MULTI-BRANCH DEEP LEARNING BASED TRANSPORT MODE DETECTION USING WEAKLY SUPERVISED LABELS

P. Vinayaraj *, K. Mede

Rakuten Institute of Technology, Rakuten Group, Inc., Tokyo, Japan
(vinayaraj.poliyapram, kyle.mede)@rakuten.com

KEY WORDS: Transport mode detection, Deep learning, Weakly supervised label generation, Global Positioning System

ABSTRACT:

Mobility data, based on global positioning system (GPS) tracking, have been widely used in many areas. These include, but not limited to: user direction guidance, analyzing travel patterns, and evaluating travel impacts. Transport Mode Detection (TMD) is an essential factor in understanding mobility within the transport system. A TMD model assigns a GPS point or a GPS trajectory to a particular transport mode based on the user's current activity. However, the complexity of the prediction procedure increases with the number of modes that need to be predicted given the increasing overlaps in feature values between multiple transportation modes. Hence, this study proposes a two-branch deep learning-based TMD model that predicts multi-class transport modes to improve prediction accuracy. In addition, it proposed a weakly supervised labelling model using snorkel to improve the volume of labelled data and resulting TMD model prediction accuracy. We considered publicly available road networks, railway networks, bus routes, etc., for creating road, bus, train labels by overlaying GPS points on these transportation networks. We introduced a boolean (true/false) based soft-labelling function, where the same GPS point overlaid on road or railway network. The raw GPS data were used to generate point-level features such as speed, speed difference, acceleration, acceleration difference, initial bearing and bearing difference, all used as derived features for the TMD model. To construct the model we opted to use two branches where raw GPS latitude and longitude values were used in one and the derived mobility features are used in the other.

1. INTRODUCTION

The use of data, based on global positioning system (GPS), or some other equivalent position system, offers the possibility of following individual trips with regard to temporal and spatial positional characteristics in a much more detailed way, when compared to traditional travel surveys and travel diaries (Clifford et al., 2009), (Nguyen et al., 2020). GPS tracking data are becoming ubiquitous in various transportation applications and have been widely used in collecting information on people and goods transport (Furletti et al., 2013). The advancements in GPS technology and devices used to collect data are improving rapidly, and data generated by GPS devices have become more accurate and reliable (Wu et al., 2016). In general, GPS tracking data provides information on longitude, latitude, date, time, speed, altitude, and direction of movement. In order to conduct further transport analysis, data processing is necessary to extract additional trip characteristics. The major processing steps are trip identification, map matching, and transport mode detection (Gong et al., 2014), (Yang et al., 2015).

Many studies are available focusing on trip identification and map matching, while studies on transport mode detection (TMD) are still sparse (Huang et al., 2019). The limited number of reviews in the area could be due to the fact that transport mode detection based on GPS tracking data is challenging, especially when the data is unlabelled, i.e., there is no information regarding the transport mode used during a trip (Huang et al., 2019). This was one of the key issues we addressed in this study by proposing a weakly supervised label generation model using snorkel (Ratner et al., 2017a). Several previous studies utilized snorkel for label generation for various other purposes (Ratner et al., 2017b), (Ratner et al., 2018), however, for the first time a study utilizing snorkel for TMD label generation.

*Corresponding author

There are a couple of studies that have utilized machine learning approaches for TMD. For example, (Ashqar et al., 2019) introduced a Hierarchical Machine Learning Classifier for TMD, while (Qin et al., 2019) proposed a deep convolutional neural network (CNN) with a recurrent neural network. Other studies utilized Long Sort-Term Memory (LSTM) models to incorporate the temporal information more effectively to detect transport mode (Asci and Guvensan, 2019). However, these studies are using readily available labelled datasets; while in our study we proposed a label generation method as well as a multi branch deep learning model which includes the raw latitude and longitude data as well. Several well known architectures of neural networks such as ResNeXt (Pant et al., 2020), Inception (Szegedy et al., 2017) are based on the designing idea of having multiple branches and have demonstrated improved performance in many applications. A multi-branch deep learning model provides the opportunity to incorporate entirely different domain dataset and a more meaningful feature concatenation. This approach has the advantage of generating features using a supervised manner. Therefore, we opted two-branch approach for our proposed deep learning architecture where one branch utilizes transportation network/ location information, while the other utilizes the mobility features.

