Mobile Phone App Recognition via Magnetic Side Channel

2021.10.13

Outline

Background and Motivation

Related Works and Limitations

Preliminary Analysis

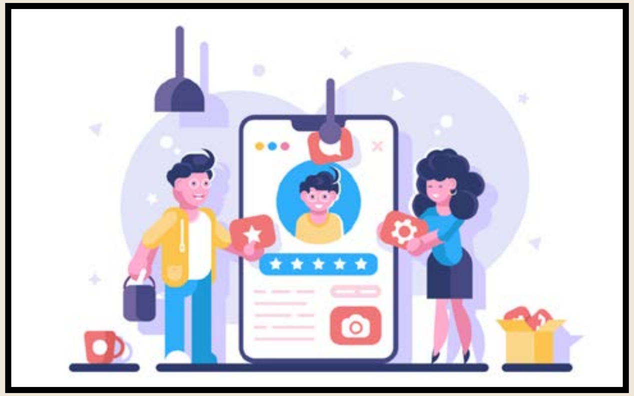
System Design

Evaluation

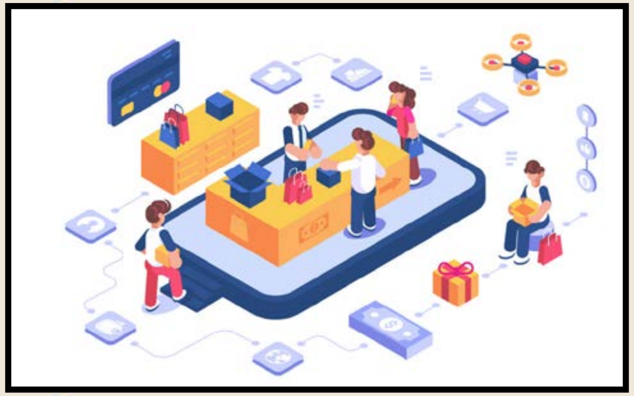
Conclusion

Mobile apps are so popular!

