image or cluster of images as input and returns the Date as well as the Time when the image was taken accurately. This module is implemented by implementing the python GPS Photo module that depends on the date and time stored along with the image in an encrypted form which is deciphered by the code run against the input image/images. The resulting output of this block will be as shown in Fig. 9.

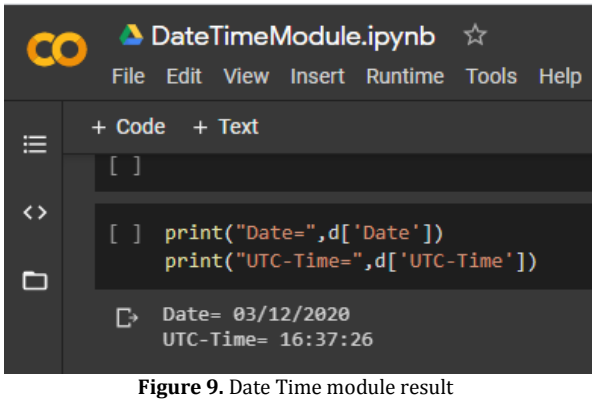

Figure 9. Date Time module result

3) GPS module

The GPS module requires that the photo be captured with the GPS switched on. The GPS module takes an image or cluster of image as input and returns the Latitude as well as the Longitude of where the photo was taken accurately. This module is implemented by implementing the python GPSPhoto module that depends on the GPS data stored along with the image in an encrypted form which is deciphered by the code run against the photo.

Both the GPS module and Data and time module have EXIF data as their backbone. EXIF is also a python module used to extract metadata from image files. GPS Photo is also another one of the key components of GPS module and Data and time module. It is a module that uses ExifRead and piexif to extract GPS Tag Data from jpeg and tiff format photos. This module was coded by the ExifRead, piexif, and Pillow (PIL) modules. The latitude and longitude are extracted as shown in Fig.10.

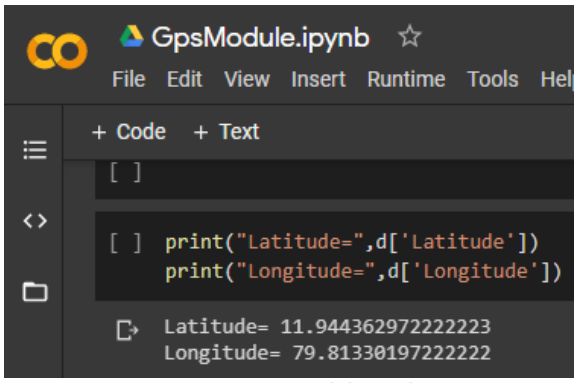

Figure 10. GPS module result

4) OCR Module

Python-tesseract is an optical character recognition library for python. That is, it'll recognize and print the text embedded in images. This module needs to be imported to run OCR against our input image. Python-tesseract may be a wrapper for Google's Tesseract-OCR Engine.

It's also a stand-alone invocation script to tesseract, because it can read all image types supported by the Pillow and Leptonica imaging modules, including jpg, png, gif, bmp, tiff, et al. Additionally, if used as a script, Python-tesseract will print the recognized text rather than writing it to a file.

The OCR module takes an image or a cluster of images as input and returns the text embedded in it as the output as shown in Fig.11.

This is a complete opposite to the traditional method which uses hardware to work like the OCR detector pen. The software module neatly captures the essence of Google's OCR service. The output of this module is shown below in fig. 12.