The key contributions of our work are summarized as below;

- Proposed a weakly supervised model approach for transport mode label generation with GPS features as well as the transportation network features
- Proposed a two-branch deep learning model to utilize raw data insights along with mobility features
- Demonstrated the model performance compared with previous machine learning model as well as with the iOS and android in-built transport mode recognition tool.

2. STUDY AREA AND DATA

The main motivation of this study is to build an elegant TMD model for the Japan region where no TMD ground truth data is openly available. However, Japan has various openly available datasets with reliable coverage, such as Open Street Map (OpenStreetMap contributors, 2017), bus routes (Global map japan, 2020), Cycle routes etc. Hence, this study utilized many of these openly available datasets, GPS raw data and GPS derived features such as speed, acceleration, jerk, bearing, etc., for building a weakly supervised model to generate ground truth data for TMD supervised modeling.

2.1 Data

This study used various datsets with the characteristics of these datasets given in Table 1. Table 1 overviews the characteristics of each dataset such as its type, provider and availability. All the datasets, other than the GPS data, are openly available and used for labelled data generation for TMD. The GPS data is from a private dataset used in this study and is derived from multiple mobile applications.

Data	Availability	Provider	Type
Road network	Public	OSM	Polyline
Railway network	Public	OSM	Polyline
Bus route	Public	Japanese govt.	Polyline
Bounary	Public	Japanese govt.	Polygon
GPS	Private	Rakuten apps	Point

Table 1: Characteristics of the datasets used in this study

3. METHODOLOGY

Two distinct methodologies are involved in this work, one to prepare ground truth data using a weakly supervised modelling approach and the other is a deep learning classification model for the TMD. However, GPS data with latitude, longitude and timestamp is the primary data source used for TMD modelling. There are no sufficient open-sourced ground-truth data available for transport modes in Japan. Hence, we proposed a transport mode label generation approach using snorkel (Ratner et al., 2017a). Further, these generated labels are used to trained a multi-branch deep learning model. The following subsections will explain in detail about weakly supervised label generation and multi-branch deep learning model.

3.1 Feature engineering

Feature engineering is one of the most important steps undergone in any machine learning or deep learning modelling approaches. This study derived various mobility related features from GPS data for feeding those into the deep learning model. Primary data source for this feature extraction was GPS pings with latitude, longitude and timestamps. These features can be sub-categorized into point level features and trajectory level features. Point level features are computed using the comparisons between previous points and current point, while, trajectory level features considers all points in a trajectory to derive features. Table 2 shows the characteristics of each feature and their data types. There are seven point level features and two trajectory level features, the equations for each of which are given below. The four fundamental features are the speed, acceleration, jerk and bearing, while trajectory features are simply average over a single trajectory as shown in Table 2.

Equation 1 shows the distance calculation, with the resulting distance used for the speed calculation. Equation 2, 3 and 3 are used to calculate the bearing

$$d = 2Rarcsin\sqrt{sin^2\frac{\Delta\varphi}{2} + cos(\varphi_1)cos(\varphi_2)sin^2\frac{\Delta\lambda}{2}} \quad (1)$$

Where, R is the radius of earth in meters (6371000), $\Delta\varphi$ is the difference in latitude (radians), $\Delta\lambda$ is the difference in longitude (radians).

$$X = cos(\phi_b) * sin(\lambda_b - \lambda_a) \quad (2)$$

$$Y = cos(\phi_a) * sin(\phi_b) - sin(\phi_a) * cos(\phi_b) * cos(\lambda_b - \lambda_a) \quad (3)$$

Where a is a and b is the point

$$b = atan2(X, Y) \quad (4)$$

Where X and Y are derived from equation 2 and equation 3 respectively.

Where, R is the radius of the earth in meters (6371000), $\Delta\varphi$ is the difference in latitude (radians), $\Delta\lambda$ is the difference in longitude (radians), d represents distance, v represents speed and a represents acceleration in Table 2.