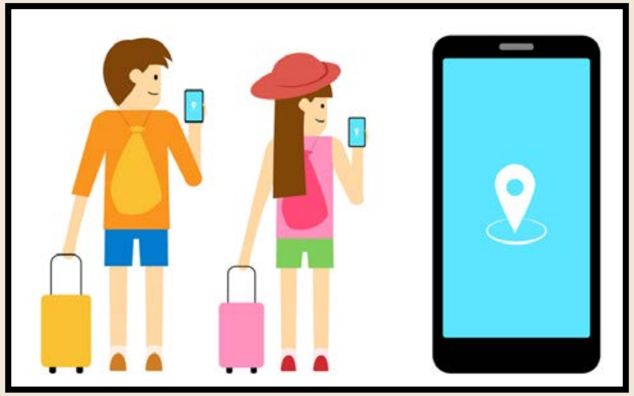
Social Network



Online Shopping

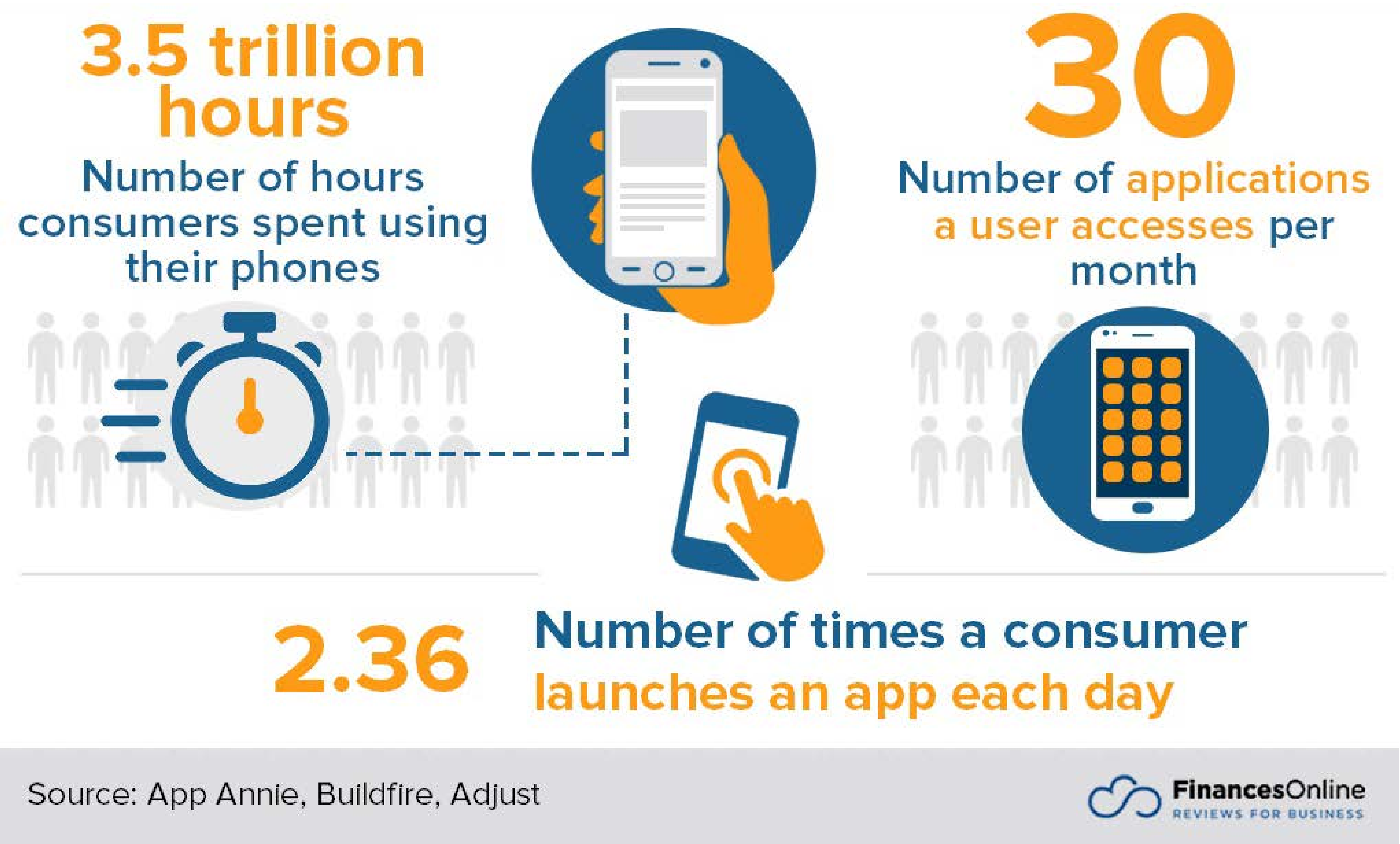


Navigation/Travel



Business/Working

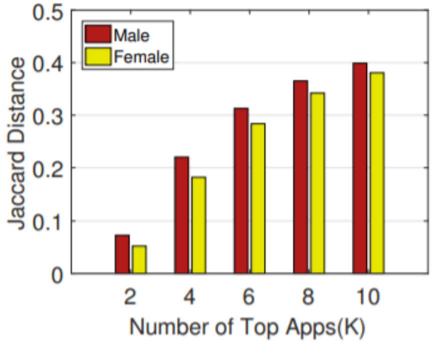
Mobile app usage by the numbers



Mobile apps may also give you away…

Discover Different Types of Mobile User

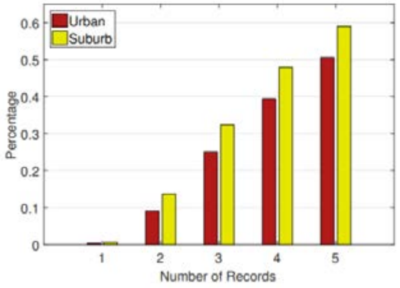
➢ Age, Gender, Income

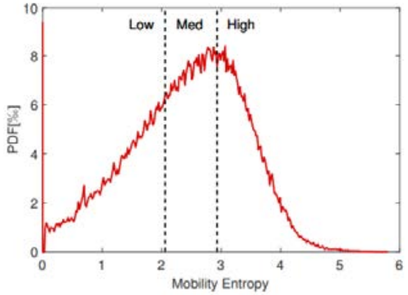


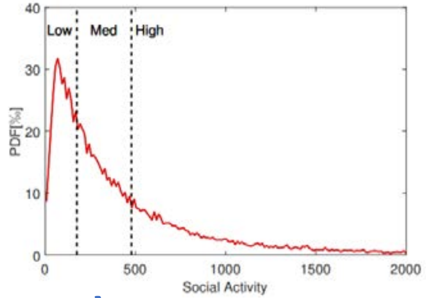
➢ Personal Interests

APPs	Attributes	Installation package
QQ电影票	Movie_fan	com.tencent.movieticket
