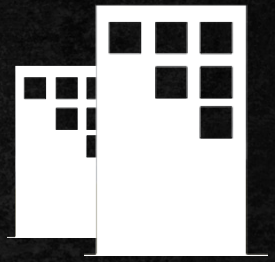
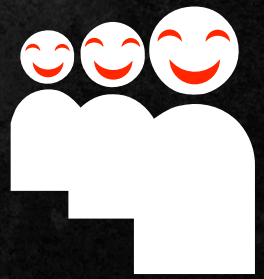
Sensor-based Positioning System that Exploits Spatio-Temporal Regularity of Human Mobility

Takuya Watanabe†, Mitsuaki Akiyama‡ and Tatsuya Mori†
† Waseda University ‡ NTT Secure Platform Labs

Your location



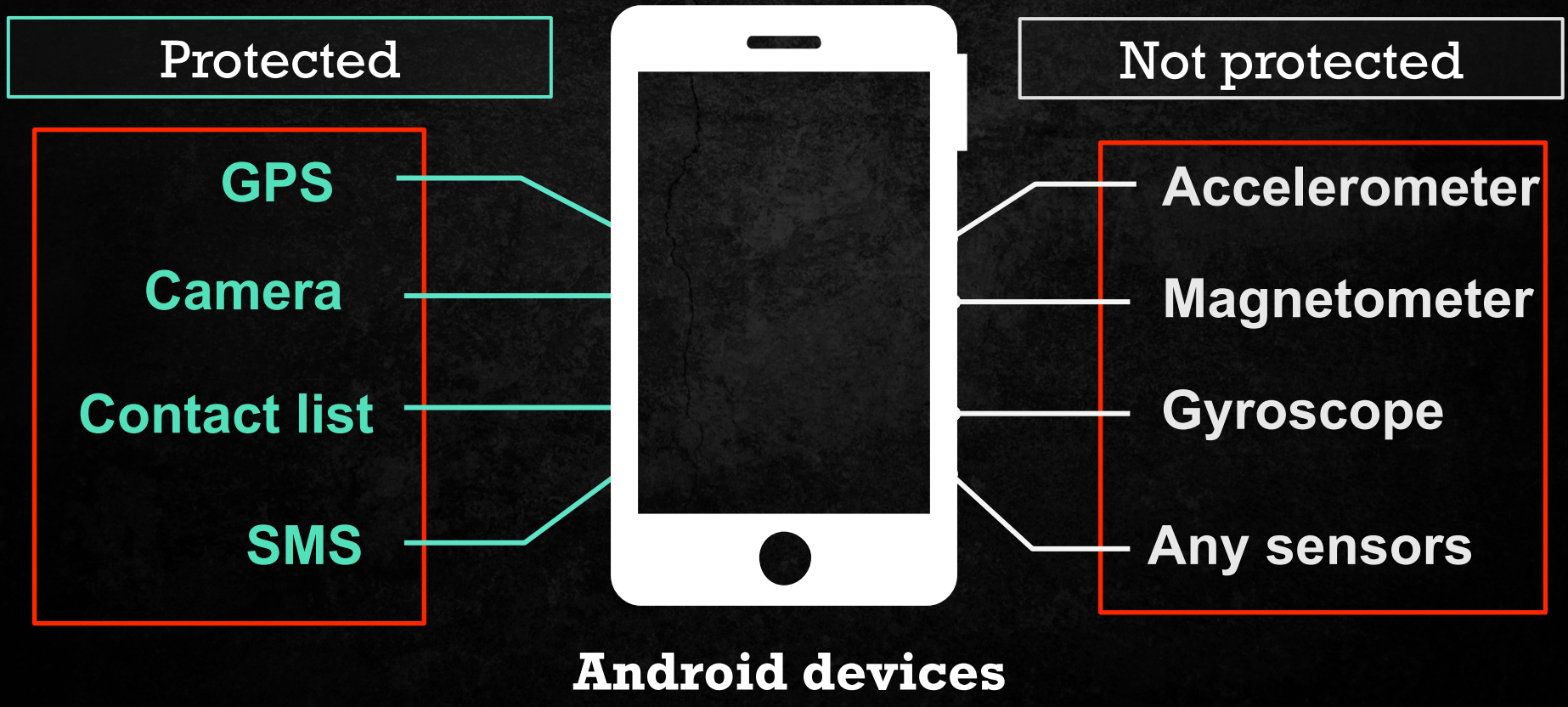
Unsolicited marketing



A stalker

A boss

The privacy protection mechanism



The ultimate goal

Sensor data

Accelerometer
Magnetometer
Gyroscope

data processing

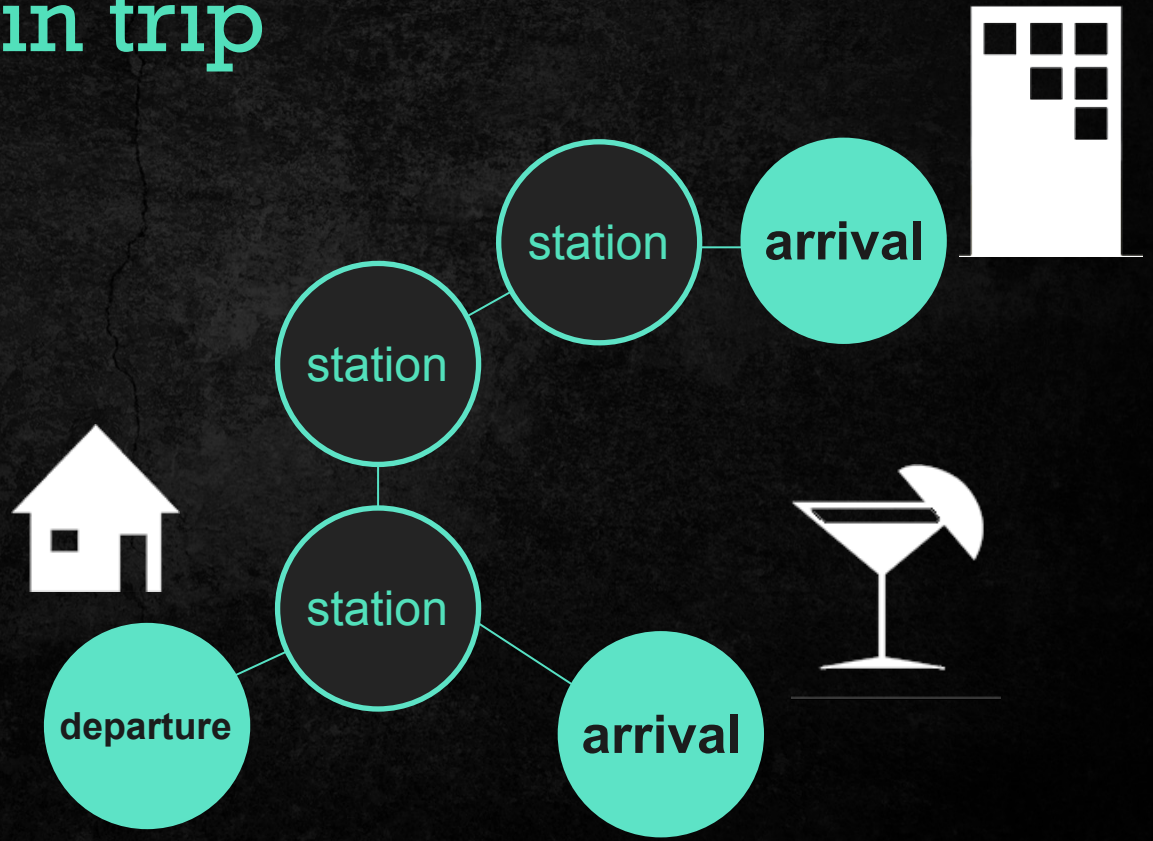
Identify absolute position

It is difficult

Then, we focus…

Features of train trip

- Static route
- Regularly
- Associated with
  - place of residence
  - workplace
  - favorite bar

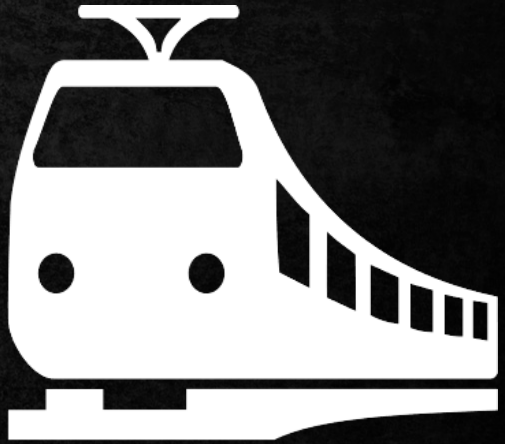


Our goal in this work

Sensor data

Accelerometer
Magnetometer
Gyroscope

Route Detector

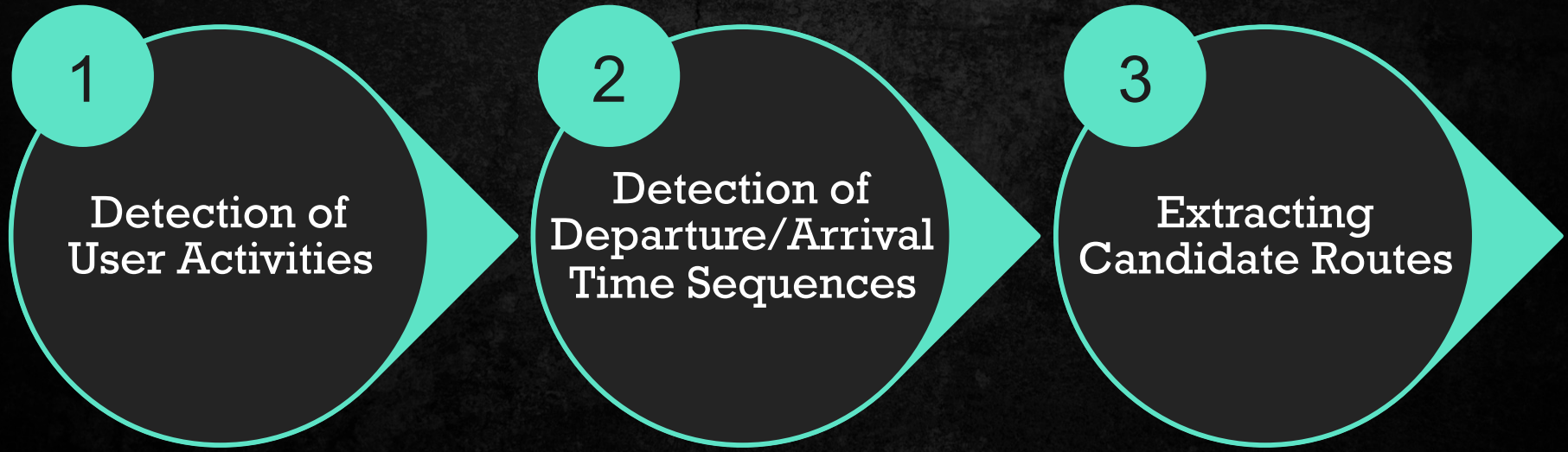


Detect the route of a train trip

Threat model

- A malicious software on a smart device
  - Only internet permission is required
- This software secretly keeps collecting sensor values
  - It estimates the owners activity (walk, on a vehicle and still)
- An adversary knows the list of public transportation systems that are used by the victim.

