Anomaly Detection with Independently Collected Human Mobility Data

Sylvan Hoover[a]

[a]*Oregon State University, Corvallis, Oregon, USA*

ARTICLE INFO

Keywords:
human mobility
LSTM

ABSTRACT

Transit agencies struggle to fully utilized their automated passenger counters. Difficulties include resources necessary for maintenance and lack of ongoing assurance of accuracy. Presented is a method to utilize incidentally collected independent mobility data to detect anomalies in primary source data.

1. Introduction and Motivation

Transit agencies collect ridership data to understand how a transit system is utilized. The collected data is analyzed to improve performance, report usage, and other applications where empirical system operation data is needed. Data is currently connected through mode-specific means including automated passenger counters (APC), automated fare collection, and manually implemented surveys. Automated collection increases the sample size agencies are reasonably able to collect but require periodic intervention to validate collected data and apply any necessary calibrations. Validation and calibration of APCs are labor intensive and costly for transit agencies. Having a method to reduce the labor associated with maintaining APCs would save transit agency resources and assure that the system is performing nominally.

To alert agencies to possible anomalous data collection, a secondary source of independently collected data on the mobility of the target population is needed for validation. While temporal patterns in the APC collected data could be used to establish typical usage, such could not account for changes in ridership without a false anomaly indication. By using an independent data source, the anomaly is only identified when the relationship between the two data sources changes. By only intervening when a likely anomaly has occurred, transit agencies can save resources and have confidence that the ridership data being collected is valid.

2. Methodology

Collected mobility data is representative of a human presence at a specific place an time. Sometimes people are grouped as an aggregate count; other times, the data may record by an individual. Regardless, every dataset presents with a place, a time, and a count. With independently collected mobility data, there are two independent places, times, and counts. As each dataset represents a subset of the population, a 1:1 comparison cannot be drawn. Nor is it established that a linear relationship exists between subsets and there are reasons to suspect one would not exist (e.g., increasing interference as density grows may disproportionately affect specific collection methods). What must be established is an anomaly detection method that can train to multivariate time-series data.

Long short-term memory (LSTM) networks are well suited to address the outlined needs. LSTM networks are an evolution of recurrent neural networks suitable for developing an understanding of what is typical, and to facilitate identification when data is atypical or anomalous. Anomaly detection frameworks based on LSTM networks have been developed and applied in a wide variety of application domains, including transportation, medicine, and computer networks. Feng, Yuan and Lu (2016) developed a framework for detecting abnormal events in a surveillance video. In one experiment, the researchers processed the surveillance video of a pedestrian walkway for which expected anomalous events included the presence of a car or a bicycle. Using the proposed LSTM framework, an 11.1% error rate was achieved, outperforming ten other comparative frameworks. Taylor, Leblanc and Japkowicz (2016) employed an LSTM network to learn typical bus command traffic on a controller area network (CAN) bus and alert to anomalies if

hooversy@oregonstate.edu (S. Hoover)
ORCID(s): 0000-0003-1099-5112 (S. Hoover)