号百彩票	Lottery	buke.besttone.caipiao.plugin
股票财经	Stocks	com.besttone.FortuneStreet.plugin
艺龙旅行	Travel	com.dp.android.elong
搜房网	Housing	com.soufun.app
高德导航	Driving	com.autonavi.xmgd.navigator
超级课程表	Student syllabus	com.xtuone.android.syllabus
美团	Group_buying	com.sankuai.meituan
美丽购	Beauty shopping	com.geili.gou
粉粉日记	Pinknote	pinkdiary.xiaoxiaotu.com

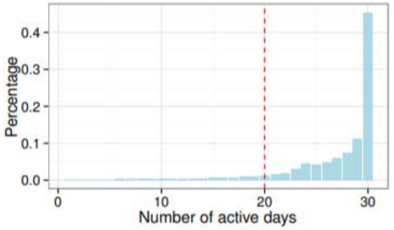
➢ Spatial Info (e.g., urban or suburb)

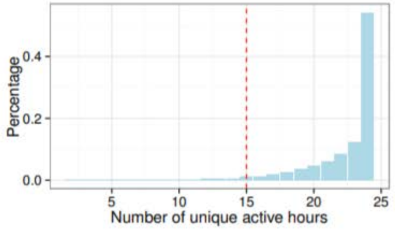


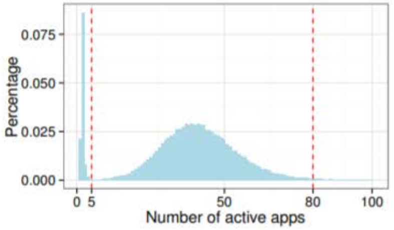




➢ Lifestyles (active days/hours/# of apps)