Overview of RouteDetector



Smart device

Sensor data

User activity

Time sequence

13:00,
13:05,
13:09

Candidate

Tokyo
↓
Osaka

A malicious app

Machine learning

Real time information

Timetable

Detection of User Activities

Sensor data

Accelerometer
Magnetometer
Gyroscope

Machine Learning
(Random Forest)

User activity

Class 0
Still

Class 1
On a vehicle

Class 2
Walk

cf: Android API to recognize activity

RouteDetector
User activities detection

No permission required

Android official API
ActivityRecognitionApi

Permission required

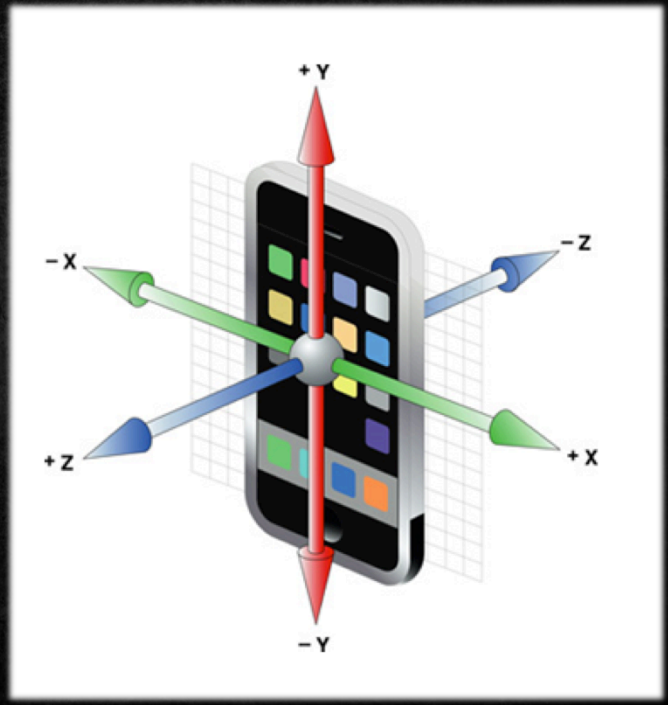
Summary of sensors

	Unit	Type	Permission	Description
Accelerometer	$m/s\hat{}2$	Physical	None	Acceleration applied to a device
Linear accelerometer	$m/s\hat{}2$	Virtual	None	Acceleration applied to a device excluding the gravity
Magnetometer	μT	Physical	None	Strength of geomagnetic field
Gyroscope	rad/s	Physcal	None	A device's rate of rotation

10 Hz : read 10 values per second

Data preprocessing for each sensor

1. Compute norm $=\sqrt{a\downarrow x\uparrow 2 + a\downarrow y\uparrow 2 + a\downarrow z\uparrow 2}$
   - To eliminate the effect of differences in the directions



Data preprocessing for each sensor

2. Divide time series data into a set of blocks

$$a_1, a_2, a_3, \ldots, a_n$$

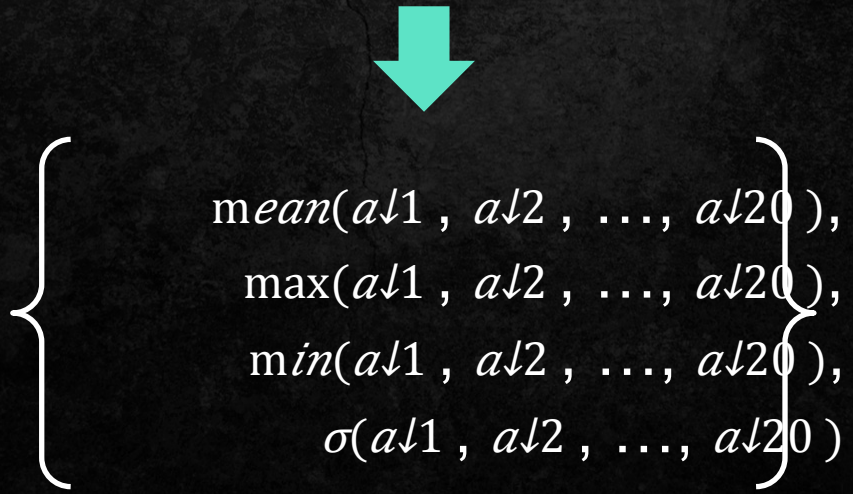
$$\{a_1, a_2, \ldots, a_{20}\}, \{a_{21}, a_{22}, \ldots, a_{40}\}, \ldots$$

block1 block2

Data preprocessing for each sensor

3. Calculate typical values: mean, max, min and **σ**

$$\text{Block1} = \{a_1, a_2, \ldots, a_{20}\}$$



Feature vectors to detect activity

{ Accelerometer
Linear acceleration
Magnetometer
Gyroscope } * { mean
max
min
σ } = 16 dimensions

=> A random forest classifier (supervised ML)

Labels creation

HTC / Smartphone

Nexus 7 / Tablet

Labels stats (# of blocks)

Data	vehicle	walk	still	total	total time
HTC_hold	1,327	510	609	2,446	4,892 sec
HTC_bag	1,360	510	691	2,561	5,122 sec
Nexus_hold	1,352	505	686	2,543	5,086 sec
Nexus_bag	1,304	505	602	2,411	4,822 sec

HTC/Nexus … devices name hold/bag … situation

Definition of FP/FN

Ground truth

Class 0 or 2
Still or Walk

Class 1
Vehicle

Prediction

Class 1
Vehicle

Class 0 or 2
Still or Walk

False Positive

False Negative

Evaluation of Activities Detection

Data	ACC	FNR	FPR
HTC_hold	**0.941**	0.042	0.078
HTC_bag	**0.965**	0.024	0.047
Nexus_hold	**0.943**	0.041	0.074
Nexus_bag	**0.969**	0.023	0.041

Performance of detecting vehicle activity

Example of the classification

0: still , 1: On a moving vehicle , 2:Walk

Ground truth:

22211111111111111111110000000000111111111111111

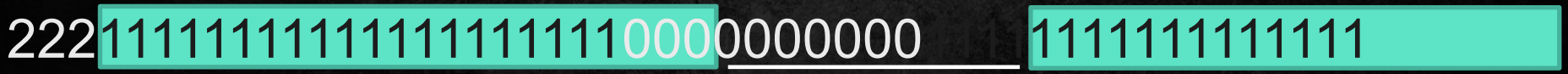
Prediction:

22211001111111011111111110010000011111011111111111

Example of the classification

0: still , 1: On a moving vehicle , 2:Walk

Ground truth:

