AAAI-20 Workshop on Artificial Intelligence of Things

In conjunction with the 34th AAAI Conference on Artificial Intelligence

February 7th, 2020 New York – New York, USA

APPLYING WEAK SUPERVISION TO MOBILE SENSOR DATA: EXPERIENCES WITH TRANSPORT MODE DETECTION

JONATHAN FÜRST[1], **MAURICIO FADEL ARGERICH**[1], KALYANARAMAN SHANKARI[2], GÜRKAN SOLMAZ[1], **BIN CHENG**[1]

JONATHAN.FUERST@NECLAB.EU, MAURICIO.FADEL@NECLAB.EU, BIN.CHENG@NECLAB.EU

[1]NEC LABS EUROPE, [2]UC BERKELEY

AGENDA

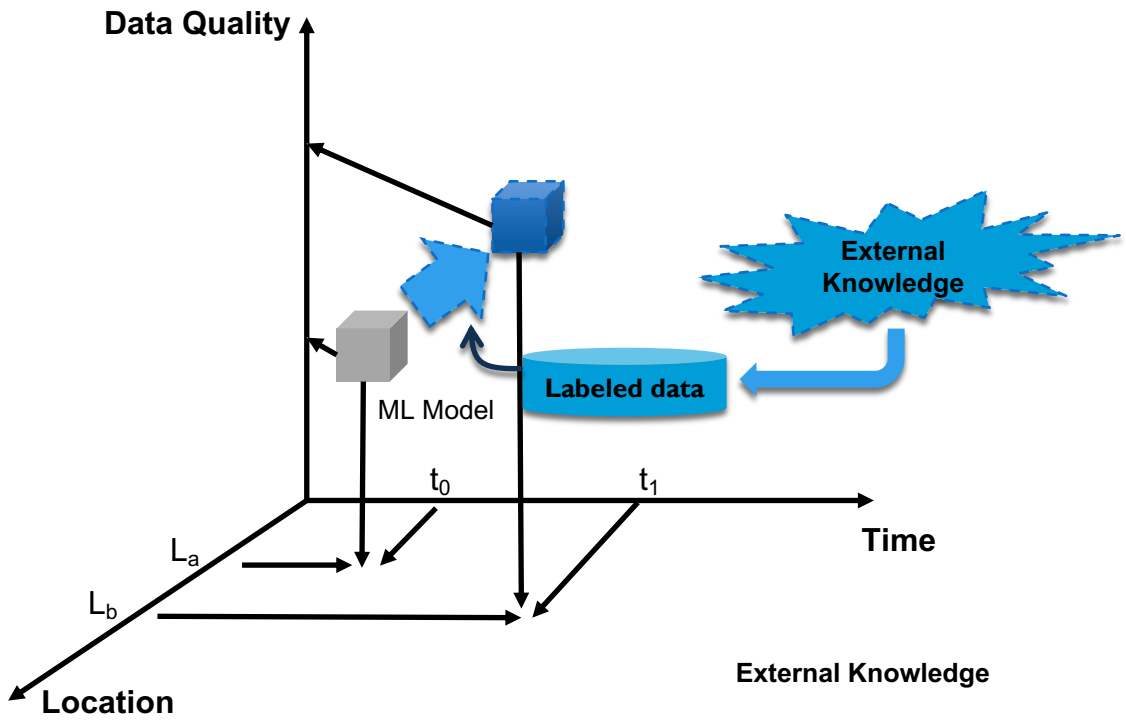
- ML in IoT
- Our domain
- Transport Mode Detection
- Weak Supervision for Transport Mode Detection
- Evaluation & Results
- Takeaways & Future work

ML IN IOT

- IoT is expanding to new domains
- ML is essential to exploit the power of IoT
- Challenges
  - Location
  - Time
  - Data Quality