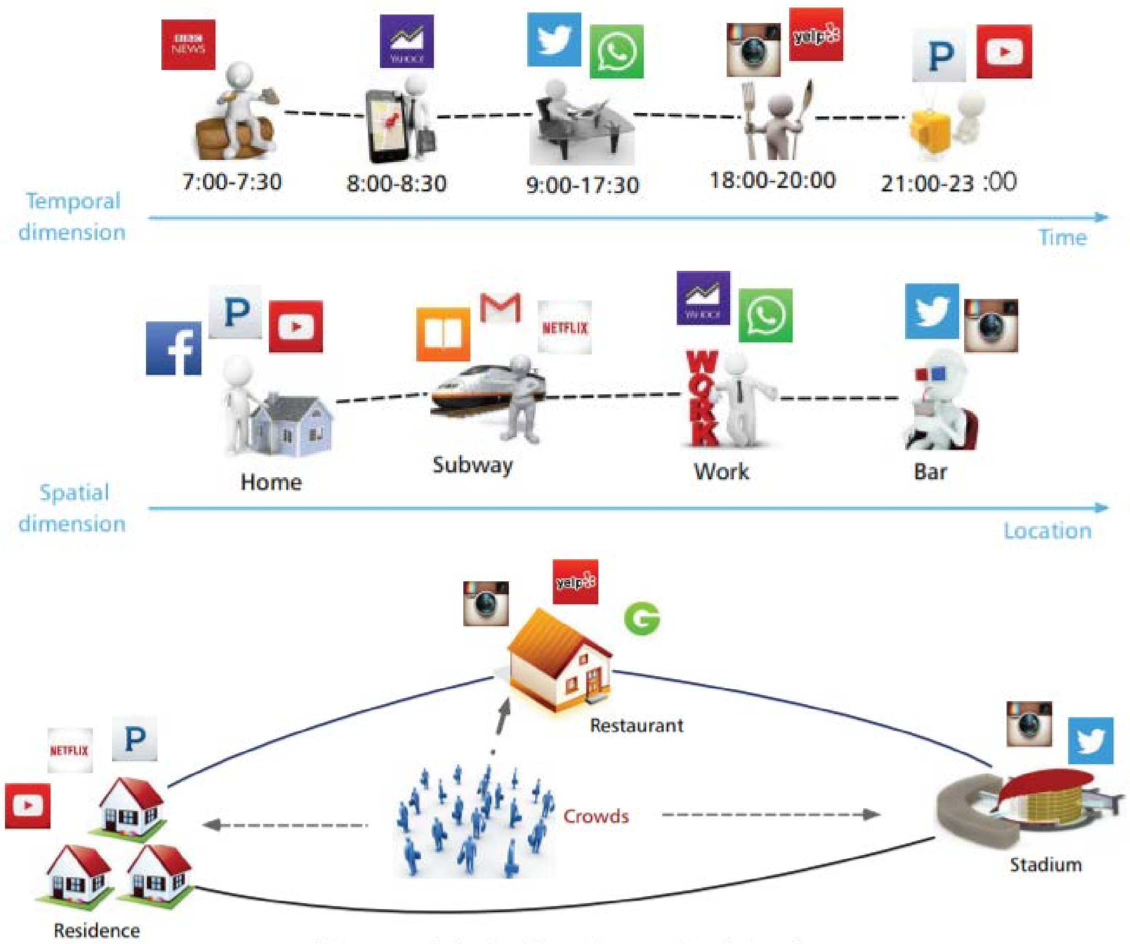




ACM UbiComp/IMWUT 2016/2018/2019

IEEE System Journal 2017

Understand Human Mobility



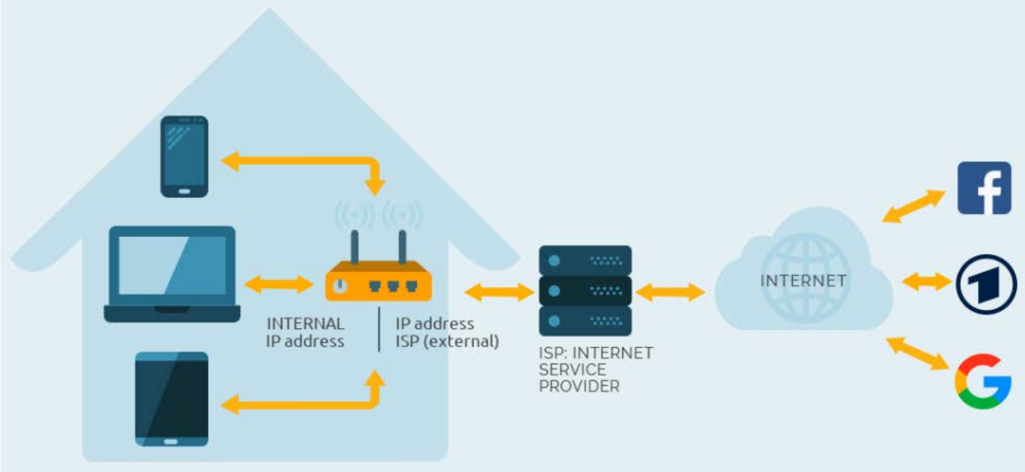
(b) App usage behavior of crowds at crowd gathering places.

IEEE Networks 2016

How to collect mobile app usage behaviors secretly?

Internet Service Provider (ISP) Datasets

- ➢ Cellular network traffic
- ➢ Extract app usage from HTTP headers

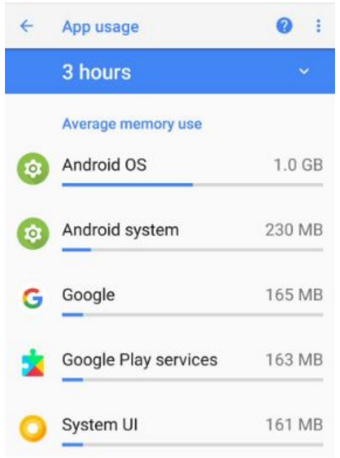


User ID	Date	Hours	Used apps	Weight
0000751aecb005a2	2015-09-01	09-10	com.miui.home	0.85
0000751aecb005a2	2015-09-01	09-10	com.android.incallui	0.85
0000751aecb005a2	2015-09-01	10-11	com.miui.home	0.15
0000751aecb005a2	2015-09-01	10-11	com.android.incallui	0.15