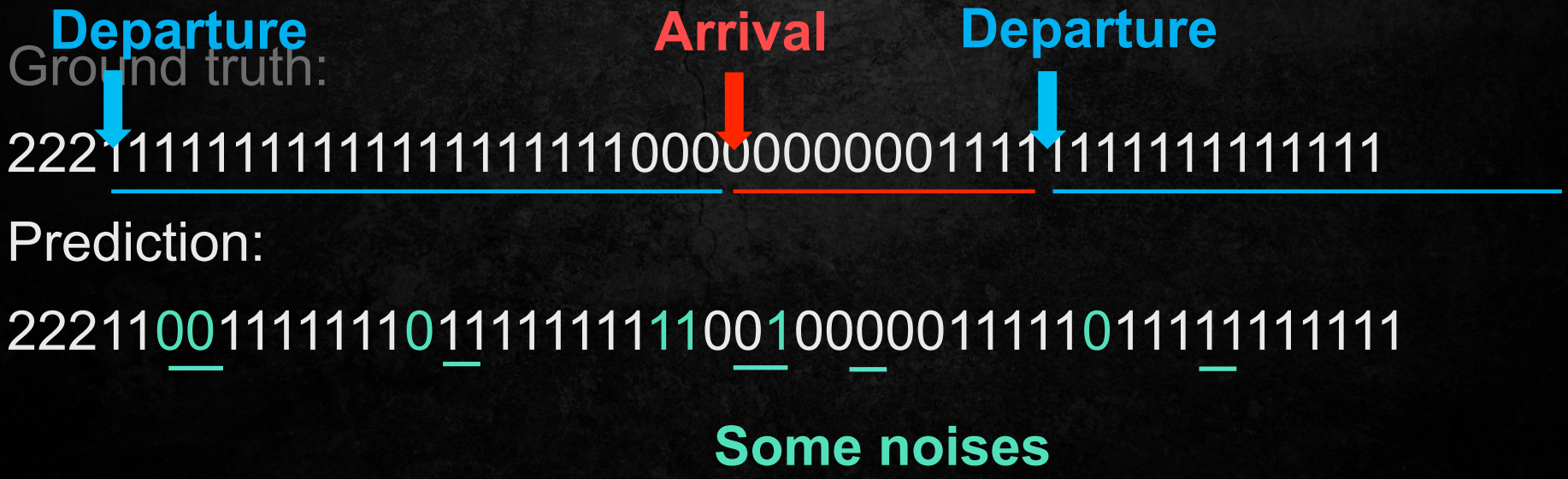


On the still train

On the moving train

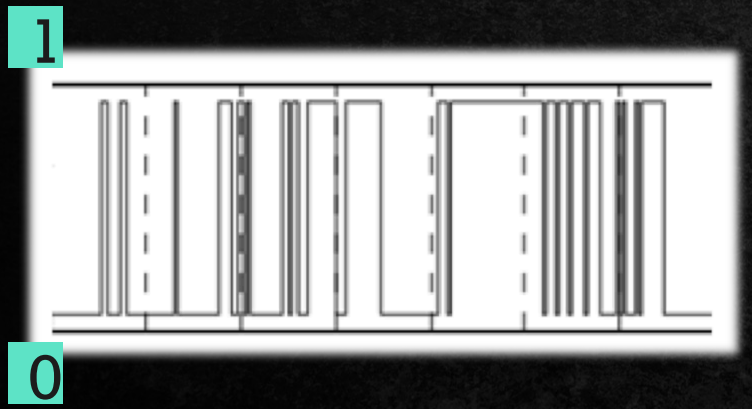
Detection of Departure/Arrival Time Sequences

0: still , 1: On a moving vehicle , 2:Walk



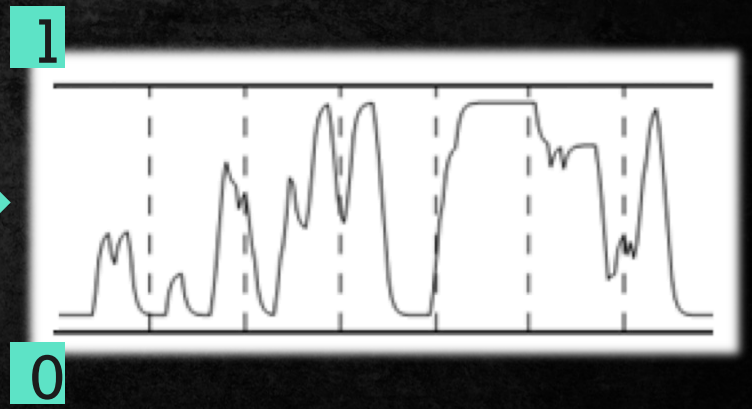
Noise reduction

Predicted user activities



EWMA

Smoothened user activities



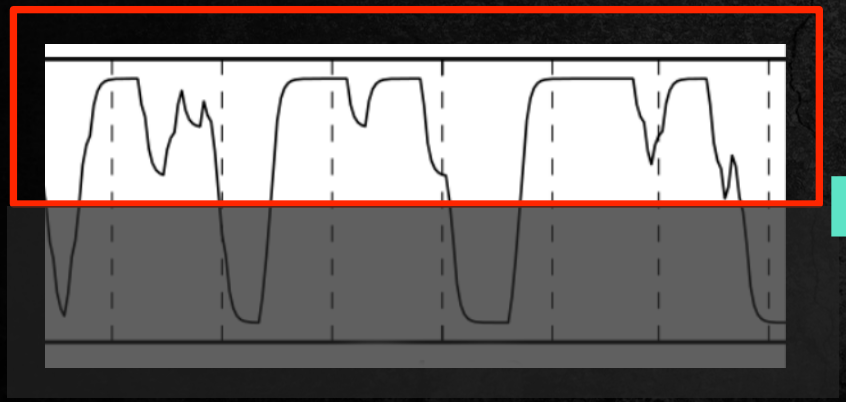
EWMA: Exponentially Weighted Moving Average

$$S_n = \lambda S + (1-\lambda) S_{n-1}$$

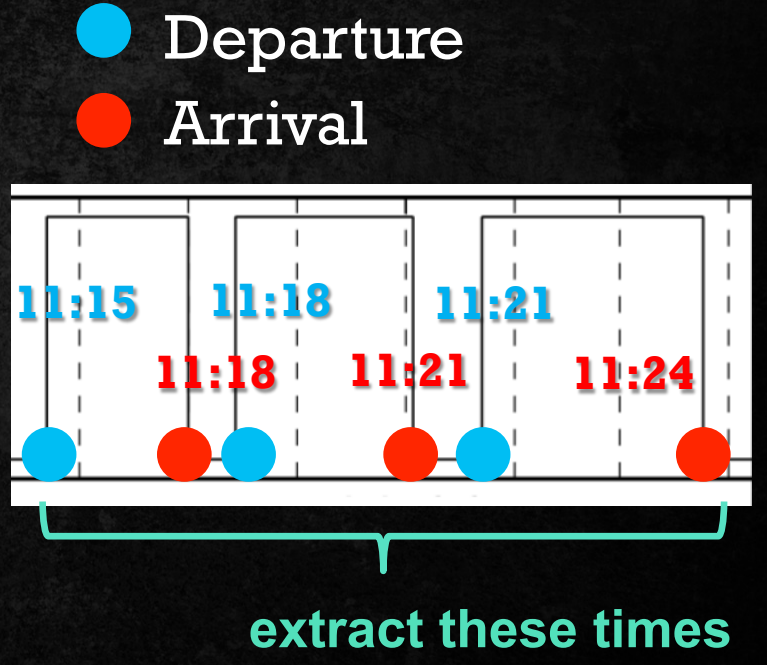
Noise reduction (cont.)

$0.5 > S \rightarrow 1$ (on vehicle)

$0.5 \leqq S \rightarrow 0$ (still)

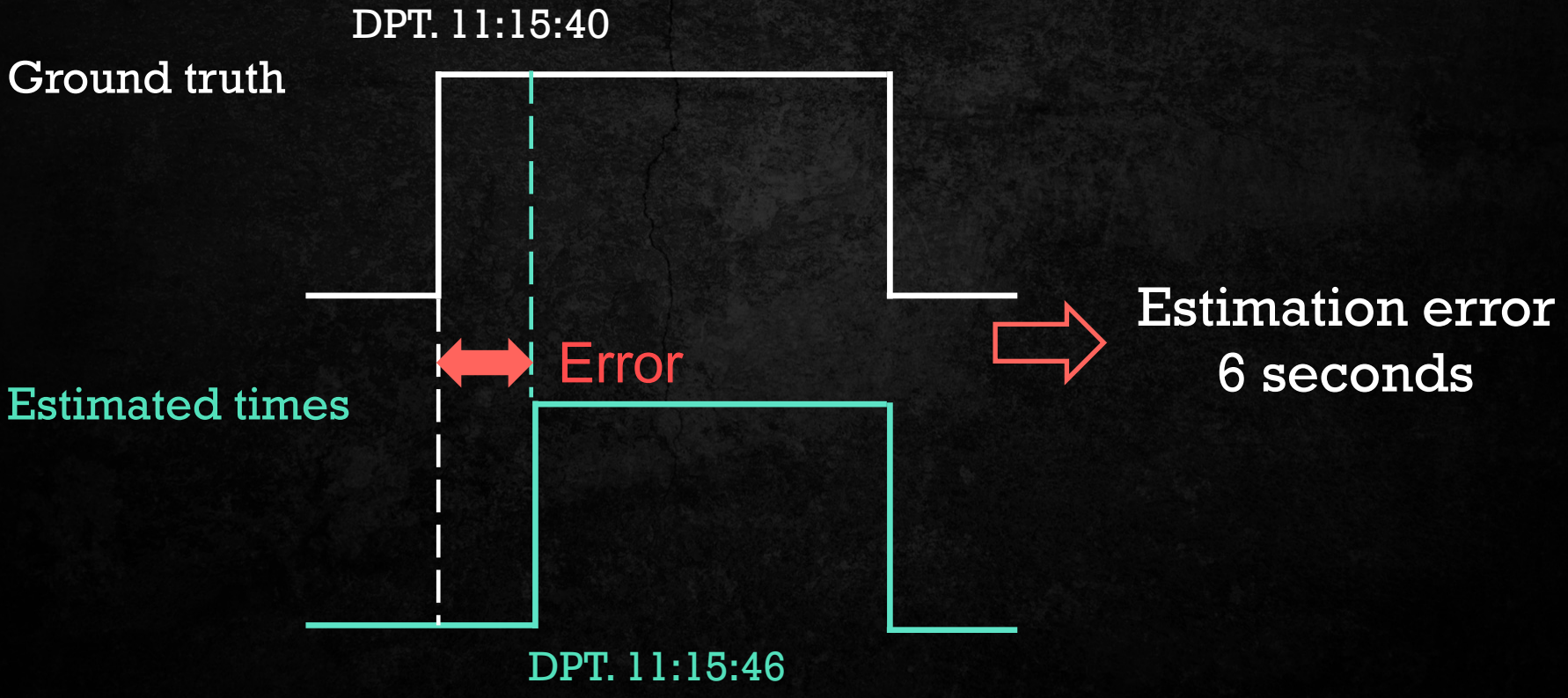


Smoothened user activities



Corrected user activities

Accuracy of detected DPT/ARR times



Stats of absolute estimation errors (secs)

Data	Departure			Arrival		
	min	max	**mean**	min	max	**mean**
HTC_hold	1.97	3.54	**2.79**	2.52	6.75	**4.13**
HTC_bag	2.04	3.06	**2.53**	1.71	4.63	**3.21**
Nexus_hold	2.33	7.94	**4.60**	3.07	10.78	**6.03**
Nexus_bag	1.55	2.76	**2.17**	2.22	5.16	**3.43**

Candidate Routes Extraction

Seq. of (DPT, ARR)

(11:30, 11:32)

⬇

(11:32, 11:35)

⬇

(11:35, 11:38)