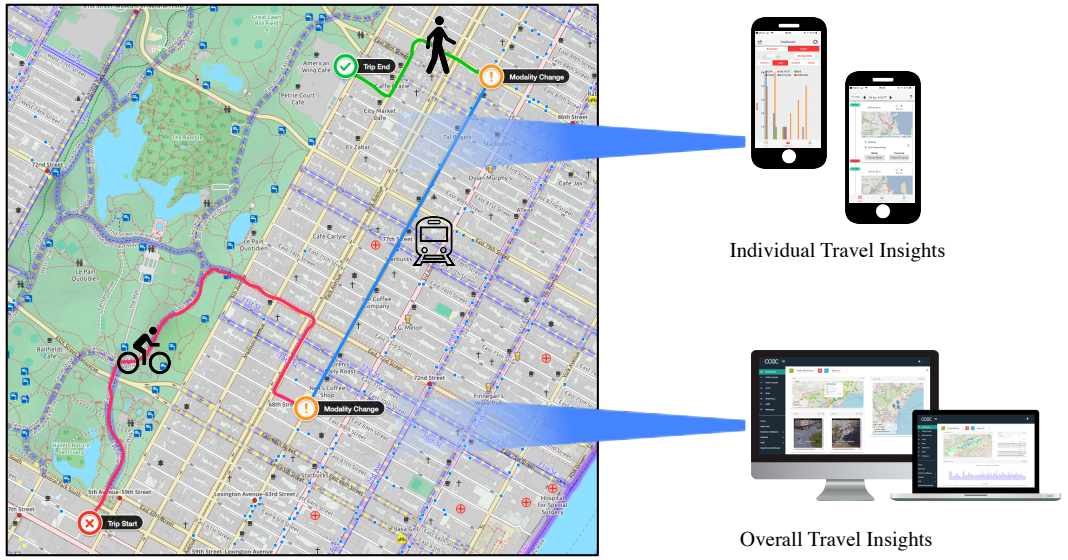
- Labeled data is very expensive

But, we can label data using noisy programmable functions that express external knowledge, and then re-train our model



OUR DOMAIN: TRANSPORT

- The city of Heidelberg wants to improve public transportation
- They need insights about how people move in the city
- Our solution was to create a mobile app
  - Citizens get transport recommendations
  - City gets an aggregated view of transportation
- We need to know
  - Location of users (start, trajectory and end point) → TGGPS
  - Transport mode of user → Manually labeled? 😵
    → Can we infer it? 🤔