Privacy-related regulations limit third-party access to data ☹

Pertain from the mobile devices directly

- ➢ App usage function
- ➢ System-kernel information
  - proc filesystem
  - memory
  - internet traffic data
  - battery and CPU



Operating systems have prompted the third-party apps to curtail access to these data ☹

Outline

Background and Motivation

Related Works and Limitations

Preliminary Analysis

System Design

Evaluation

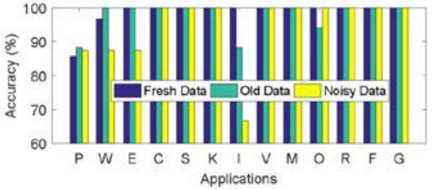
Conclusion

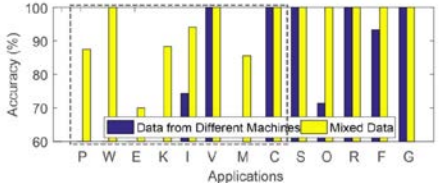
Related Work: *Application launching process* identification with EM side-channel signals

Use smartphone to sense victim's app usage on surrounding laptops

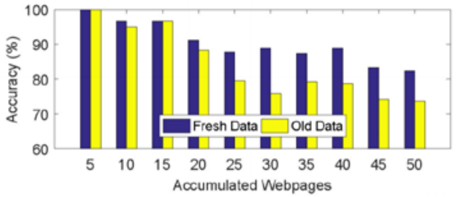
Sniff app usage on the smartphone with built-in magnetometer

Applications classification:



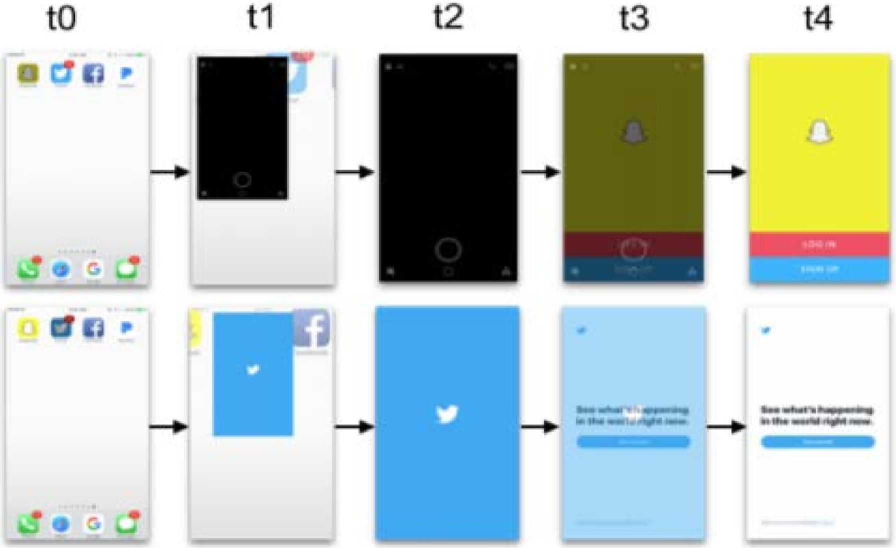


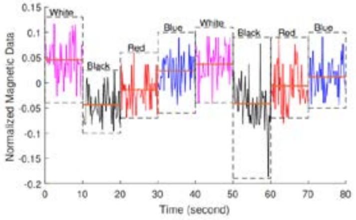
Websites classification:

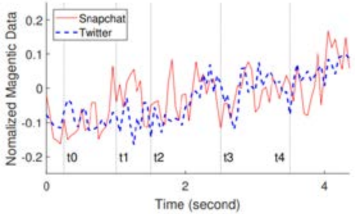


MagAttack (ACM AsiaCCS 2019)
Magneticspy (ACM WPES 2019)