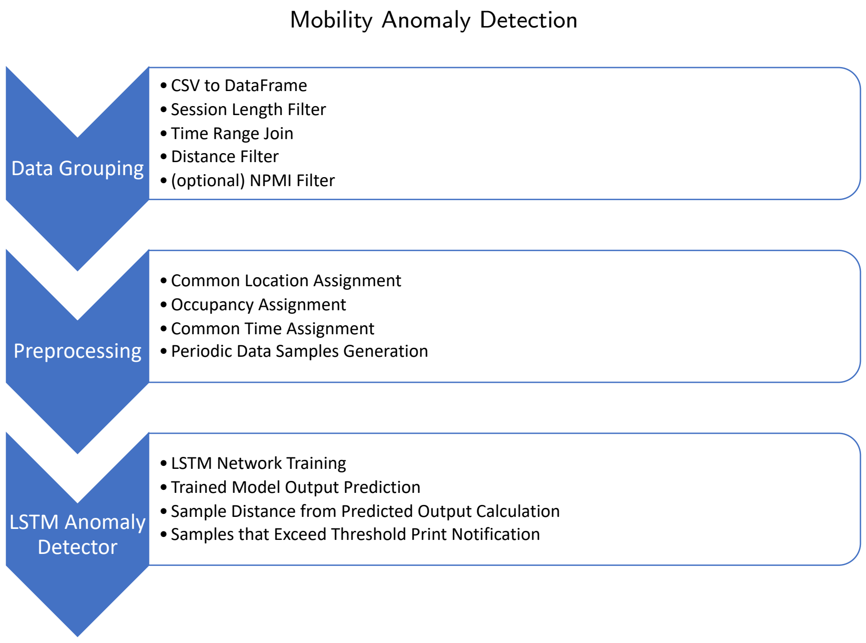

Mobility Anomaly Detection
Data Grouping
• CSV to DataFrame
• Session Length Filter
• Time Range Join
• Distance Filter
• (optional) NPMI Filter
Preprocessing
• Common Location Assignment
• Occupancy Assignment
• Common Time Assignment
• Periodic Data Samples Generation
LSTM Anomaly Detector
• LSTM Network Training
• Trained Model Output Prediction
• Sample Distance from Predicted Output Calculation
• Samples that Exceed Threshold Print Notification

Figure 1: The data flow for anomaly detection.

unexpected data appeared. Due to the nature of the interactions between the operator and the vehicle, the focus was on achieving a false positive rate (FPR) level that would prevent operator fatigue (i.e., causing vehicle operators to ignore alerts thus nullifying the purpose of the system). Performance varied across the devices on the CAN bus highlighting the variable effectivity based on device behavior and demonstrating the challenge of limiting FPR while maintaining utility as measured by the true positive rate (TPR). This research also explored what would be required to achieve a 100% TPR, which resulted in some devices having as high as a 63.4% FPR (a rate likely to lead to operator fatigue). It is evident that the anomaly detection performance of an LSTM network cannot be a simple determination but relies heavily upon both the nature of the typical data and the atypical anomaly.

Mobility data is a good candidate for LSTM anomaly detection methods. The data possess temporal patterns (e.g., peak/off-peak and seasonality), are not prone to high variability, and common equipment anomalies are persistent and not sporadic. As transit agencies are unlikely to know the onset of anomalous data and are likely not to have used the second data source before, it is necessary that the LSTM training occurs unsupervised (or self-supervised). The LSTM is trained with a time-series of input features resulting in an encoded output, in the case of an autoencoder, one of the input features. This allows comparison between the predicted output over a period and the recorded data. Sample periods where the recorded data deviate beyond a defined threshold from the model are identified as anomalous.

3. Implementation

This package draws mainly from the work of prior packages that facilitate the use of OpenStreetMap and machine learning. For OpenStreetMap, OSMnx (Boeing, 2017) provides a gateway between the online dataset and local network representation. For machine learning, Keras is employed as a high-level API using TensorFlow as the backend.

3.1. Modules

OSM-Multiplex (OSMmp) (Hoover, 2019) includes four modules. The first, osm_download.py, serves to interface with OpenStreetMap. Figure 1 show the data flow of the other three, count_data.py, lstm_preprocessing.py, and lstm.py, which constitute the ridership data anomaly detection process.

3.1.1. OpenStreetMap Network Generation - osm_download.py

OpenStreetMap (OSM) (OpenStreetMap contributors, 2017) serves as a potential base network for multi-modal transportation to be built. The package OSMnx serves to fetch OSM geographic selections and returns a NetworkX multidigraph. Two types of geographic selections are possible, an already geocoded place name or a coordinate bound-

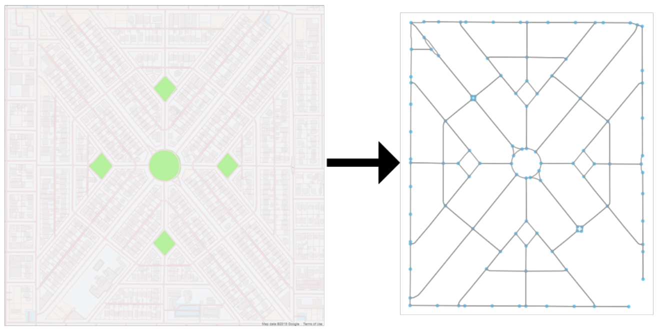


Figure 2: The evolution from raster map to automobile network representation.

ing box. OSMnx fetches a network layer only for single modes. To build a multi-modal multiplex network, layers for each mode are fetched with OSMnx, nodes in common between layers are identified, and edges are added connecting common nodes to allow mobility between mode layers. Figure 2 shows the results of creating a transportation network representative of a neighborhood.

3.1.2. Data Grouping - count_data.py

Independent datasets to be used for comparison need to be filtered and joined to identify records likely to represent the same mobility event.

First, it is required that all data be imported be presented in CSV format. This is due to the current importation method into a pandas DataFrame. As pandas allows for various import methods, the package could later be adapted to import other file formats.