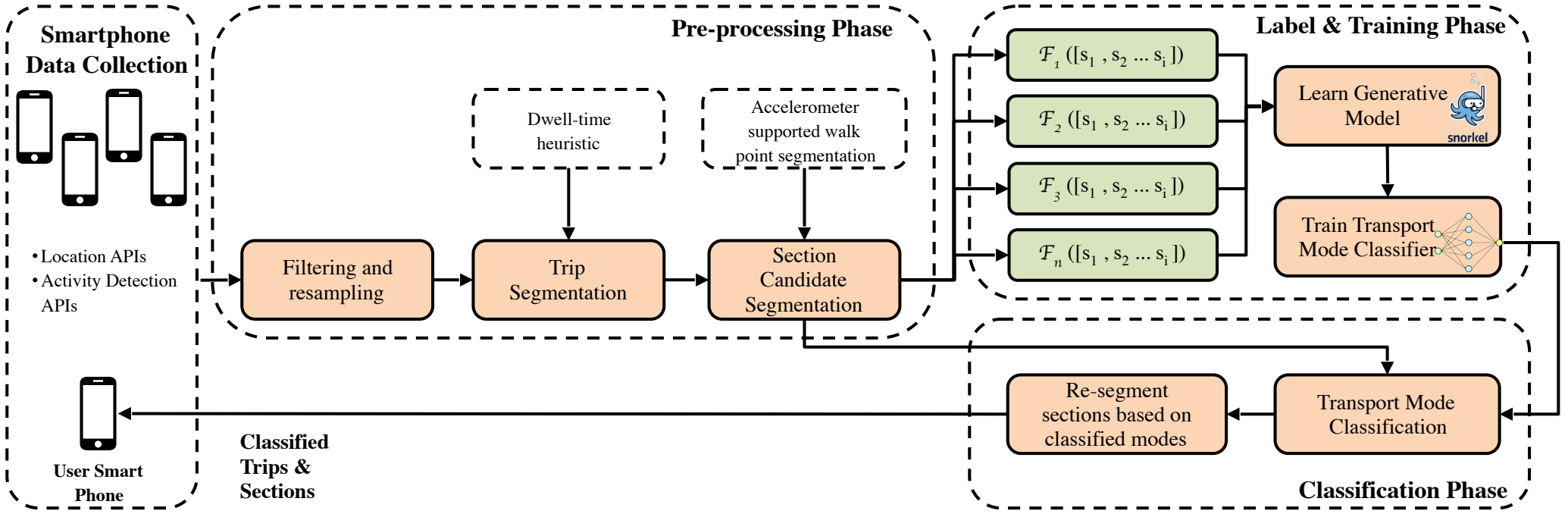


TRANSPORT MODE DETECTION

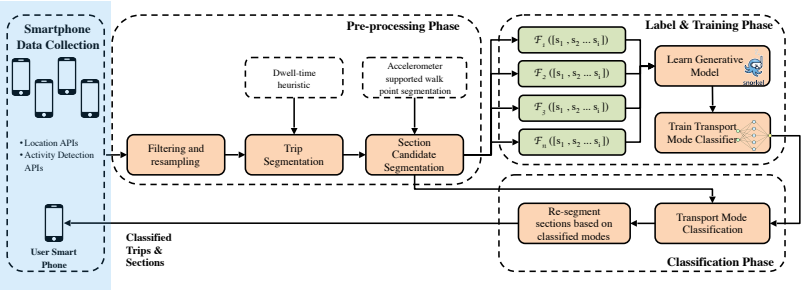
- Transport mode detection is fundamental to optimize urban multimodal human mobility
- It requires two steps:
  1. Segmentation
  2. Classification
- Current studies have used GPS, accelerometer, barometer and GIS data to train supervised ML models
  - Data has to be labeled manually → Data is labeled semi-automatically
  - Training sets are small (guess why) and then model is overfitted → Training set can be much larger, less overfitting
  - Data quality vs. battery and OS limitations → The more the data, the less data quality needed

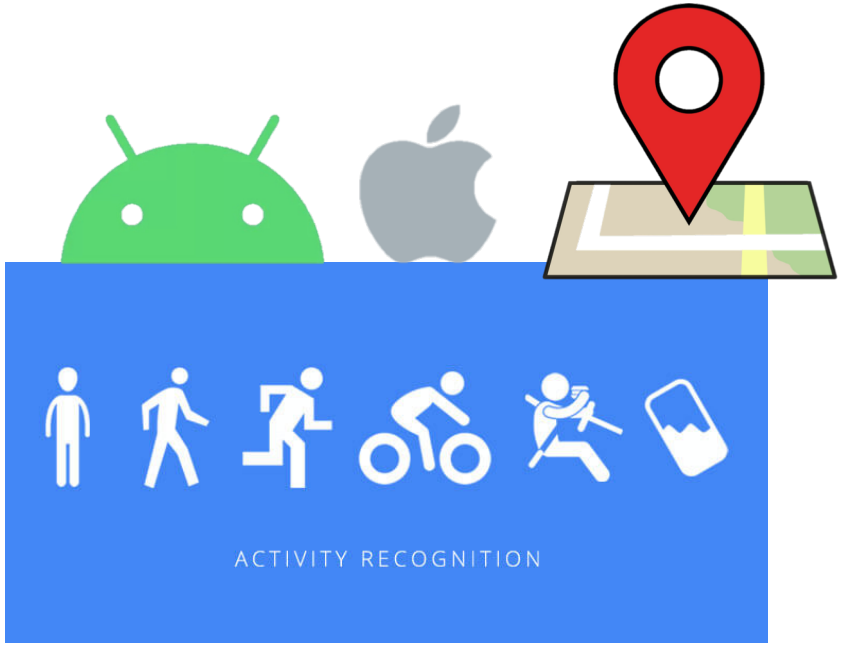
Our take: improve data availability using weak supervision