Feature	Level	Derivation	Unit
Speed	Point	$d\,/\,({}_{t-1} - {}_{t0})$	m/s
Speed difference	Point	$\mathrm{v}_{t-1} - v_{t0}$	m/s
Acceleration	Point	$(\mathrm{v}_{t-1} - v_{t0})/({}_{t-1} - {}_{t0})$	m/s^2
Acceleration difference	Point	$\mathrm{a}_{t-1} - a_{t0}$	m/s^2
Jerk	Point	$(\mathrm{a}_{t-1} - a_{t0})/({}_{t-1} - {}_{t0})$	m/s^3
Bearing	Point	b = atan2(X,Y)	$^\circ$
Bearing difference	Point	$\mathrm{b}_{t-1} - b_{t0}$	m/s
Average speed	Trajectory	$\frac{1}{n}\sum_{i=i}^{n} v_i$	m/s
Average acceleration	Trajectory	$\frac{1}{n}\sum_{i=i}^{n} a_i$	m/s^2

Table 2: Features derived from the GPS dataset.

3.2 Weakly supervised label generation

In any supervised learning approach, including deep learning, labelled data availability is critical to maximize model performance. As like in several other domains labelled ground truth data is not available in case of transport mode. As far as we know, there is no open ground-truth data available for transport mode detection for the Japan region. Hence, this study proposes a transport mode label generation approach using snorkel (Ratner et al., 2017a). Snorkel is a weakly supervised labelling function, a first-of-its-kind system that enables users to train state-of-the-art models without hand labelling any training data. Instead, users

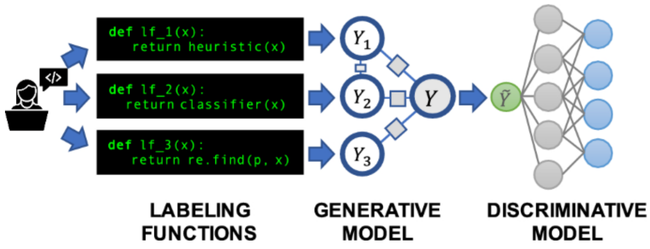

Figure 1: Example for a snorkel label generation model

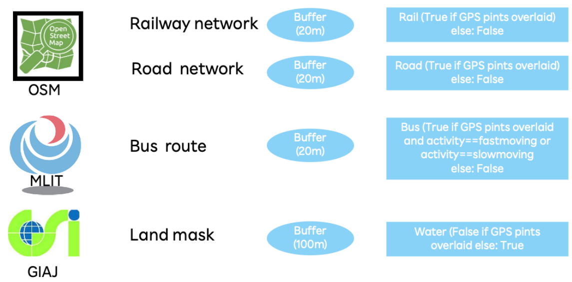

Figure 2: Soft-labelling approach using transportation network and geospatial datasets

write labelling functions that express arbitrary heuristics based on their expertise and understanding of the data in the related field, transport mode identification in this case. The primary difference between manual labelling and programmatic labelling is the type of input that the user provides. With manual labelling, user input comes in the form of individual labels, created one by one. With programmatic labelling, users instead create labelling functions, which capture labelling rationales and can be applied to vast amounts of unlabelled data and aggregated to auto-label large training sets. This approach leads to a number of benefits over manual labelling such as scalability, adaptability and governability (Ratner et al., 2017a).

In this study we used snorkel for generating the ground truth data for transport mode. We considered publicly available datasets related to road network, railway network, bus routes, etc.These datasets were then used for creating road, bus, train labels by overlaying GPS points on associated transportation network. The road and rail network s were extracted from Open Street Map (OSM) (OpenStreetMap contributors, 2017), bus routes downloaded from the openly available sources from Ministry of Land, Infrastructure, Transport and Tourism (MLIT) (Ministry of Land Infrastructure and Tourism, 2020) and Japan boundary from the Geospatial Information Authority of Japan (GIAJ) (Global map japan, 2020). Figure 2 shows, the flowchart of soft-labelling generation using underlying these transportation networks. A buffer of 20m was created around road and railway network to identify the overlaid points. We chose a 20m buffer size as the average spatial resolution of the GPS points in our dataset were approximately 20m. The boundary polygon was assigned a large 100m buffer to account for the polygon dataset being out of date and low spatial resolution.