Next, for any records with session lengths (e.g., WiFi or Cellular) sessions that indicate a static presence (i.e., session length longer than what is needed to transit the coverage cell) are removed.

Once the records are filtered to indicate movement, a range join is executed between the mode-specific and the mode-agnostic datasets. This identifies records that may represent the same individual.

Finally, a Haversine distance filter is applied to select only sessions that occur with a defined geographic range of each other. The time range join and the distance filter could be swapped, but that would require the Haversine distance be calculated between every record pair, so it was determined to be more efficient to join based on session initiation time and filter based on distance based on the pairs created from that operation.

An additional filter may be optionally applied. The Normalized Pointwise Mutual Information (NPMI) filter can be used when both datasets have individual identifiers. It looks at the ‘corpus‘ of all paired records and calculates a metric to identify the dependence of identifiers between datasets. The aim is to filter out identifier pairs where the identifiers occur more often with other identifiers than they do with each other. The metric is primarily employed in computational linguistics to identify words that regularly only occur with the accompaniment of another (e.g., "puerto" and "rico") in a corpus.

3.1.3. Preprocessing - lstm_preprocessing.py

Independent datasets may not represent the same point of collection, the same period, or similar sampling rates. To facilitate comparison, it is necessary to reconcile the spatial and temporal states of the datasets to be compared.

First, the spatial assignment is reconciled. Locations in the processed dataset are reconciled around either the locations in either dataset or the node nearest to the mean of the locations of paired records in the data.

Next, for every record, an occupancy level is determined. For individually identified data, the occupancy is fixed to 1. For grouped data with boardings and alightings (e.g., bus), a daily cumulative sum by vehicle identifier is calculated. As is often noted with APCs, biases exist in the data, so there is regularly a non-zero occupancy at the end of the day for a given vehicle. There are post-processing methods that some employ to improve data quality, but at this time, they are not implemented.

Finally, time is reconciled and formatted to facilitate LSTM network processing. A single time identifier is determined for each record; it can be either the time for a record from either dataset or the mean of the two times. The data is then pivoted and re-sampled at a defined interval. Currently, the package is configured to identify weekly data samples with 15-minute intervals. This configuration provides a relatively fine-grained time-series of weekly activity collected by each data source. It is not well suited for seasonal variations, and such accommodation will likely appear in later versions.

3.1.4. LSTM Anomaly Detector - lstm.py

For the LSTM-based anomaly detection, the core structure was borrowed from Chen (2019). The structure of this anomaly detector is as an LSTM auto-encoder employing the Keras API (Chollet et al., 2015) with the TensorFlow (2015) backend. This module trains a two-layer neural network with the first layer being an LSTM layer and the second being a fully-connected layer the size of the number of time steps in each sample. The default training/validation split is 80/20 requiring at least 5 sample weeks for processing. The LSTM layer takes an input of three features (dataset 1 count, dataset 2 count, and the difference between the two) and trains with a target output as the difference. This structure accounts for time-series patterns that exist in both the data sources and trains to replicate the difference that exists between the two sources samples.

An anomalous week is identified by producing a week's count difference by predicting from the trained network. The difference at every sample point is then calculated between the predicted source difference and the sample observed source difference. The norm of the resulting matrix is calculated, and any weeks where the norm is greater than a standard deviation multiplier (default multiplier is 1.5) from the average is identified as anomalous.

3.2. Installation

While there is a conda package available (hooversy/osm-multiplex), there still exists unresolved discrepancies between the package installation through conda and running from source. The OSMnx package has documented difficulties with using pip for installation. If not employing any elements that require OSMnx (e.g., anomaly detection where location does not coalesce around OSM nodes), then pip is a viable installation route. Installation of the package using conda also functions, but there are still existing issues with the conda packaging. The most reliable installation option at this point is to execute from source with the packages listed in setup.py installed via conda.

3.3. Execution

Primary interaction with OSMmp is through command line execution and arguments. With the two current uses of OSMmp, there are two distinct input/output flows.