WEAK SUPERVISION FOR TRANSPORT MODE DETECTION



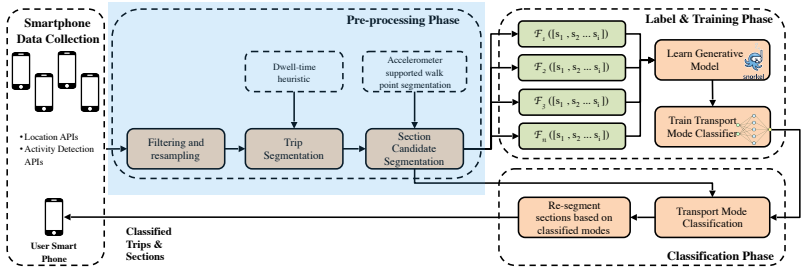
MOBILE SENSOR DATA COLLECTION



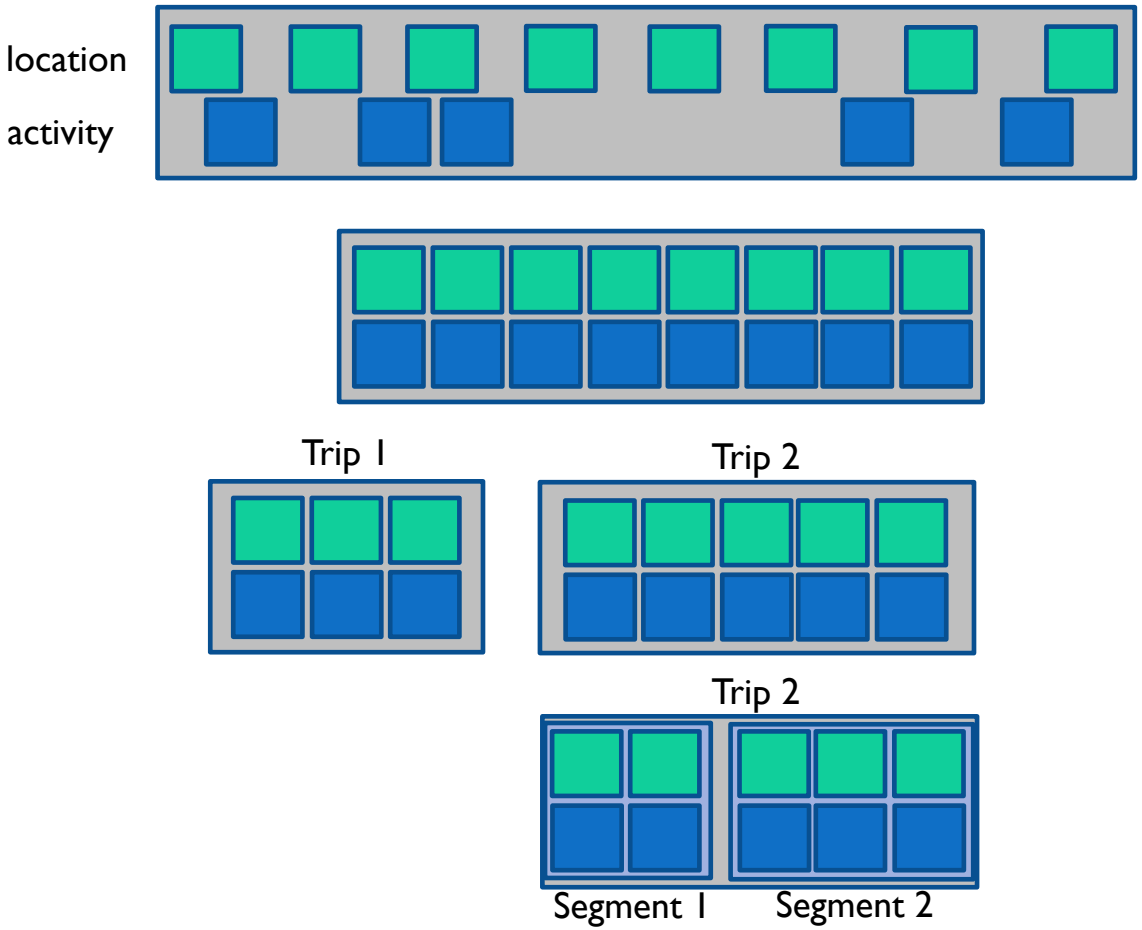


- Collecting data from mobile sensors drain a lot of battery
  - Sensing location using GPS
  - Accelerometer and barometer → high frequency
- Instead, we use Android and iOS native APIs (Location and Activity)
  - Highly optimized for battery consumption
  - BUT, sparse and noisy sensor data