There are multiple occasions where the road, bus and train classes overlap each-other, especially in a city region. Hence, we introduced a boolean type (True/False) based soft-labelling function, where same GPS point might have multiple True values for road or railway or bus. These boolean values were utilized in the snorkel model along with mobility features such as speed, acceleration, etc.

```
# Define the labeling functions for Snorkel
@labeling_function()
def lf7(x):
    return CAR if (x.speed>= 15 and x.speed<=33 and  x.rail == False) else ABSTAIN
@labeling_function()
def lf8(x):
    return CAR if (x.speed>=1.2 and x.speed<=33 and x.rail == False) else ABSTAIN
@labeling_function()
def lf9(x):
    return CAR if (x.acceleration>=3 and x.road == True) else ABSTAIN
@labeling_function()
def lf10(x):
    return CAR if (x.acceleration_difference >= -11.0 and x.acceleration_difference <= 11.0 and x.road == True)
        else ABSTAIN
@labeling_function()
def lf11(x):
    return BUS if (x.speed>=2.5 and x.speed<=10 and x.bus==True) else ABSTAIN
@labeling_function()
def lf12(x):
    return BUS if (x.acceleration_difference <= -6 and x.speed<=6 and x.bus==True) else ABSTAIN
@labeling_function()
def lf13(x):
    return WALK if (x.speed <=1.4 and x.acceleration < 1.5 ) else ABSTAIN
@labeling_function()
def lf14(x):
    return WALK if (x.speed <=1.4  and x.bearing_difference > 100 and x.rail == False) else ABSTAIN
@labeling_function()
def lf15(x):
    return WALK if (x.speed <=1.4  and x.bearing_difference > 150 and x.rail == False) else ABSTAIN
@labeling_function()
def lf16(x):
    return WALK if (x.acceleration_difference >= -6.0 and x.acceleration_difference <= 5.0 and x.speed <=1.4)
        else ABSTAIN
@labeling_function()
def lf17(x):
    return BIKE if (x.speed>=2.5 and x.speed<=4.0  and x.bearing_difference > 150 and x.road == True and
        x.rail == False) else ABSTAIN
```

Figure 3: Sample of labelling functions used for weakly supervised labelling

3.3 Multi-branch deep learning

The snorkel-based label generation model needs additional transportation network-based soft-labelled classifications; these were generated using computationally expensive spatial joining and spatial indexing jobs. Hence, this approach is not scalable for big data TMD. Therefore, this study proposed a separate deep learning model which can utilize mobility features as well as the raw GPS latitude and longitude for TMD. Transportation network based soft-labelling and other mobility features are used to define labelling functions in snorkel. These label functions are then used to create true ground truths by means of a generative machine learning model with a limited number of GPS data. The generated labels (walk, cycle, bus, car, train, boat/ship) were used to train the proposed deep learning model. For the TMD model we propose a two-branch deep learning architecture where raw GPS latitude and longitude values are used in one branch and derived mobility related features are used in the other branch Figure 4. We used 3 fully-connected hidden layers for raw GPS data (latitude/longitude) with 256, 128 and 32 hidden layers respectively. For mobility features we used 4 fully connected hidden layers for mobility features with 256, 128, 64 and 64 hidden layers accordingly. Features derived from the two branches are concatenated in feature domains. Furthermore, 3 fully connected hidden layers with 128, 64, 32 hidden layers and softmax cross-entropy were used as a loss function. The proposed deep learning model has 108,614 trainable parameters and Adam is used as an optimizer. The TMD model is implemented using the Keras python API (Chollet et al., 2015).