3.3.1. OpenStreetMap Multiplex Network Generation

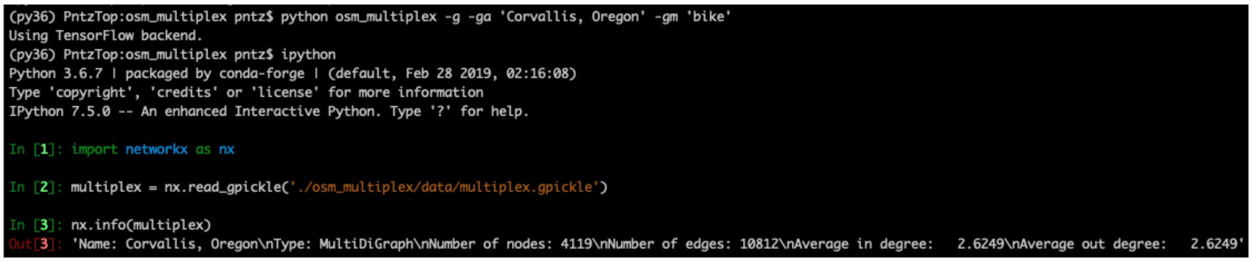
To generate an OSM multiplex graph for an OSM geocoded place from command line use '-g' to indicate graph generation, '-ga' to specify graph area, and '-gm' to specify graph modes. For example, if seeking the bike network for Corvallis, Oregon:

```
python osm_multiplex -g -ga 'Corvallis, Oregon' -gm 'bike'
```

3.3.2. Ridership Data Anomaly Detection

To detect anomalies in independently sourced human mobility data from the command line, specify the datasets and relevant fields. The anomalous weeks are printed on the as the location is processed:

```
python osm_multiplex -ad
-d1 './osm_multiplex/data/dataset1.csv'
-e1 'tagID' -ts1 'timestamp'
-lat1 'lat' -lon1 'lon'
-d2 './osm_multiplex/data/dataset2.csv'
-e2 'PIN' -ss2 'SessionStart_Epoch' -se2 'SessionEnd_Epoch'
-lat2 'GPS_LAT' -lon2 'GPS_LONG'
```



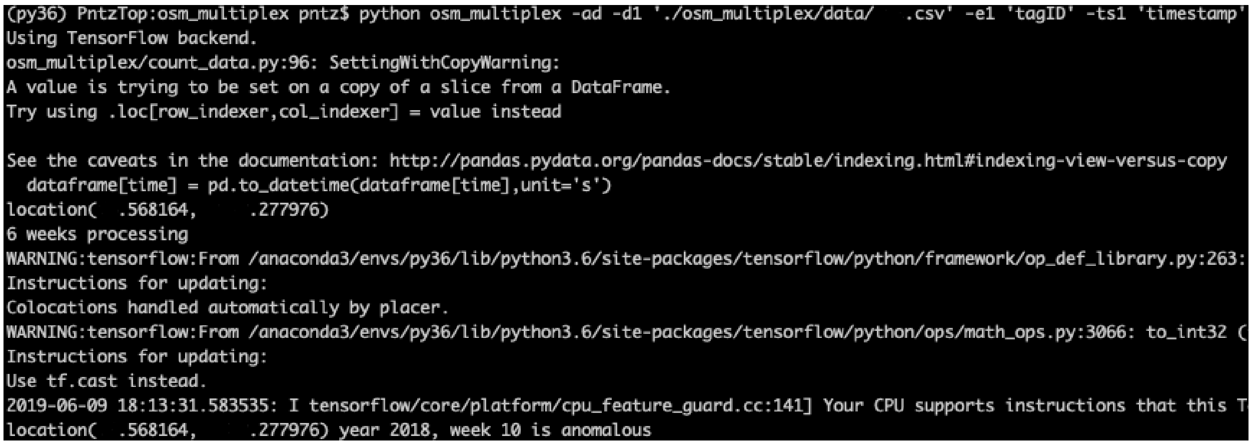
```
(py36) PntzTop:osm_multiplex pntz$ python osm_multiplex -g -ga 'Corvallis, Oregon' -gm 'bike'
Using TensorFlow backend.
(py36) PntzTop:osm_multiplex pntz$ ipython
Python 3.6.7 | packaged by conda-forge | (default, Feb 28 2019, 02:16:08)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.5.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import networkx as nx

In [2]: multiplex = nx.read_gpickle('./osm_multiplex/data/multiplex.gpickle')

In [3]: nx.info(multiplex)
Out[3]: 'Name: Corvallis, Oregon\nType: MultiDiGraph\nNumber of nodes: 4119\nNumber of edges: 10812\nAverage in degree:   2.6249\nAverage out degree:   2.6249'
```