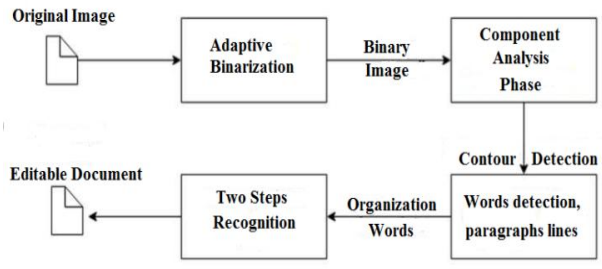

Figure 11. OCR module

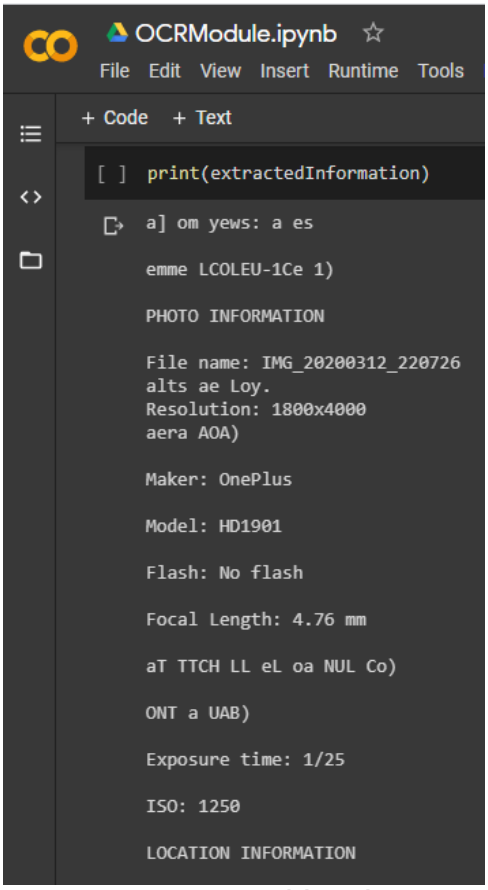

Figure 12. OCR Module result

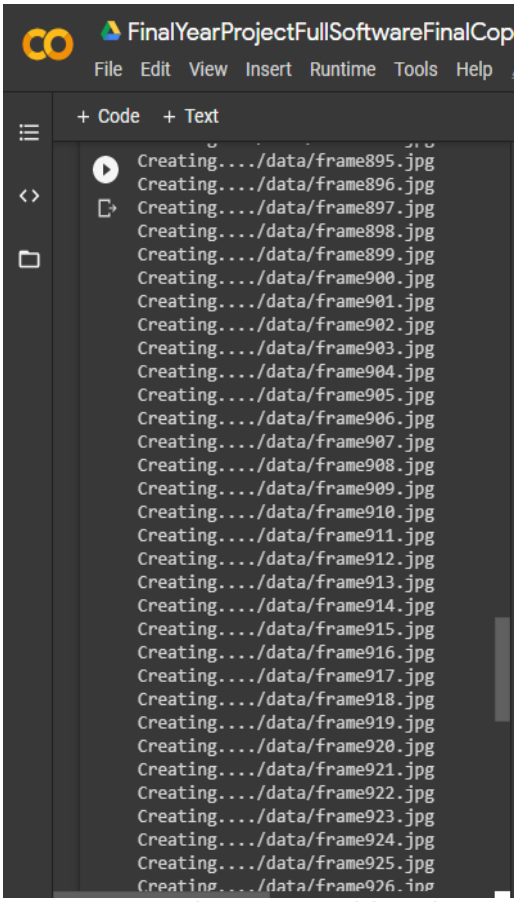

Figure 13. Video to images module result

5) Video to Images Module

OpenCV is a cross-platform library using which we can develop real-time computer vision applications. It mainly focuses on image processing, video capture and analysis including features like face detection and object detection. The openCV services is made available by importing cv2 module to facilitate video processing. The Video to Images module is implemented using CV2 library of openCV python library. It takes a video as input and converts it to frames. These frames are images which are stored in cloud and can be accessed by the Colab jupyter notebook which can be later used for upgrading from image only interface to image/video interface and the corresponding output is shown in Fig.13.

6) Spreadsheet Module

The Spread Sheet module is implemented using Google spread sheet API v4 which is an interface between the Colab jupyter notebook and Google's Spread Sheets application. The spread sheet module is imported in python. After importing the module the jupyter notebook needs access to Google sheets to create a spread sheet and append data to it. The extracted information from the image is stored in spread sheets which are accessed by the user's authentication key for his/her Google account. The Spread sheet itself is stored in google drive which can be accesses from anywhere anytime. The spreadsheet module and its corresponding sheet is shown in fig.14 and fig.15.