This particular two-branch model structure achieves better accuracy as it combines raw data as well as the derived mobility features in the network. An example of this is the relationship between latitude/longitude and one of the road driving classes, thus inferring the location is on a road. Note, many of these inferences that improve classification accuracy are possible via dramatically more advanced pre-processing to build out additional features, but that process can be time consuming and would be unlikely to catch all the potential inferences that an un-biased set of deep learning layers can inherently extract. When building classification models, a data scientist first conducts feature engineering to incorporate their industry knowledge to build features from the raw data to make it easier to decide what class the input data point is, and thus increase the model's classification accuracy. With this, the model is built trained using the extracted features alone. By incorporating the raw data as well as one of the features, there are two options in a single branch model architecture: 1) Using a similar complexity (number of nodes per layer) model, which can cause high risk of the additional raw data reducing the effectiveness of the node tuning and thus reducing the overall model's

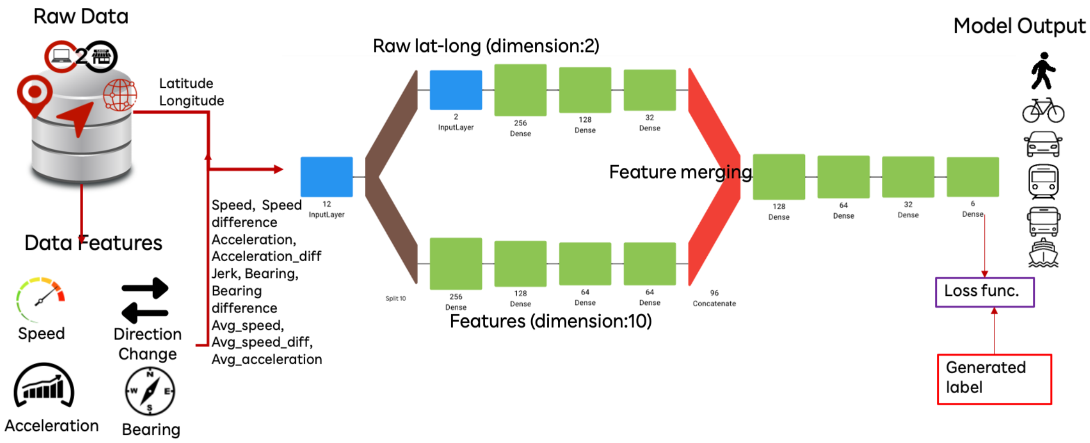

Figure 4: Multi-branch deep learning model for TMD.

classification accuracy. Or 2) increasing the model complexity (increasing the number of nodes per layer), enabling increased tunability at the risk of causing tuning biases that would require greater training duration and larger training data volume to help mitigate those biases. Thus, in our model we created a separate and specifically designed branch to draw out any additional intelligence/classification indicators not already included in the extracted features without overly complicating the model.

3.4 Pipeline and deployment

The TMD model introduced in the previous section has been deployed to predict transport mode on big commercial mobile application data that is collected every day in a large volume. Hence, the pipeline deployment, job scheduling and maintenance are very critical for this application. However, this study utilized various Free and Open Source Solutions (FOSS) for implementing the deep learning model using a scheduled pipeline for estimating TMD on a daily basis. Multiple dedicated mobile applications collect location information, and the pipeline will gather daily data and process using a function as a service (FaaS). The jobs are scheduled and triggered as daily batch using he open source scheduler Apache Airflow (Kotliar et al., 2019). The implementation utilizes Apache Hadoop (Nandimath et al., 2013) clusters to collect, gather and process big GPS data and Apache Hive tables are used to store the daily batch TMD results. We used PySpark via the Python API for Apache Spark (Spark, 2018) as an open source distributed computing framework. In addition to providing an API for Spark, PySpark helped us to interface with Resilient Distributed Datasets (RDDs) by leveraging the Py4j library. Several geo-spatial analysis were carried out to pre-process and prepare deep learning ready data using geopandas (Jordahl, 2014), shapely (Westra, 2015) and rtree (Alfarrarjeh et al., 2020).

Figure 5 shows architecture of the pipeline implementation. Mobile application collected data is stored and maintained in Apache hive tables, with queries performed using spark sql and processing using pyspark. Furthermore, these results are stored in Apache hive tables. Apache Airflow is used to orchestrate scheduling jobs on the previously mentioned daily basis. All compute and data storage tasks described here are performed using Rakuten's internal cloud infrastructure, ensuring all data privacy, usage restrictions and security regulations are adhered to.