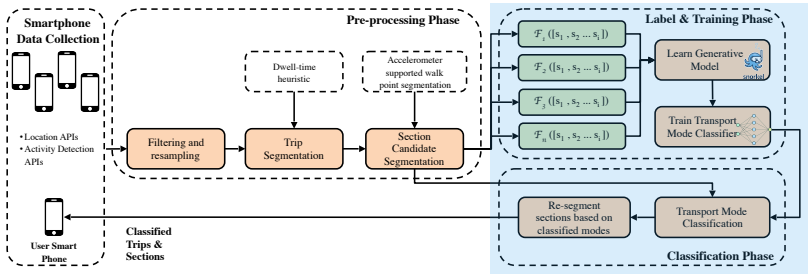
TIME SERIES SEGMENTATION



1. Filter and re-sample data
   - Sparsity
   - Location and activity data are not aligned
   - No fixed sampled interval
2. Segment time series into Trips
   - Dwell time heuristics
3. Segment Trips into Segments
   - Walk-point-based



LABELING, TRAINING AND CLASSIFICATION



- We use Data Programming (Ratner et al. 2017)
- Labeling functions
  - Programmable functions
  - Use external knowledge
  - Cast a (noisy) vote on each data point
- Votes create a Labeling Matrix (LM)
- LM + lab. propensity + accuracy + correlation = Generative Model
- We label data points with generative model and use data to train an end model

"if the maximum speed of a segment is less than 3 m/s, then it's probably a walking segment"
"instead, if it's higher than 3 m/s but less than 10 m/s, then it's probably a bike segment"
...

```
def max_velocity(self, mv):
    if mv < 3:
        return WALKING
    elif mv < 10.0:
        return BIKE
    elif mv < 48.44:
        return CAR
    else:
        return TRAIN
```

$$\begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \\ 1 & -1 & -1 \end{bmatrix}$$

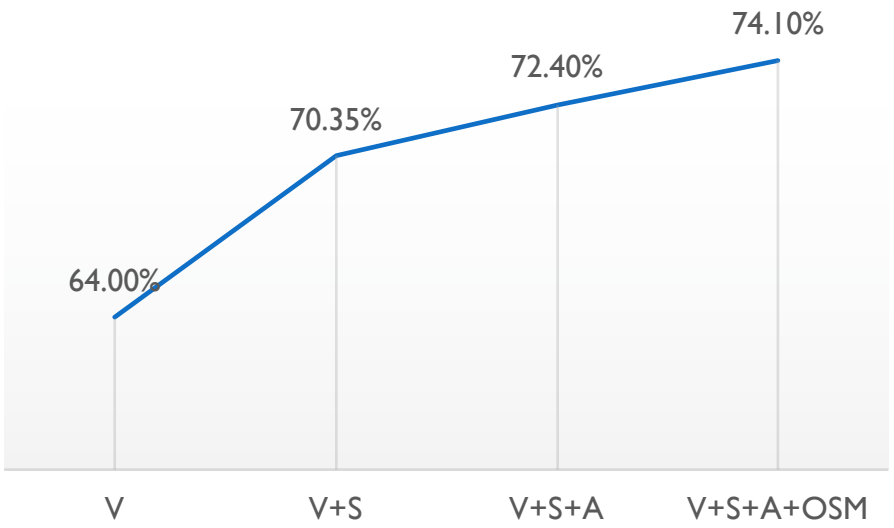
data point 1
data point 2
data point 3
data point 4

lab. function 1
lab. function 2
lab. function 3

EVALUATION & RESULTS (1)