Timetables

Candidate 1

- Tokyo
- →Kanda
- →Akihabara
- →Okachimachi

Candidate 2

- Shinjuku
- →Shin-Okubo
- →Takadanobaba
- →Mejiro

Timetables

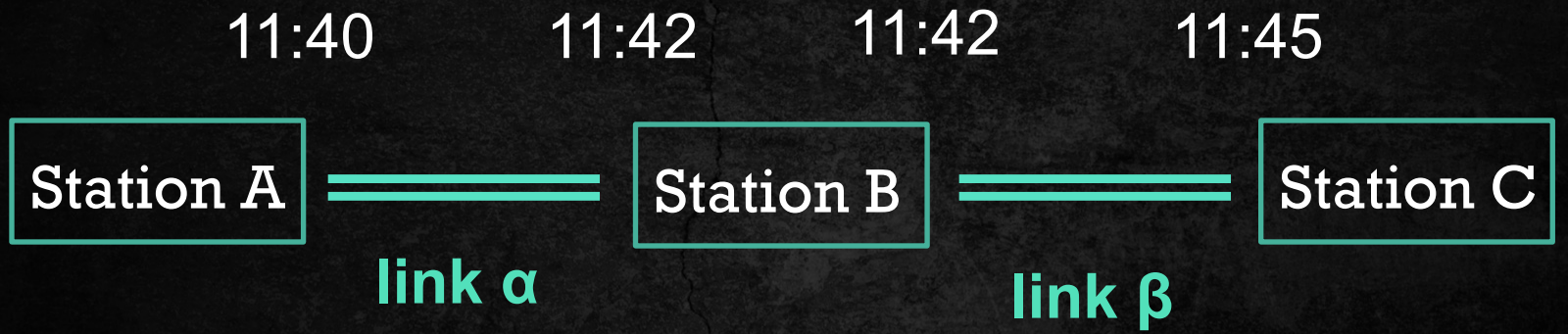
- Collected timetables of passenger train companies operating in Japan
- The DB covers all the prefectures in Japan
  - 9,090 railway stations
  - 597 lines
  - 172 railway companies
  - 2,277,397 "*links*"

Osaki	DPT.	11:46	11:51
Gotanda	DPT.	11:48	11:52
Meguro	DPT.	11:50	11:55
Ebisu	DPT.	11:53	11:57
Shibuya	DPT.	11:55	12:00
Harajuku	DPT.	11:58	12:02
Yoyogi	DPT.	12:00	12:05
Shinjuku	DPT.	12:02	12:07
Shin-Okubo	DPT.	12:04	12:09
Takadanobaba	DPT.	12:07	12:11
Mejiro	DPT.	12:09	12:13
Ikebukuro	ARR.	12:11	12:15

http://ekikara.jp/

Definition of a link

Link = {
- departure station
- arrival station
- departure time
- arrival time
- next link
}

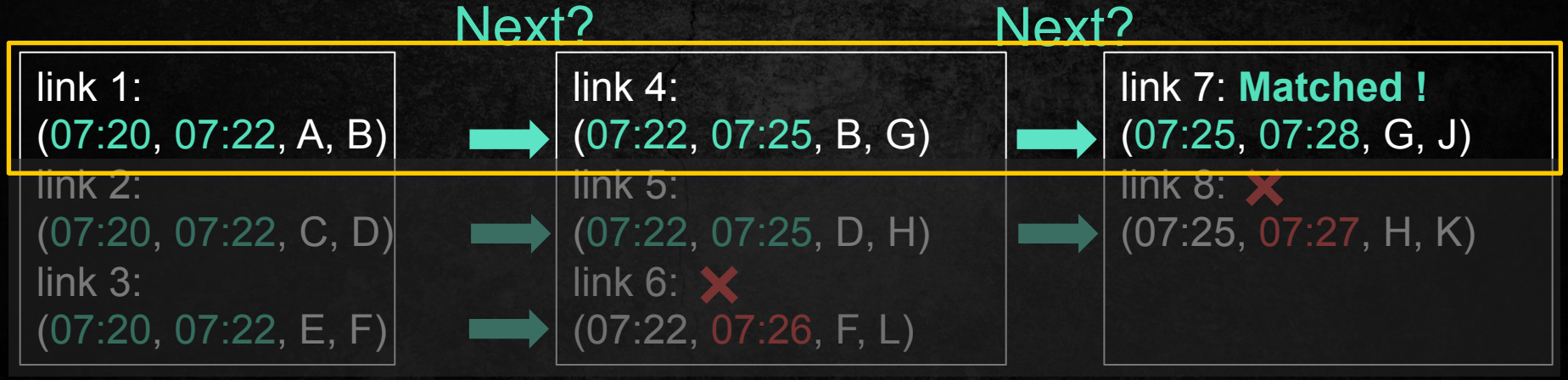


link α = { Sd = Station A, Sa = Station B, Td = 11:40, Ta = 11:42, Next = link β }

link β = { Sd = Station B, Sa = Station C, Td = 11:42, Ta = 11:45, Next = None }

Searching Candidate Routes

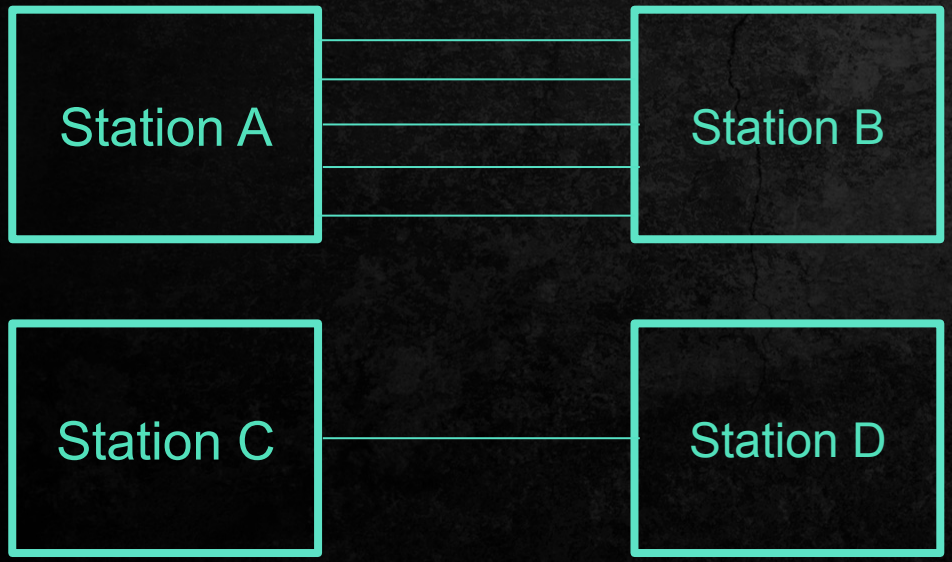
Input: [(07:20, 07:22), (07:22, 07:25), (07:25, 07:28)]



Results:

Score calculation

Count # of Links { Same origin / destination
Different Td / Ta }



Score: +5
(more popular)

Score: +1

Performance of the algorithm (1)

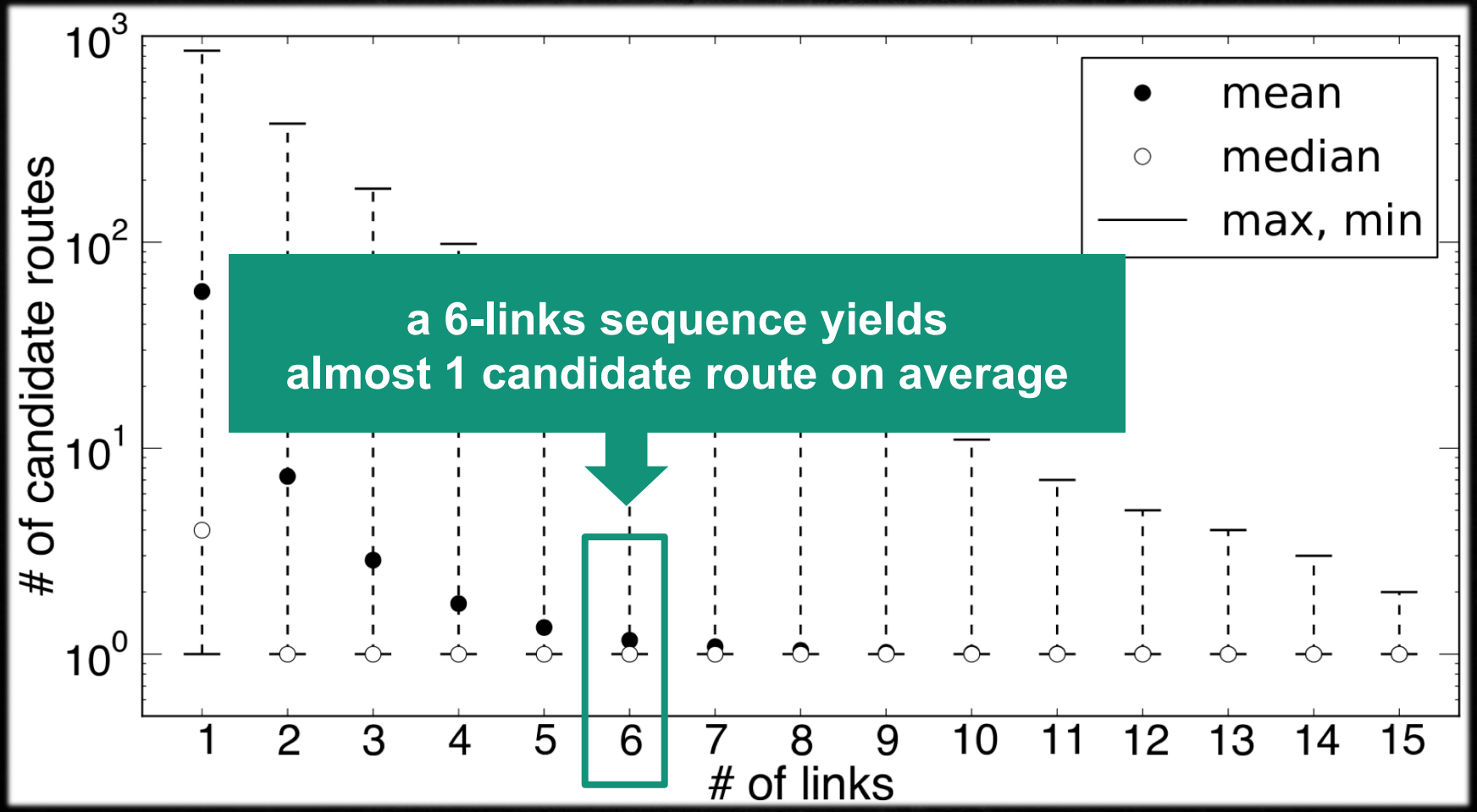
- Enumerate all the possible routes

  → extracted 6.4 billion routes

  – Sequence length = less than 15 links

  – At most 2 line changes

- Compute the relationship between the number of links and the number of candidate routes.