Infer app usage with magnetometer readings by training CNN model







Distance to Refrigerator (cm)	25	50	100
Magnetic Model (Cross Model Mix) + Motion	0.9721	0.9817	0.9769
Orientation Model (Cross Model Mix) + Motion	0.9768	0.9761	0.9782

Deepmag (IEEE PerCom 2018)

Different manners of launching an app

Cold Start (from scratch)

***Cold* start has four tasks:**

1. Loading and launching of the app
2. Displaying a theme starting window
3. Creating the application process
4. Inflating & rendering of layouts

Warm Start (from memory)

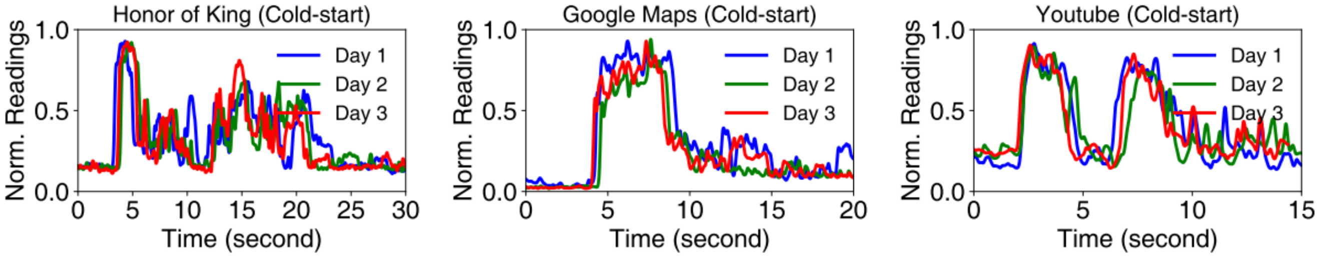
***Warm* starts has one tasks:**

1. Switching back to the app from “warming” memory.

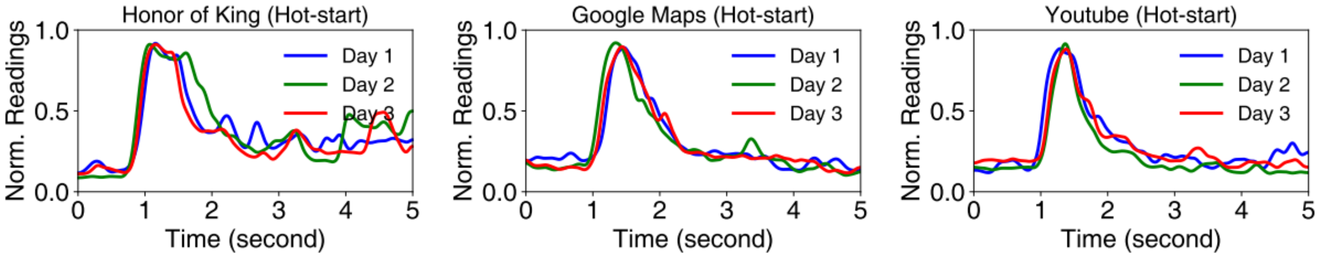
A high-frequency method used to launch apps for the mobile users ☺