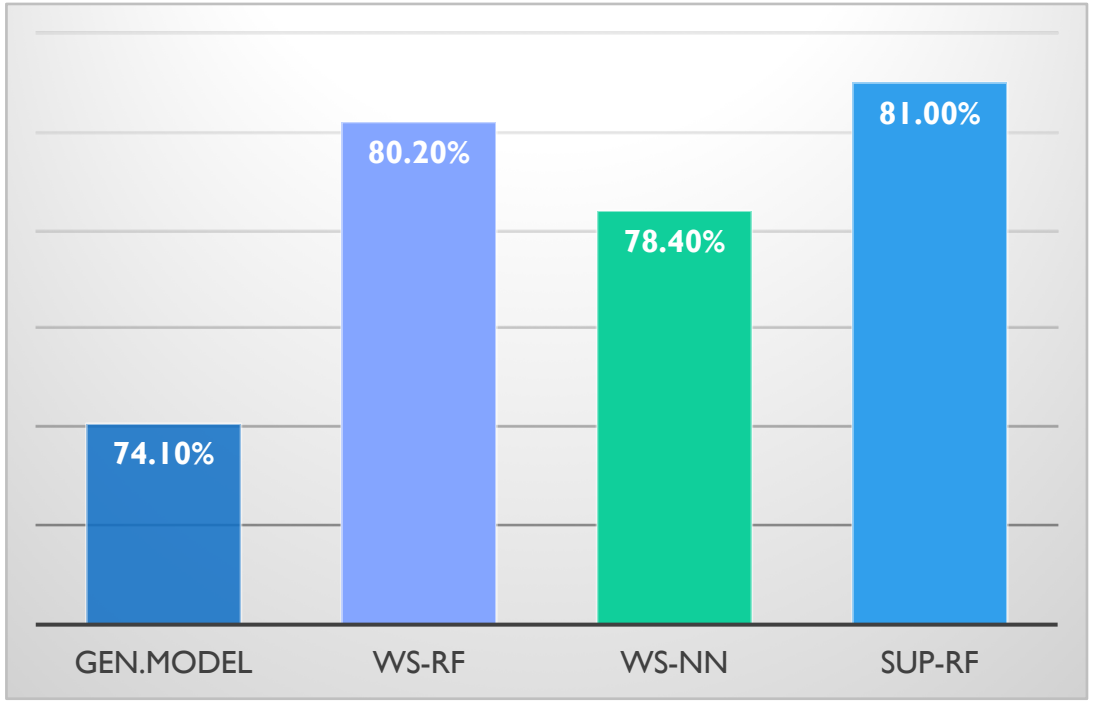
- Our data
  - 8 users collected data for 4 months: 300k data points
  - Features
    - GPS location (through iOS and Android Location API)
    - Accelerometer based activity data (through Activity API)
  - Users partially labeled data using a visual labeling tool
  - 4 transport modes: walk, bike, car, train
  - Train/test split: 50/50
- We implemented 7 labeling functions using
  - Sensed speed
  - Velocity (calculated with GPS)
  - OpenStreetMaps (to check train stops)
- We tested the Generative Model accuracy with different sets of labeling functions

	Labeling Function	Classes	Coverage	Accuracy
(V)	LF_max_velocity	[0, 1, 2, 3]	100 %	72.1 %
(V)	LF_median_velocity	[0, 1, 2, 3]	99.6 %	68.5 %
(A)	LF_motion_activity	[0, 1, 2]	82.9 %	80.5 %
(V)	LF_std_velocity	[0, 1, 2, 3]	100 %	46.8 %
(OSM)	LF_osm	[3]	10.4 %	37.0 %
(S)	LF_median_sensed_speed	[0, 1, 2, 3]	100 %	73.6 %
(S)	LF_quantile_sensed_speed	[0]	65.7 %	78.3 %



EVALUATION & RESULTS (2)

- We label all the train data using the generative model and train a Random Forest and a Neural Network
- We also train a Random Forest using the manually labeled data from users



LESSONS LEARNT & FUTURE WORK

- Extensive manually labeling is not necessary for IoT data if we use external knowledge
  - Domain/Expert knowledge
  - Physical knowledge
- Access to external knowledge is not always easy
- Granularity in which IoT series should be labeled

- We will gather more data to continue the evaluation of our application in Heidelberg
- We will evaluate our approach with data from other cities, to test the generalizability

Thank you!

mauricio.fadel@neclab.eu
bin.cheng@neclab.eu
jonathan.fuerst@neclab.eu