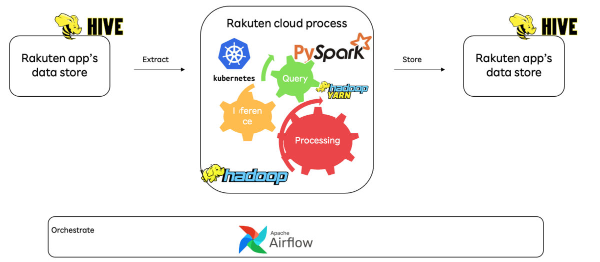

Figure 5: General solution architecture for pipeline deployment

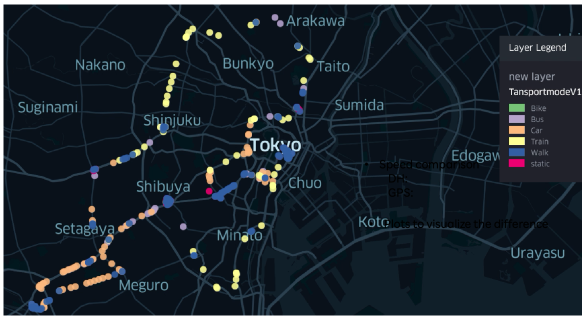

Figure 6: Demonstration of transportation mode for a single user in a day

4. RESULT AND DISCUSSION

We used weakly supervised labels for the deep learning model training and evaluation, hence it is important to evaluate performance of the model with unseen data. We carried out quantitative and qualitative evaluations, where test data have been collected by multiple users for various parts of Kanto, Japan.

4.1 Comparison with previous model

The proposed deep learning model was trained on a subset of the user GPS data, with the model showing an overall accuracy of 0.85. However, it is also beneficial if the model performance can be evaluated in detail with real-world use-cases. Therefore, we compared the proposed model with our own previous model, hereafter call as TMD V0.1. TMD V0.1 is a machine learning model trained using a XGBoost classifier (Chen et al., 2015), this model also utilized a similar set of input mobility features as the model proposed in this work. The primary difference between the two versions are the proposed one is a multi-branch deep learning model with raw latitude/longitude information as an additional input.

Figure 6 demonstrates the TMD for GPS trajectory of a single user for a single day who uses one of the transportation modes supported by the models. Figure 6 overlays the resulting predicted transportation modes onto a map, where each GPS point assigned a transport mode. We compared the results against the popular XGBoost classifier (TMD V0.1), with our model producing over 5% higher accuracy for the benchmark Geolife dataset (Zheng et al., 2011). The qualitative evaluation of the results is carried out in two ways, including demonstration of user trajectory with model detected transport mode as shown in 6 and comparing the obvious user coverage expected in Japan. Table 3 shows the user coverage increase is largest for car and train modes. Apart from that, we also identified the increase in train user coverage in city areas where the primary transportation mode of people is train.

KPI	TMD V0.1	TMD V1.0
Number of modes	4	6
Car user coverage	20%	32%
Train user coverage	4%	9%
Prediction accuracy	35.62%	66.99%

Table 3: Comparison between TMD V0.1 & TMD V1.0 results on field testes conducted by various voluntary users.

Moreover, we collected smartphone-based GPS trajectories for multiple modes of transportation collected by testers in Tokyo, Japan. This test is conducted on a completely new and dataset with a different spatial resolution and temporal resolution compared to the dataset used for training the model. With this new absolute ground truth data we compared the resulting predicted classes between TMD V0.1 and TMD V1.0. Figure 7(a), 7(b) and 7(c) show the locations of field tests where user's transport mode was Car. Figure 7(d), 7(e), and 7(f) show the locations of field tests where user's transport mode was Car. Table 3 shows significant improvement in the average overall accuracy for the unseen dataset collected over the Kanto region. Experiments show that TMD V1.0 results are promising with improved accuracy and increases in number of labelled data points. Apart from that, TMD V1.0 can predict six transport modes while TMD V0.1 can only only predict four modes.

This study also compared the results with on-device iOS (CMMotionActivity, 2020) and android (AndroidActivity recognition, 2020) in-built activity recognition tools using multiple field test carried out by different users in different popular road and railway routes the Kanto, Japan region. Table 4 shows the resulting accuracy for each field test. The proposed model shows significant improvement in average accuracy. Table 4 includes, highways, expressways, local roads, rapid train routes, local train routes. We also found that the proposed model (TMD V1.0) works reasonably well in both Android and iOS operating systems. The evaluation shows activity mode recognition improvement in average accuracy between all field tests from 74.4% to 92% compared to Android and iOS in-built systems. Of key note is that the iOS and android in-built activity recognition tools provide the 'automotive' class as a single class, while our proposed model efficiently distinguishes automotive classes as car, bus, and train with improved accuracy.