Figure 3: Execution and resulting network for Corvallis, Oregon bike transportation network.



```
(py36) PntzTop:osm_multiplex pntz$ python osm_multiplex -ad -d1 './osm_multiplex/data/    .csv' -e1 'tagID' -ts1 'timestamp'
Using TensorFlow backend.
osm_multiplex/count_data.py:96: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  dataframe[time] = pd.to_datetime(dataframe[time],unit='s')
location(  .568164,      .277976)
6 weeks processing
WARNING:tensorflow:From /anaconda3/envs/py36/lib/python3.6/site-packages/tensorflow/python/framework/op_def_library.py:263:
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From /anaconda3/envs/py36/lib/python3.6/site-packages/tensorflow/python/ops/math_ops.py:3066: to_int32 (
Instructions for updating:
Use tf.cast instead.
2019-06-09 18:13:31.583535: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that this T
location(  .568164,      .277976) year 2018, week 10 is anomalous
```

Figure 4: Execution and resulting detected anomalies for real-world dataset. (Image cropped for readability)

4. Results

Results are demonstrated through the successful execution of a data pipeline. Future development of employing synthetic datasets with induced anomalies will better serve to show the effectivity of OSMmp.

4.1. OpenStreetMap Multiplex Network Generation

Figure 3 demonstrates the execution of the multiplex network generation and summarizes the resulting output. The resulting average degree measures are an indication of the connectedness of the bicycle transportation network. Plotting reveals a similar network at to that present in Figure 2, but its scale does not lend it well to visualizing in this context.

4.2. Ridership Data Anomaly Detection

Figure 4 demonstrates the execution and results of mobility data anomaly detection on a real-world dataset. Due to the proprietary data currently used for testing, some of the fields are obfuscated. The datasets held approximately 1.8 million records and 90 thousand records, respectively, and the anomaly detection process took over 2 hours. The long processing time is due to a single step in the process, a range join, which currently employs SQLite. SQLite is used because it allows processing in disk storage as other tried methods that would operate faster exceeded memory capacity (64GB) of available systems for testing. Optimization of the range join would likely reduce processing time from a couple hours to a couple minutes.

5. Conclusions and Future Work

Transit agencies continue to experience pressure to do more with less. Ensuring reliable APC counts without added capital expenditures both facilitates the optimization of delivered services and reduces maintenance costs. The creation of an easily implemented software package to alert agencies to anomalous APC data explores the potential of agencies using available open source software to add value to their existing analyses.

Challenges remain in any potential fielding of this software or the methods contained within. Transit agencies are slow to adopt new technologies and often rely on outside contractors to deliver technical assistance. Developing independent sources of mobility data sampling from the transit population will also prove a significant hurdle as often the data is privately held and not openly discussed. The development of public or public-private partnership wireless networks is one avenue already being utilized to generate potentially useful datasets.

While the software currently accounts for the evolving nature of a time-series, it does not utilize the network nature of transportation systems. Each node is analyzed independently of neighboring nodes. This concentrates and simplifies the anomaly detection, but does not take advantage of the relationships that potentially exist between nodes. Future work integrating network analysis will be necessary to develop a holistic view of multi-modal transportation and evolve the understanding of how modes interact.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X., 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. URL: https://www.tensorflow.org/. software available from tensorflow.org.

Boeing, G., 2017. OSMnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks. Computers, Environment and Urban Systems 65, 126–139. URL: https://linkinghub.elsevier.com/retrieve/pii/S0198971516303970, doi:10.1016/j.compenvurbsys.2017.05.004.

Chen, X., 2019. keras-anomaly-detection. URL: https://zenodo.org/record/3243650, doi:10.5281/zenodo.3243650.

Chollet, F., et al., 2015. Keras. https://keras.io.

Feng, Y., Yuan, Y., Lu, X., 2016. Deep Representation for Abnormal Event Detection in Crowded Scenes, in: Proceedings of the 2016 ACM on Multimedia Conference - MM '16, ACM Press, Amsterdam, The Netherlands. pp. 591–595. URL: http://dl.acm.org/citation.cfm?doid=2964284.2967290, doi:10.1145/2964284.2967290.

Hoover, S., 2019. hooversy/osm_multiplex: ME599 Spring 2019 Submission. URL: https://zenodo.org/record/3242155, doi:10.5281/zenodo.3242155.

OpenStreetMap contributors, 2017. Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org.

Taylor, A., Leblanc, S., Japkowicz, N., 2016. Anomaly Detection in Automobile Control Network Data with Long Short-Term Memory Networks, in: 2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA), IEEE, Montreal, QC, Canada. pp. 130–139. URL: http://ieeexplore.ieee.org/document/7796898/, doi:10.1109/DSAA.2016.20.