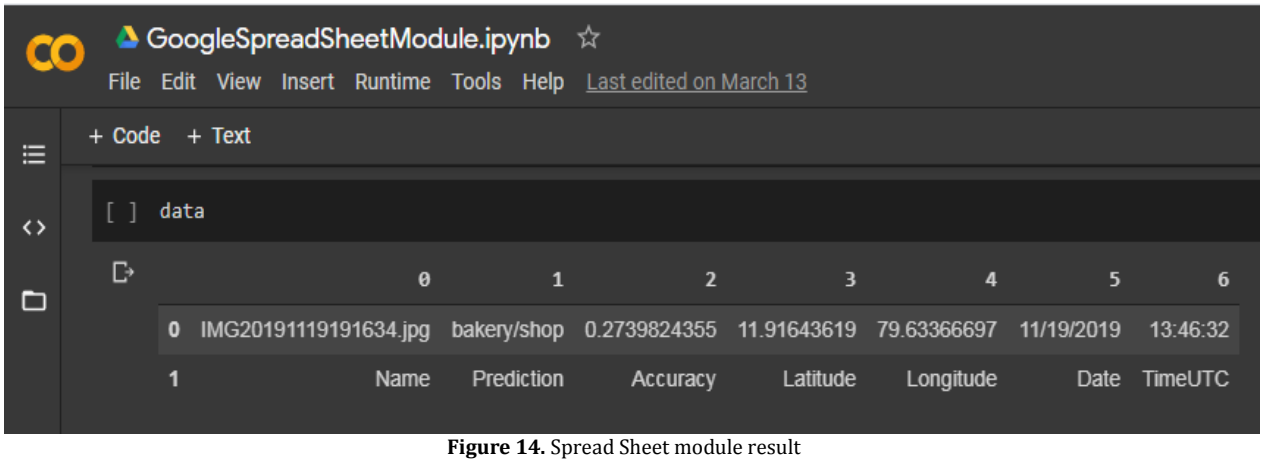

Figure 14. Spread Sheet module result

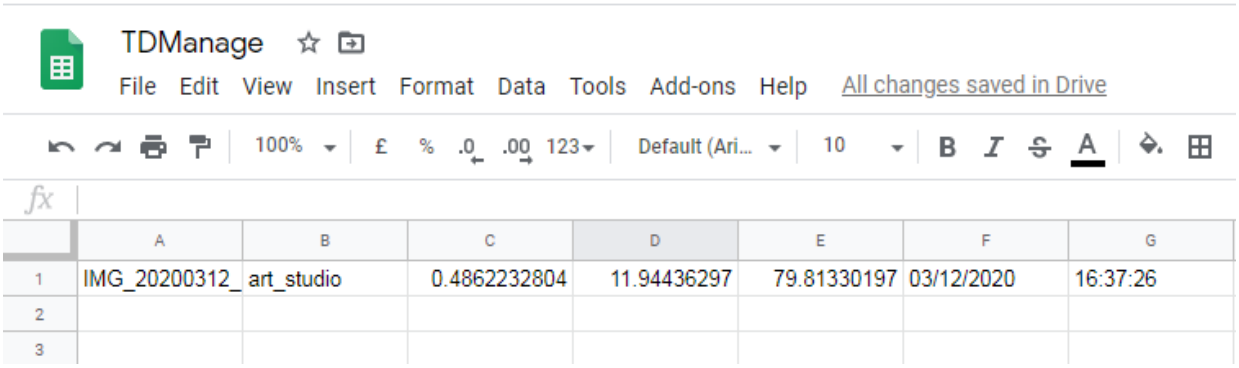

Figure 15. Data stored in Spread Sheets

7) Data Visualization Module (Non Pictorial)

Online Google Spread Sheet application is used for further data processing. It can be accessed using a browser on any and all device. All a person needs is the authentication key for his/her Google account. Later the stored data can be plotted using a chart in two dimension of latitude vs. longitude.

Also to print out the data contained in the spreadsheet the Pandas module is imported in the jupyter notebook. This module is used to print the xsls sheet in the terminal for physically visualizing data or weed out corrupt data if any embedded in the spreadsheet