5. CONCLUSION

This research work proposes a two step TMD model, where the first step is to generate labels in a weakly supervised manner and the second is to train a multi-branch deep learning model. We were motivated to create this two step approach for two reasons, a) the alternative label data preparation mechanisms are computationally intensive for big spatial data and b) the alternative approach of robust feature engineering by experts to incorporate industry expertise is time consuming and unlikely to catch all inherent relationships between the raw data and the transportation mode. The weak supervised learning application to generate labels for TMD models, is one of the first of its kind, if not the first, and proved successful in reducing need for compute capacity even when processing big spatial data. The multi-branch deep learning TMD model to incorporate raw data and remove the need for robust feature engineering also accomplished its goal of improving classification by nearly 2x (35.63% to 66.99%) compared to our previous v0.1 model, and also resulted in higher average accuracy compared to the Android and iOS built in activity classification in our field testing (92% over the OS-based 74%).

Location	tmode	OS	Acc (OS)	Acc (v1.0)
Edogawa	car	iOS	96%	90%
Nishikasai	car	iOS	99%	95%
Yokohama	car	android	100%	83%
Jiyugaoka	train	android	47%	90%
Yokohama	train	android	70%	89%
Nagatsuta	train	iOS	100%	99%
Tokaichiba	train	iOS	67%	98%

Table 4: Comparison with iOS and android activity recognition results

(a) Edogawa (b) Nishikasai (c) Yokohama
(d) Jiyugaoka (e) Yokohama (f) Nagatsuta

Figure 7: Selected filed test locations in Kanto, Japan

We also noted that our model performed well due to the fact the transport mode is highly dependent on the location of the GPS points whether its overlying on the road or railway track or water. Our multi-branch approach will provide an underlying information about location, in addition, these underlying location facts are not frequently changed. However, the challenge of this study is its application in more complex urban transportation networks of metropolitan cities like Tokyo. For such cities, roads and railways are in close proximity, run directly parallel to each other and even directly overlap vertically introducing many cases with multiple viable transportation modes for a given GPS point. We assume that the combination with other mobility features have added distinguishing characteristics to adequately resolve the majority of those cases, but will add additional features and data sources in future versions as needed should further testing invalidate that assumption. This work highly depends upon Free and Open Source Solutions (FOSS) for data preparation, mobility feature generation, deep learning model training, and big data computing. One future stage of this study is to implement the proposed deep learning model on smartphones to enable on-device near real-time transport mode detection using tensorflowlite in Andoid devices as coreml in iOS devices.

REFERENCES

Alfarrarjeh, A., Kim, S. H., Hegde, V., Shahabi, C., Xie, Q., Ravada, S. et al., 2020. A class of r*-tree indexes for spatial-visual search of geo-tagged street images. In: 2020 IEEE 36th international conference on data engineering (ICDE), IEEE, pp. 1990–1993.

AndroidActivity recognition, 2020. https://developers.google.com/android/reference/com/google/android/gms/location/ActivityRecognitionApi. Accessed: 2022-03-30.

Asci, G. and Guvensan, M. A., 2019. A novel input set for lstm-based transport mode detection. In: 2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), IEEE, pp. 107–112.

Ashqar, H. I., Almannaa, M. H., Elhenawy, M., Rakha, H. A. and House, L., 2019. Smartphone transportation mode recognition using a hierarchical machine learning classifier and pooled features from time and frequency domains. IEEE Transactions on Intelligent Transportation Systems 20(1), pp. 244–252.

Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., Chen, K. et al., 2015. Xgboost: extreme gradient boosting. R package version 0.4-2 1(4), pp. 1–4.

Chollet, F. et al., 2015. Keras.

Clifford, E., Zhang, J. and Stopher, P., 2009. Determining trip information using gps data.