10^3
10^2
10^1
10^0
# of candidate routes
mean
median
max, min
a 6-links sequence yields
almost 1 candidate route on average
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
# of links

Performance of the algorithm (2)

- More than 6.4 billion routes were searched within $74\ mins$

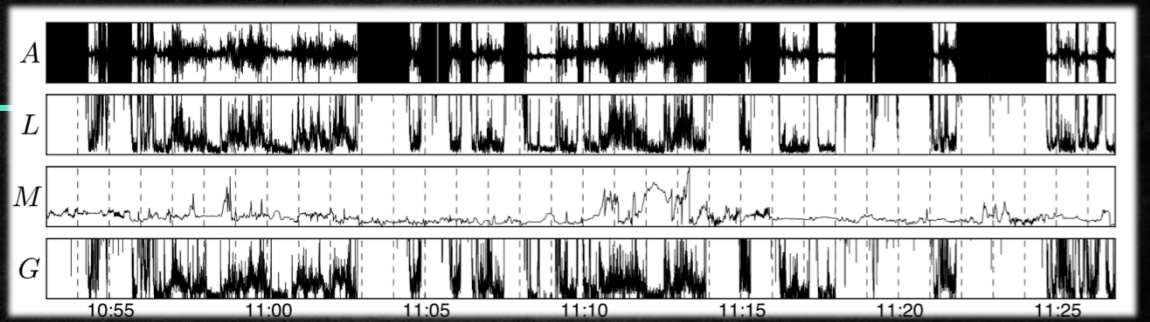
- A route was searched

  within $74 mins / 6400000000 = 7.1\ \mu sec$

  on average

Case study

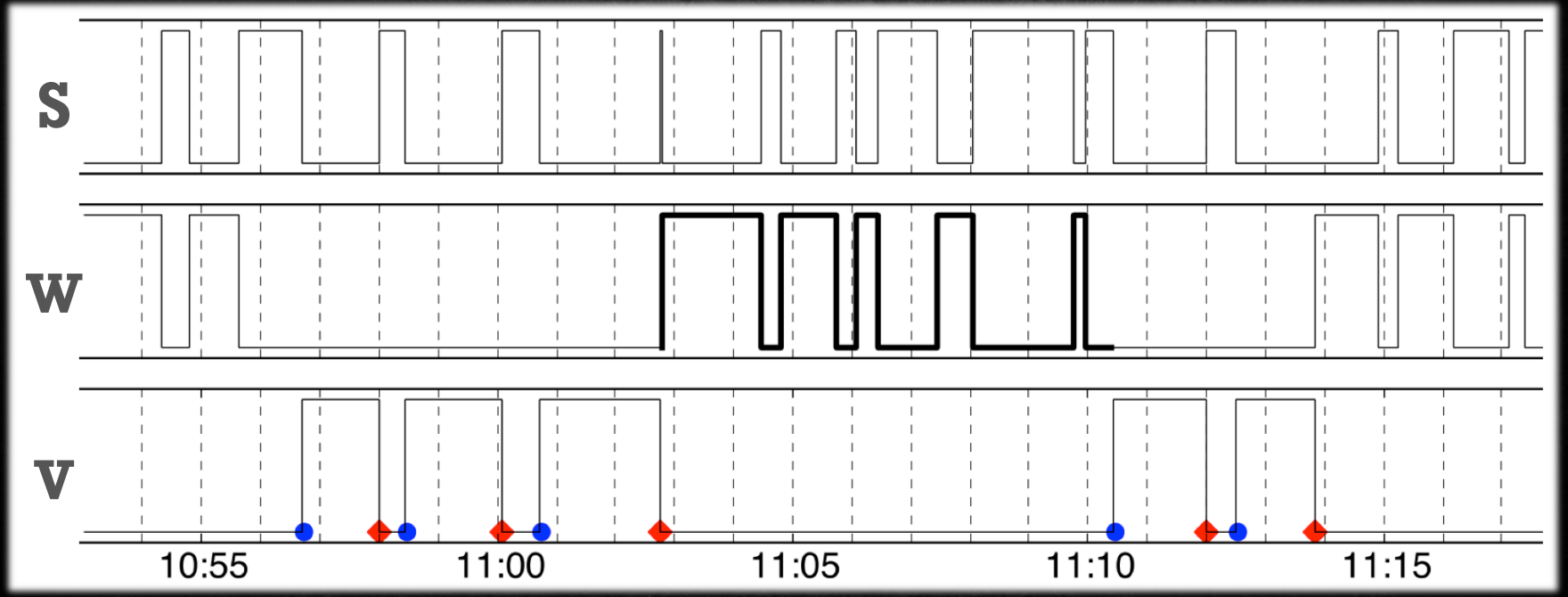
A victim



Sensor data

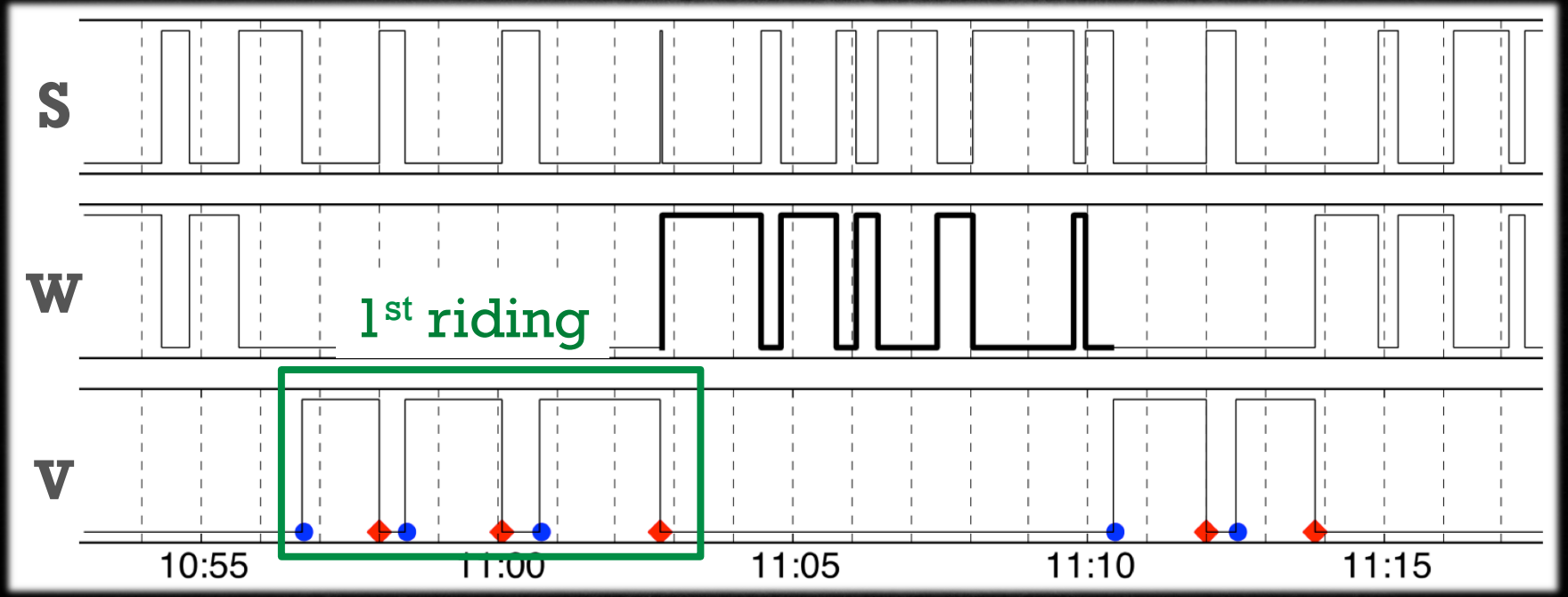
Route Detector

Case study 1
Results of activity detection