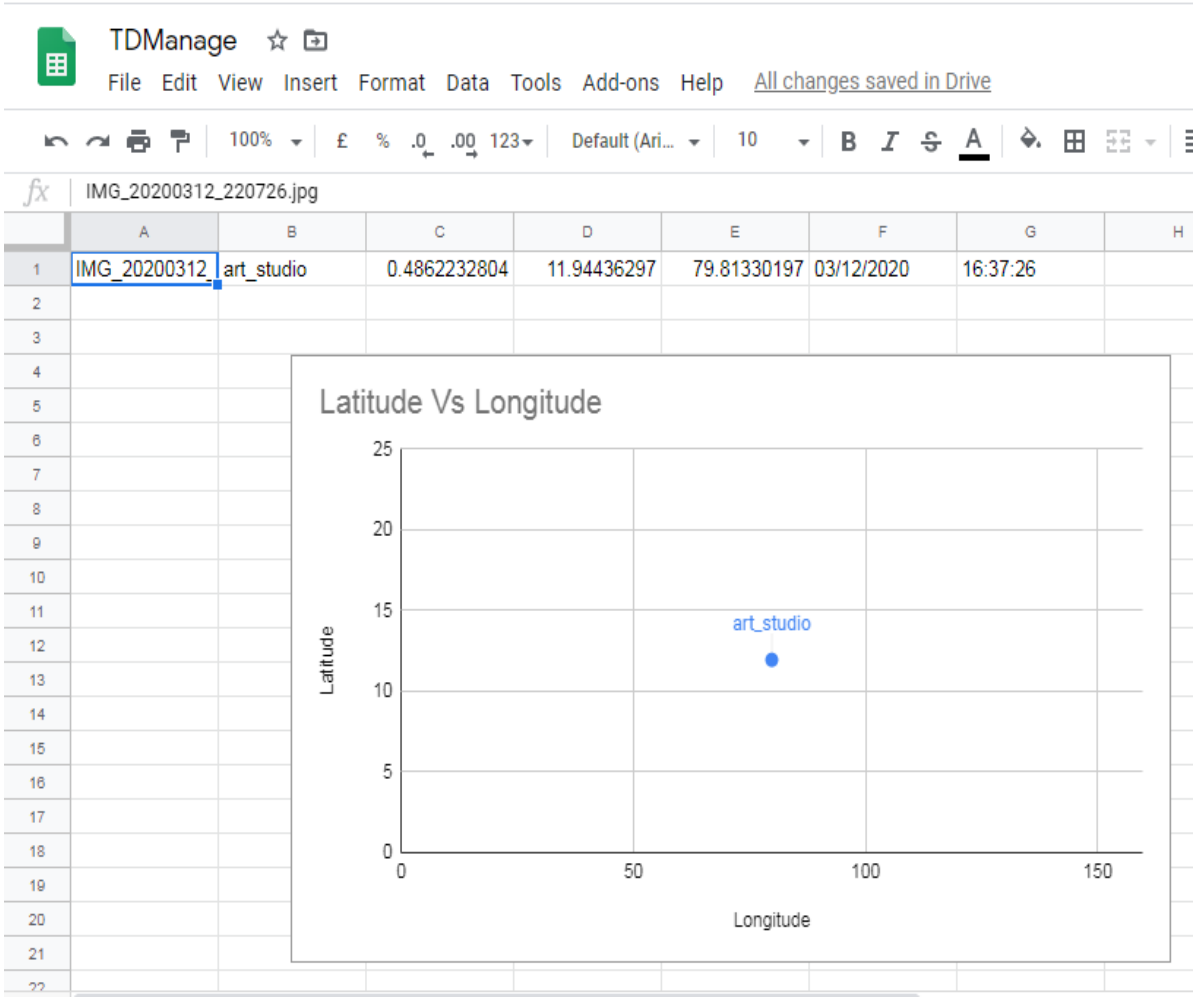

Figure 16. Pictorial Representation using graph

Finally live update visualization is done by uploading the excel sheet containing the update data to my maps feature in Google maps. The upload is direct. The columns containing key parameters are verified for plotting a location on the map trace.

The pictorial graph formed by an uploaded image is shown in fig.16. and the same in pictorial visualization using Google maps is shown in fig.17.

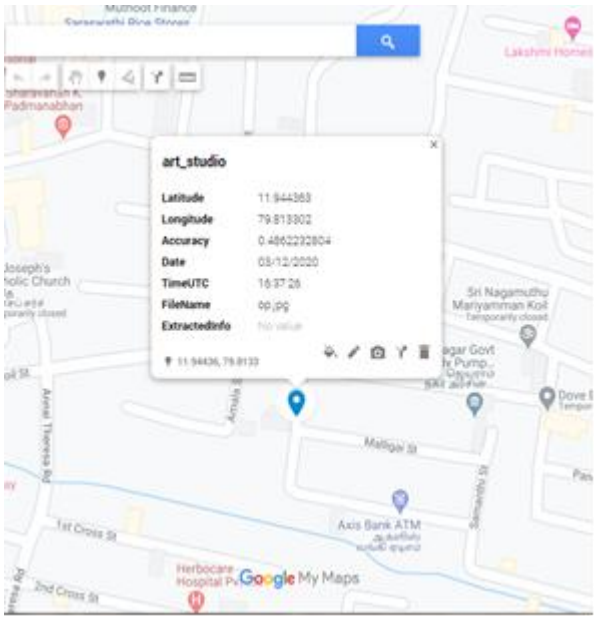

Figure 17. Pictorial Visualization using Google Maps

CONCLUSION

The proposed software solution is tested to be a reliable solution for many travellers, door delivery services and taxi firms for their navigation. The map is updated by the crowed source ranging from taxi drivers to delivery boys. The map updating process is made simpler to the users by providing them to upload an image to the server from their smartphone instead of old fashioned textual procedures. The R-CNN prediction model backed by Google colab is used to predict the images uploaded to the server and later it is marked in the picture plane for navigation. Thus the proposed map updating process by machine learning is considered to be a real game changer in the world of automation and can be further improved with the virtually abundant libraries of python and inhuman capabilities of Artificial Intelligence.