CMMotionActivity, 2020. https://developer.apple.com/ documentation/coremotion/cmmotionactivity. Accessed: 2022-03-30.

Furletti, B., Cintia, P., Renso, C. and Spinsanti, L., 2013. Inferring human activities from gps tracks. In: Proceedings of the 2nd ACM SIGKDD international workshop on urban computing, pp. 1–8.

Global map japan, 2020. https://www.gsi.go.jp/ kankyochiri/gm_japan_e.html.

Gong, L., Morikawa, T., Yamamoto, T. and Sato, H., 2014. Deriving personal trip data from gps data: A literature review on the existing methodologies. Procedia-Social and Behavioral Sciences 138, pp. 557–565.

Huang, H., Cheng, Y. and Weibel, R., 2019. Transport mode detection based on mobile phone network data: A systematic review. Transportation Research Part C: Emerging Technologies 101, pp. 297–312.

Jordahl, K., 2014. Geopandas: Python tools for geographic data. URL: https://github. com/geopandas/geopandas.

Kotliar, M., Kartashov, A. V. and Barski, A., 2019. Cwl-airflow: a lightweight pipeline manager supporting common workflow language. Gigascience 8(7), pp. giz084.

Ministry of Land Infrastructure and Tourism, 2020. ” https://www.mlit.go.jp/en/”.

Nandimath, J., Banerjee, E., Patil, A., Kakade, P., Vaidya, S. and Chaturvedi, D., 2013. Big data analysis using apache hadoop. In: 2013 IEEE 14th International Conference on Information Reuse & Integration (IRI), IEEE, pp. 700–703.

Nguyen, M. H., Armoogum, J., Madre, J.-L. and Garcia, C., 2020. Reviewing trip purpose imputation in gps-based travel surveys. Journal of Traffic and Transportation Engineering (English Edition) 7(4), pp. 395–412.

OpenStreetMap contributors, 2017. OpenStreetMaps. https://www.openstreetmap.org . Accessed: 2010-09-30.

Pant, G., Yadav, D. and Gaur, A., 2020. Resnext convolution neural network topology-based deep learning model for identification and classification of pediastrum. Algal Research 48, pp. 101932.

Qin, Y., Luo, H., Zhao, F., Wang, C., Wang, J. and Zhang, Y., 2019. Toward transportation mode recognition using deep convolutional and long short-term memory recurrent neural networks. IEEE Access 7, pp. 142353–142367.

Ratner, A., Bach, S. H., Ehrenberg, H., Fries, J., Wu, S. and Ré, C., 2017a. Snorkel: Rapid training data creation with weak supervision. In: Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases, Vol. 11number 3, NIH Public Access, p. 269.

Ratner, A., Hancock, B., Dunnmon, J., Goldman, R. and Ré, C., 2018. Snorkel metal: Weak supervision for multi-task learning. In: Proceedings of the Second Workshop on Data Management for End-To-End Machine Learning, pp. 1–4.

Ratner, A. J., Bach, S. H., Ehrenberg, H. R. and Ré, C., 2017b. Snorkel: Fast training set generation for information extraction. In: Proceedings of the 2017 ACM international conference on management of data, pp. 1683–1686.

Spark, A., 2018. Apache spark. Retrieved January 17(1), pp. 2018.

Szegedy, C., Ioffe, S., Vanhoucke, V. and Alemi, A. A., 2017. Inception-v4, inception-resnet and the impact of residual connections on learning. In: Thirty-first AAAI conference on artificial intelligence.

Westra, E., 2015. Python Geospatial Analysis Essentials. Packt Publishing Ltd.

Wu, L., Yang, B. and Jing, P., 2016. Travel mode detection based on gps raw data collected by smartphones: a systematic review of the existing methodologies. Information 7(4), pp. 67.

Yang, F., Yao, Z. and Jin, P. J., 2015. Gps and acceleration data in multimode trip data recognition based on wavelet transform modulus maximum algorithm. Transportation Research Record 2526(1), pp. 90–98.

Zheng, Y., Fu, H., Xie, X., Ma, W.-Y. and Li, Q., 2011. Geolife gps trajectory dataset-user guide. Geolife GPS trajectories 1, pp. 2011.