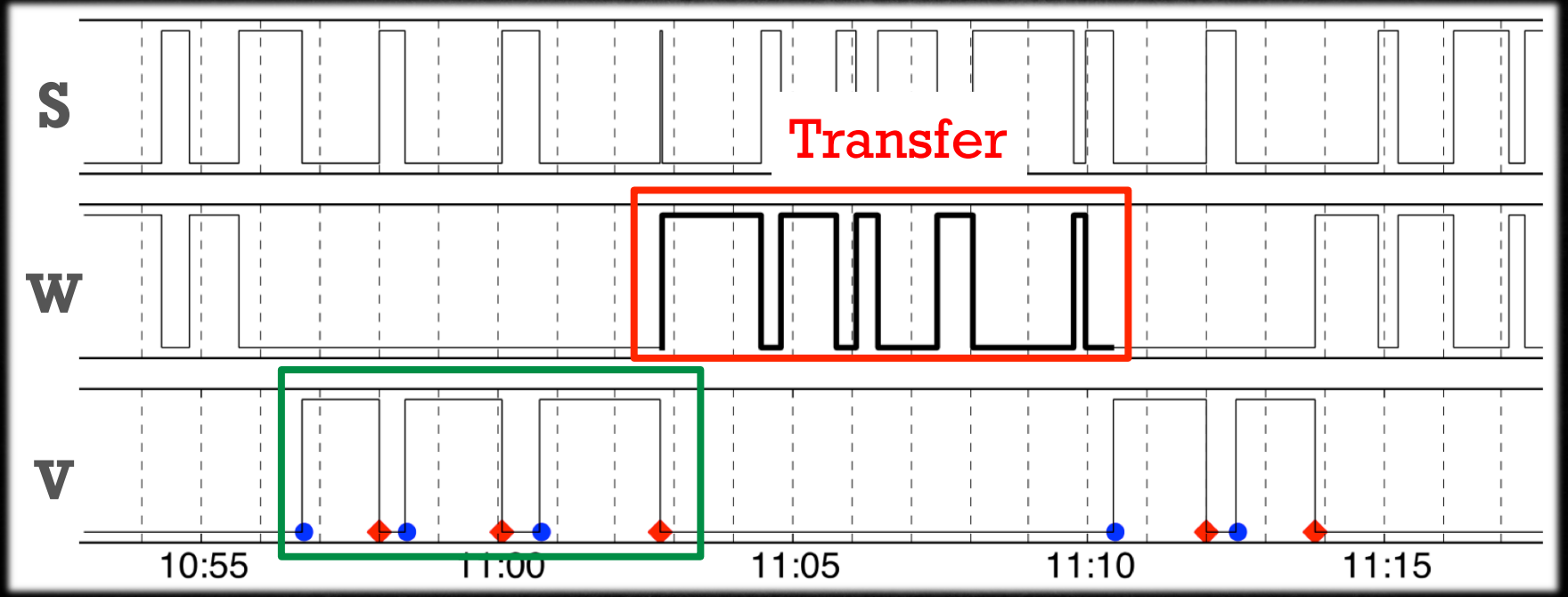
S: Still W: Walk V: Vehicle

Case study 1
Results of activity detection



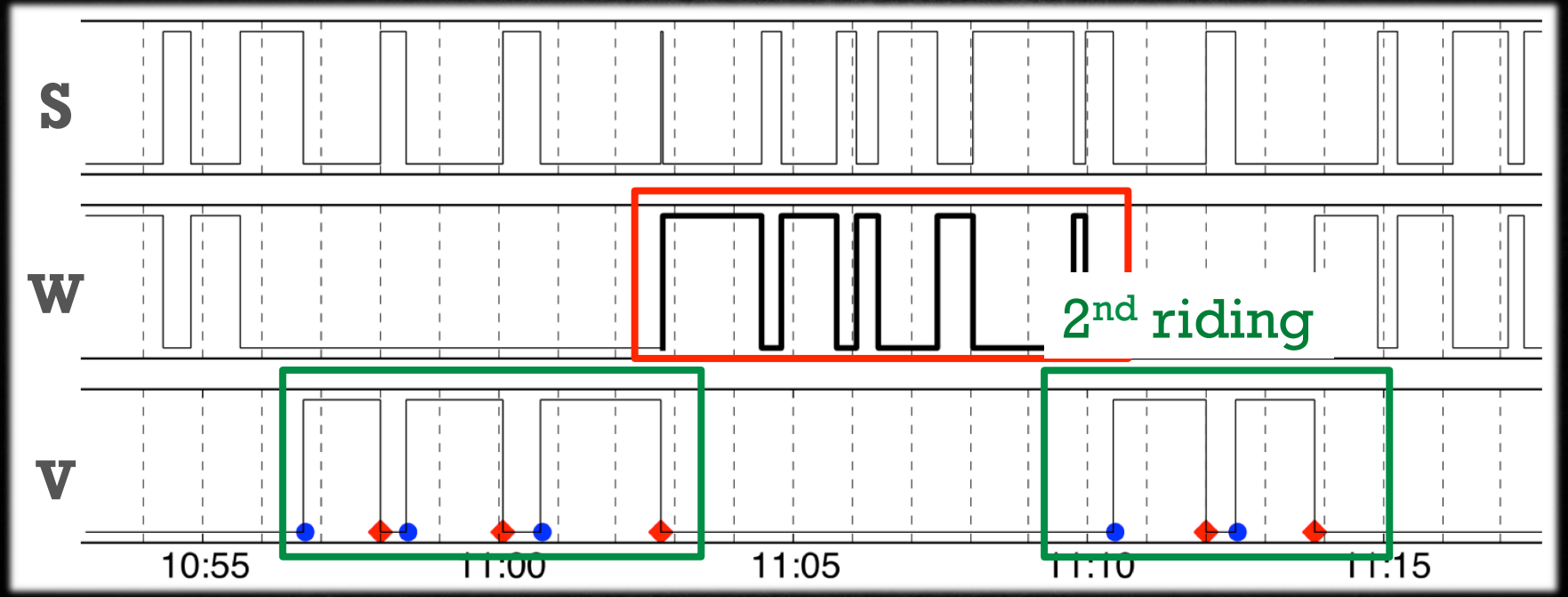
S: Still W: Walk V: Vehicle

Case study 1
Results of activity detection



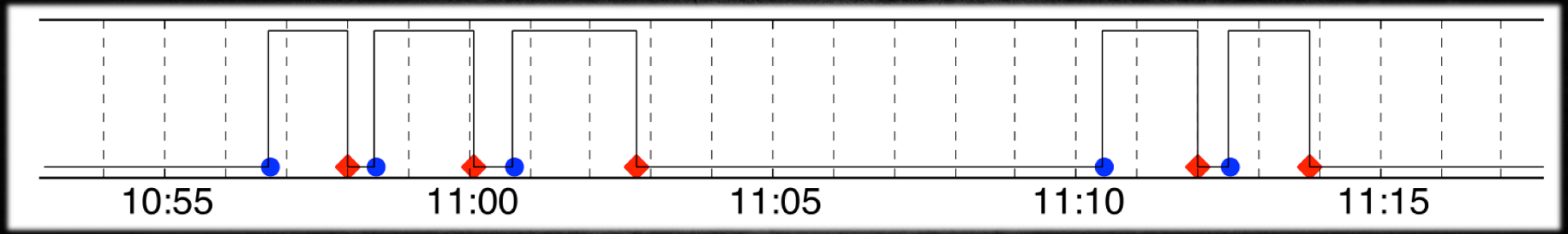
S: Still W: Walk V: Vehicle

Case study 1
Results of activity detection



S: Still W: Walk V: Vehicle

Case study 1
Results of Dpt./Arr. time detection

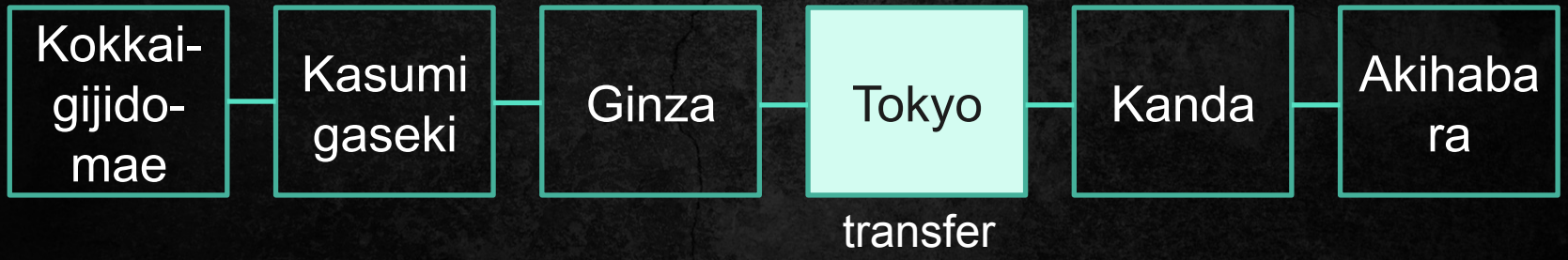


(10:56, 10:58) → (10:58, 11:00) → (11:00, 11:03) →
(change the line) → (11:10, 11:12) → (11:12, 11:14)