A. Salient Features of Proposed Model

1) No human interference needed
2) Faster than manual updating of location services
3) Real time updation data processing for location services
4) Able to keep up with the changes in metropolitan areas
5) Ability to inculcate automation in every field
6) Updating of location services in real time
7) Able to act as monitoring system for threats
8) Able to act up on cases of mishaps and accidents
9) Able to identify offenders of any sort using image processing and identification matching
10) Fast update in real time

B. Future Scope

Machine learning can be further upgraded to run faster, smoother and better. All it needs is a little push. With big data coming into the picture we could increase the capabilities of artificial intelligence manifold.

More attractive features can be added to make the system robust. But with data collection comes responsibility. It is the user's responsibility to protect and safeguard the data he need to keep safe from the vying eye.

Also if we come upon confidential or high risk data we need to do the right thing and dispose of it for the safety of the individual or groups involved. Also we can implement our method to support the videos if we are able to cross interface GPS and video camera i.e. Location live streaming

REFERENCES

1. Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba, "Places: A 10 million Image Database for Scene Recognition", 10.1109/TPAMI.2017.2723009, IEEE Transactions on Pattern Analysis and Machine Intelligence
2. Haoyu Wang, Mohammad Hadian, and Xiaohui Liang," Efficient and Privacy-preserving Roadmap Data Update for Autonomous Vehicles", 2018 IEEE
3. Chu Cao, Zhidan Liu, Mo Li, Wenqiang Wang, and Zheng Qin," Walkway Discovery from Large Scale Crowdsensing", 2018 17th ACM/IEEE International Conference on Information Processing in Sensor Networks
4. Yanmin Zhu, Yin Wang, George Forman and Hong Wei, "Mining Large-Scale GPS Streams for Connectivity Refinement of Road Maps", Computational Intelligence, Machine Learning and Data Analytics, The Computer Journal, Vol. 58 No. 9, 2015
5. Suo Li, Haipeng Wang and Feng Xu, "Land Cover Generation From Optical Image", IGARSS 2018
6. Briana M. Sullivan," ChUM: Chart Update Mashup", 2012 IEEE
7. Thunyasit Pholprasit, Suporn Pongnumkul, Chalermpol Saiprasert, Sarinthon Mangkorn-ngam and Lalida Jaritsup, "LiveBusTrack : High-frequency Location Update Information System for Shuttle/Bus Riders", 2013 13th International Symposium on Communications and Information Technologies (ISCIT)
8. Yuuichi Teranishi , Junzo Kamahara and Shinji Shimojo, "MapWiki: A Ubiquitous Collaboration Environment on Shared Maps", International Symposium on Applications and the Internet Workshops (SAINTW'06)
9. Hiromitsu Sugawara, Nobuyoshi Sato, Tsuyoshi Takayama and Yoshitoshi Murata, "Early Evaluation of Road Width Estimation on Rapid Road Map Survey System Using GPS Trajectories as Collective Intelligence", 2011 International Conference on Network-Based Information Systems
10. Mohammad Nazmul Hasan and Md. Sharif Hossen, "Development of An Android Based Real Time Bus Tracking System", 1st International Conference on Advances in Science, Engineering and Robotics Technology 2019 (ICASERT 2019)
11. R.Immanuel Rajkumar, P.E.Sankaranarayanan and G.Sundari, "GPS and Ethernet based Real Time Train Tracking System", 2013 International Conference on Advanced Electronic Systems (ICAES)
12. Mark Smith, "Intelligent Mobile Application for Traffic Monitoring", 2015 12th International Conference on Information Technology - New Generations
13. He Li and Lai Zhijian, "The Study and Implementation of Mobile GPS Navigation System Based on Google Maps", International Conference on Computer and Information Application（ICCIA 2010）
14. Yin Wang, Xuemei Liu, Hong Wei, George Forman, Chao Chen and Yanmin Zhu, "CrowdAtlas: Self-Updating Maps for Cloud and Personal Use", MobiSys'13, June 25-28, 2013, Taipei, Taiwan