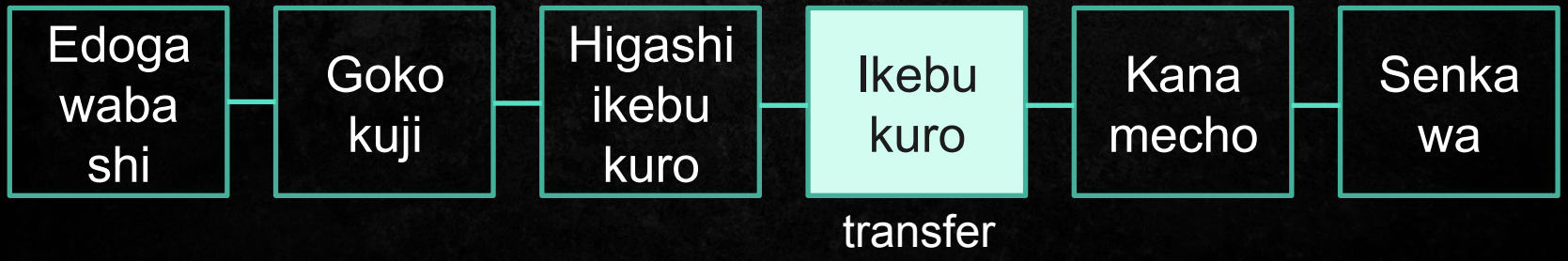
» Candidate Routes

Case study 1
Results of Candidate Routes Detection

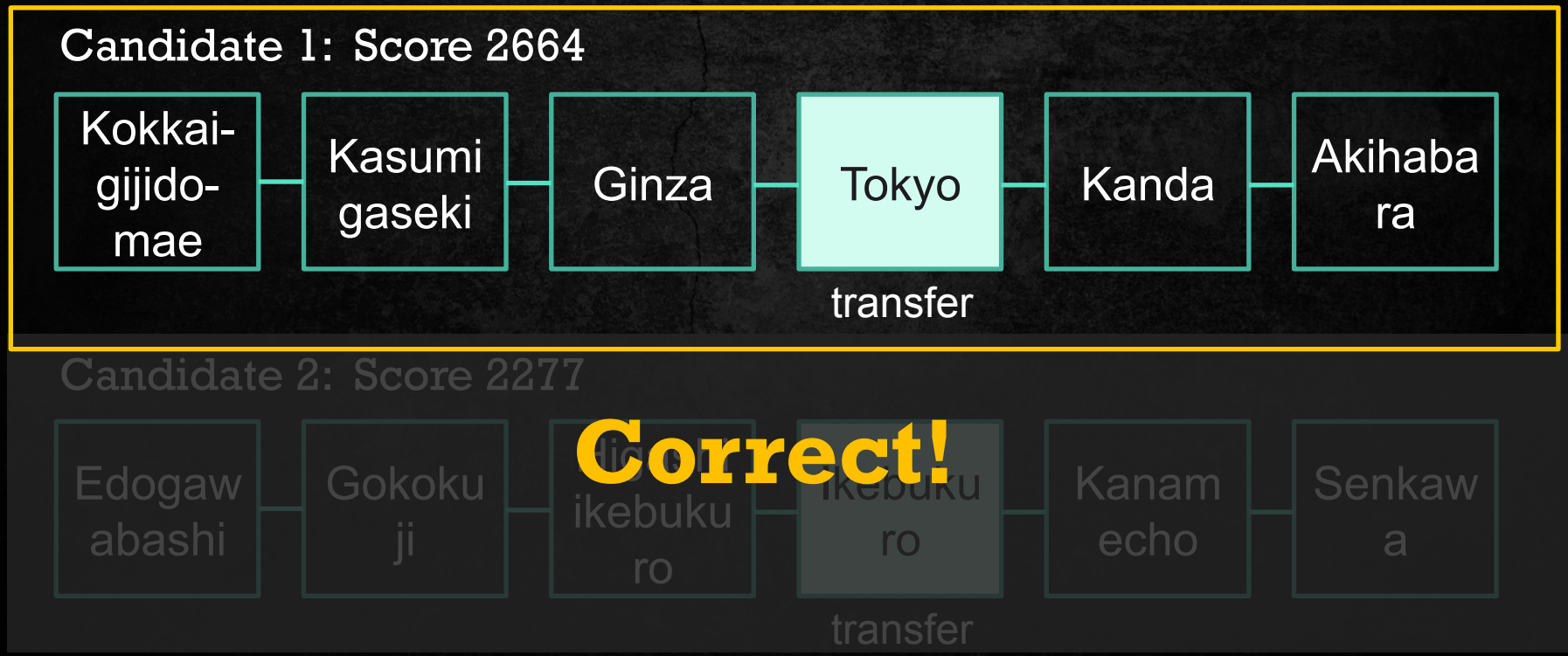
Candidate 1: Score 2664



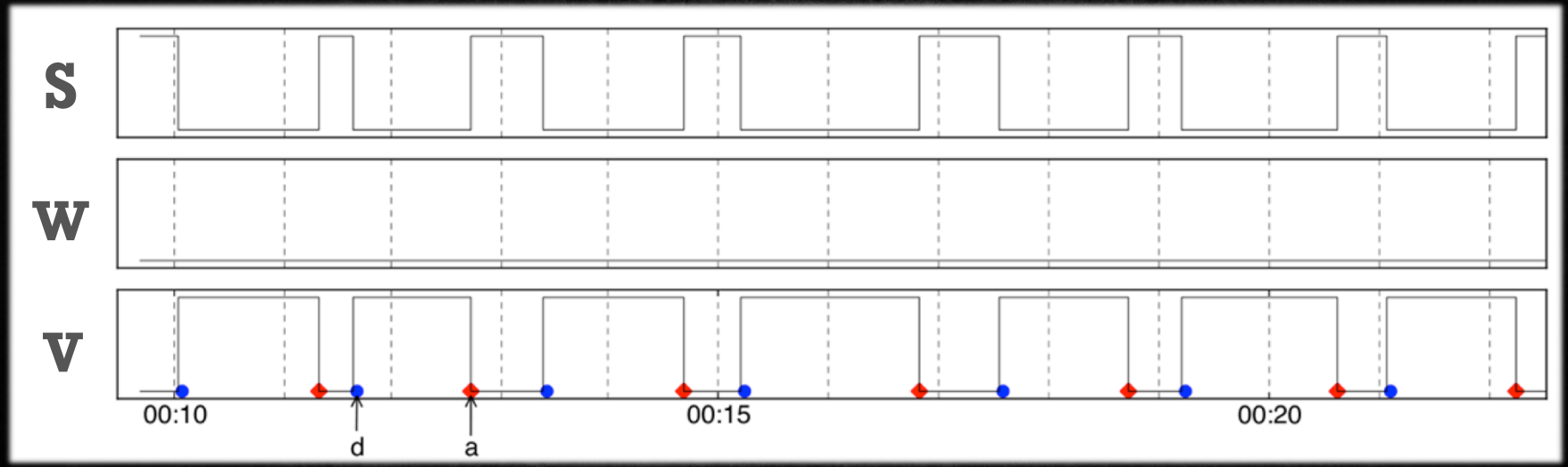
Candidate 2: Score 2277



Case study 1
Results of Candidate Routes Detection



Case study 2



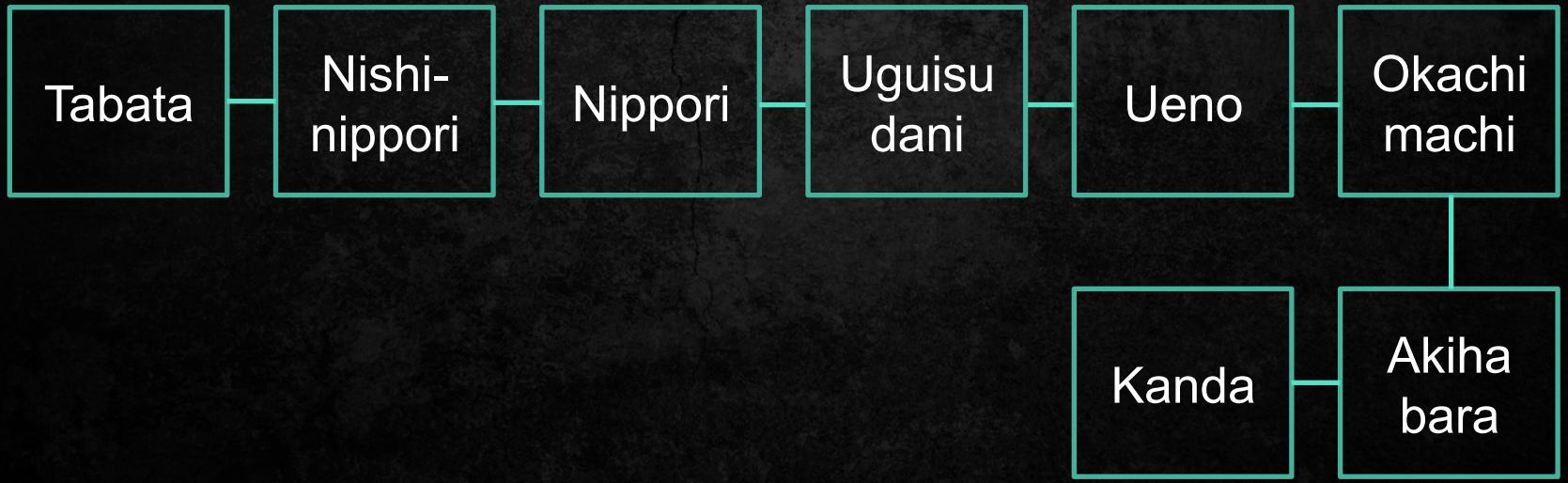
S: Still W: Walk V: Vehicle

(0:10, 0:11) → (0:11, 0:13) → (0:13, 0:15) → (0:15, 0:17)

→ (0:17, 0:19) → (0:19, 0:21) → (0:21, 0:22)

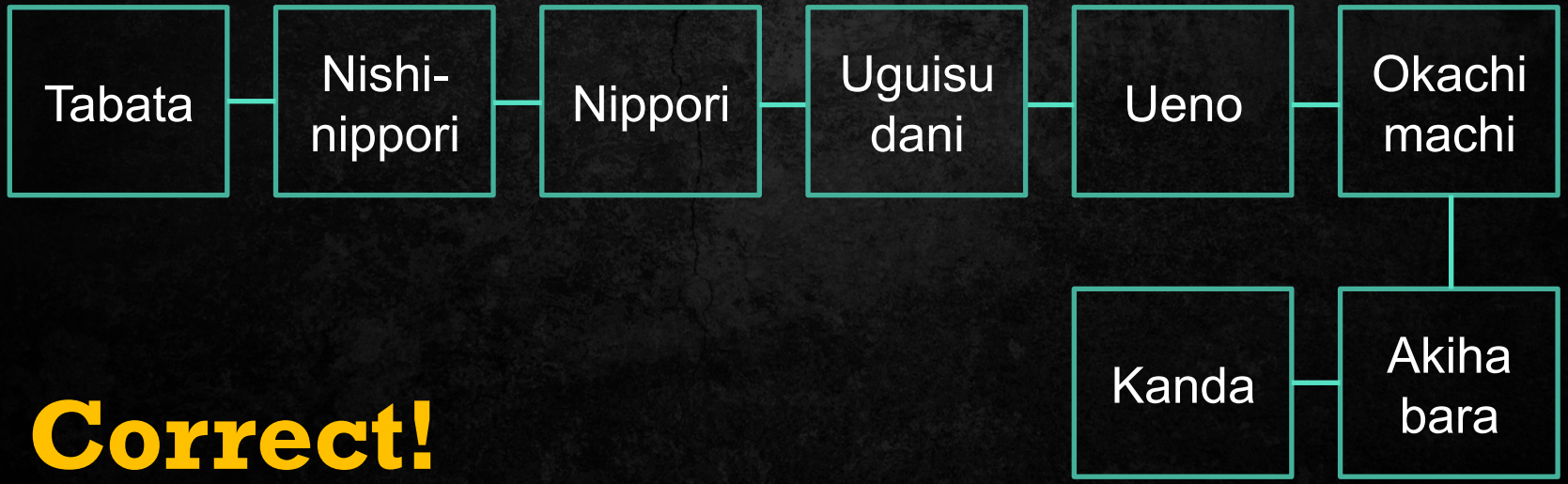
Case study 2
Results of Candidate Routes Detection

Candidate 1 (The only): Score 3892



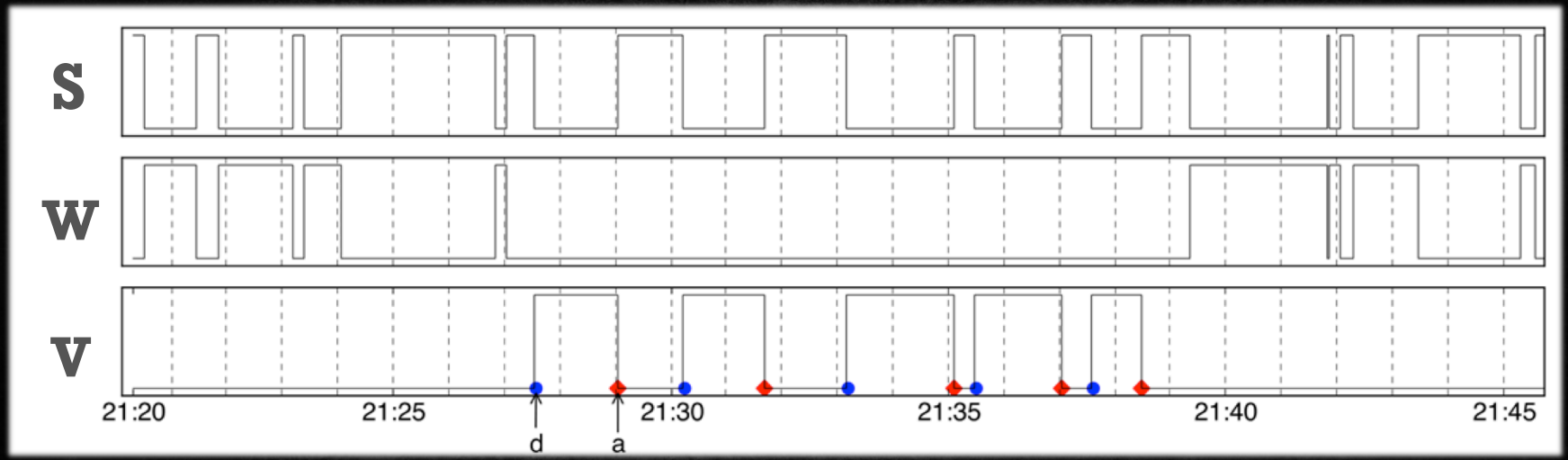
Case study 2
Results of Candidate Routes Detection

Candidate 1 (The only): Score 3892



Correct!

Case study 3



S: Still W: Walk V: Vehicle

(21:27, 21:29) → (21:30, 21:32) → (21:33, 21:35)

→ (21:35, 21:37) → (21:37, 21:39)

Case study 3
Results of Candidate Routes Detection

None

Why we failed to detect in Case 3 ?

	DPT	ARR	DPT	ARR	DPT	ARR	DPT	ARR	DPT	ARR
Scheduled	21:26	21:28	21:28	21:32	21:32	21:35	21:35	21:37	21:37	21:39
Observed	21:27	21:29	21:30	21:32	21:33	21:35	21:35	21:37	21:37	21:39
Detected	21:27	21:29	21:30	21:32	21:33	21:35	21:35	21:37	21:37	21:39

Why we failed to detect in Case 3 ?

	DPT	ARR	DPT	ARR	DPT	ARR	DPT	ARR	DPT	ARR
Scheduled	21:26	21:28	21:28	21:32	21:32	21:35	21:35	21:37	21:37	21:39
Observed	21:27	21:29	21:30	21:32	21:33	21:35	21:35	21:37	21:37	21:39
Detected	~~21:27~~	~~21:29~~	~~21:30~~	~~21:32~~	~~21:33~~	~~21:35~~	~~21:35~~	~~21:37~~	~~21:37~~	~~21:39~~

DPT./ARR. time detection was perfect

Why we failed to detect in Case 3 ?

	DPT	ARR	DPT	ARR	DPT	ARR	DPT	ARR	DPT	ARR
Scheduled	21:26	21:28	21:28	21:32	21:32	21:35	21:35	21:37	21:37	21:39
Observed	21:27	21:29	21:30	21:32	21:33	21:35	21:35	21:37	21:37	21:39
Detected	21:27	21:29	21:30	21:32	21:33	21:35	21:35	21:37	21:37	21:39

The train was delayed at the time of measurement

Discussion - Train Operation

#1001	On time
#1002	On time
#1003	On time
#1004	On time

Detectable

#8001	Delayed
#8002	Delayed
#8003	Suspended
#8004	Delayed

Undetectable

Discussion - Train Operation (cont.)

- Multiple observations
  - An adversary can figure out locations frequently visited by the target in a statistical way
- Automatic train operation
  - It will work to increase the accuracy of train operations

Countermeasures

- Restricting access to raw sensor data
  - Requiring permission
  - Wrapping in APIs
- Low-pass filtering
  - The trade offs between functionality and security
- Staying away from Japan

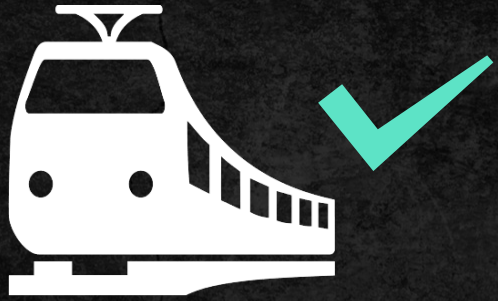
Conclusion

- A novel, proof-of-concept side-channel attack framework called RouteDetector was introduced

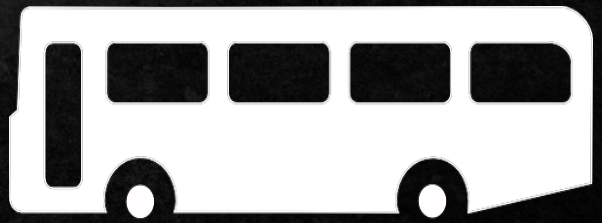
- It needs only sensors data which is not protected on the Android platform
- We successfully demonstrated that a route can be identified, used for a trip by train using timetables and route maps.

Q&A

Discussion - Types of Vehicles



?



?

Discussion - Types of Vehicles (cont.)

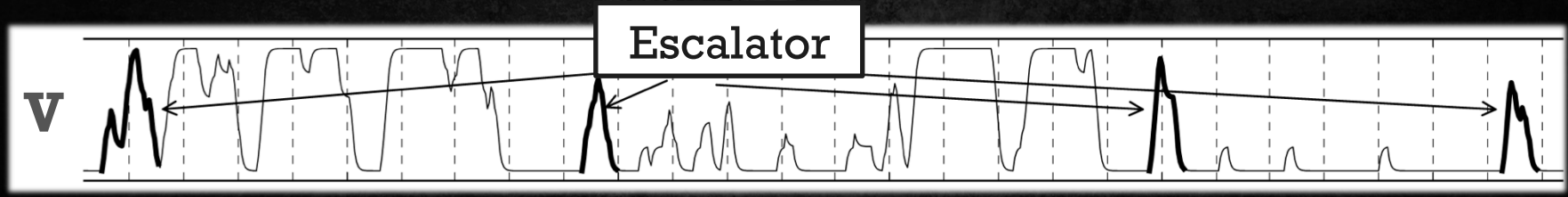
- An airplane
- A mono rail
  - Possible candidate to be attacked

- A bus

average time waiting at traffic lights **>** average time waiting at bus stops

Discussion – Detection errors

- Our classifier can predict the activity of riding an escalator as "Vehicle"



- To distinguish between escalator and train, we apply a timing heuristics;

Discussion – Detection errors (cont.)

- We consider a short "vehicle activity" (less than 60 seconds) as other activities

  - Coping with long escalator ride,
    e.g., more than 60 seconds, is for future work.

Image sources

Map: https://www.google.co.jp/maps

Boss icon: http://pictogram-free.com/03-mark/042-mark.html

Location icon: http://www.cliparthut.com/map-icons-clip-art-clipart-W13lHd.html

Train photo: http://blogs.yahoo.co.jp/yuuki_20140313/40442989.html

Sensor image: http://fscomps.fotosearch.com/bigcomps/CSP/CSP139/k1398123.jpg

Train icon: https://commons.wikimedia.org/wiki/File:Bahn_aus_Zusatzzeichen_1024-15.svg

Timetable icon: http://simpleicon.com/book-2.html

Airplane icon: http://www.sozai-library.com/wp-content/uploads/2013/05/00289-450x337.jpg

Bus icon: http://4vector.com/i/free-vector-bus-symbol-black-clip
-art_110561_Bus_Symbol_black_clip_art_hight.png

Home icon: http://free-icon.org/data/dl_05/m_06.gif

Bar icon: http://map-icon.com/material/eatanddrink/m_05.gif

Malware icon: http://freeiconbox.com/icon/256/30992.png

Smartphone with xyz axis: http://vnreview.vn/image/61/36/613648.jpg?t=1373940855536

Photo in the train :http://www.uraken.net/rail/alltrain/ec/syanai/207e.jpg

Nexus7: http://i.ytimg.com/vi/Vj1koPa9FGQ/maxresdefault.jpg

Htc j: http://adcdn.goo.ne.jp/images/sumaho/model/au/htc_j_butterfly_htl21_c.jpg

Acknowledgements

- A part of this work was supported by JSPS Grant-in-Aid for Challenging Exploratory Research (KAKENHI), Grant number 15K12038.