Problems of app launching identification

EM signals of app launching via Cold Start



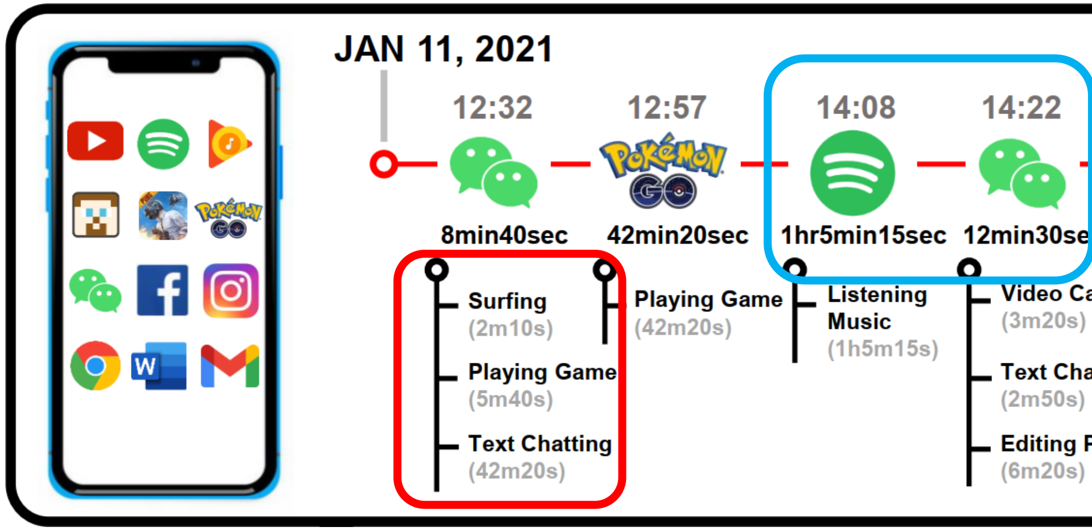
EM signals of app launching via Cold Start



Classification results of EM signals generated by app launching

	kNN	LDA	SVM	RF	MLP
Cold	89.7%	93.5%	93.7%	94.9%	95.6%
Hot	11.67%	12.92%	13.37%	15.72%	16.14%

PROBLEM 1: warm start of app launching is HARD to identification.



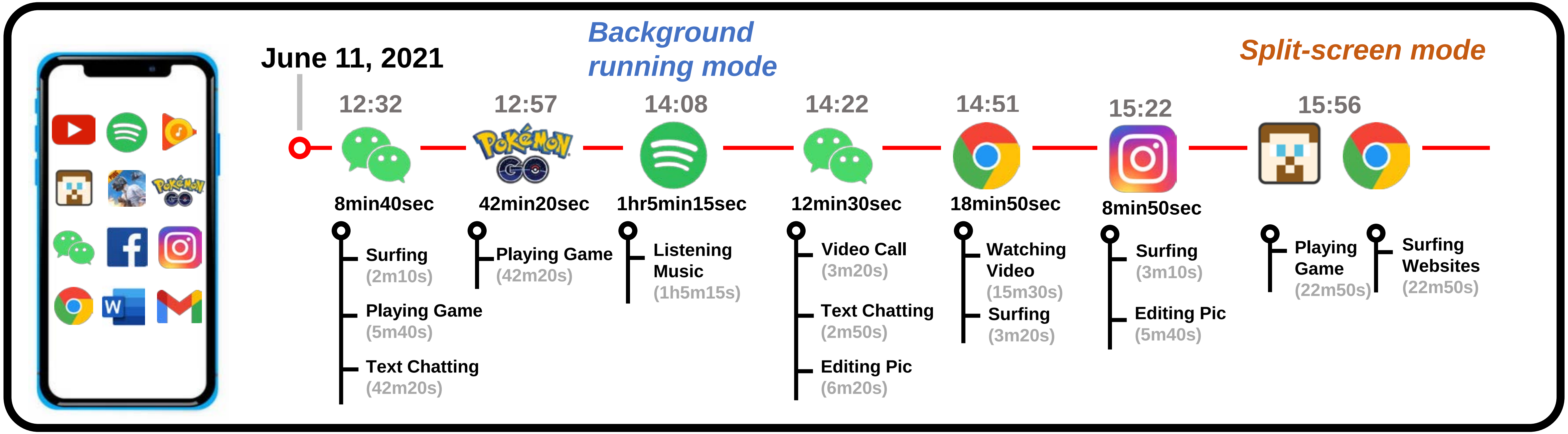
Complete app usage behaviors contains:

1. Start/Switch/Close timestamp
2. In-app service when using an app
3. Simultaneous usages of multiple apps

(in split-screen mode/background running)

PROBLEM 2:
App's launching information
≠
Complete app usage behaviors

Our Target



Tracking the complete app usage behaviors in real time :

- **Multi-label problem:**
  - **Identify the *app & in-app services* types**
- **Multi-target problem:**
  - **Identify multiple running apps, including *background running* and *split-screen modes***

Outline

Background and Motivation

Related Works and Limitations

Preliminary Analysis

System Design

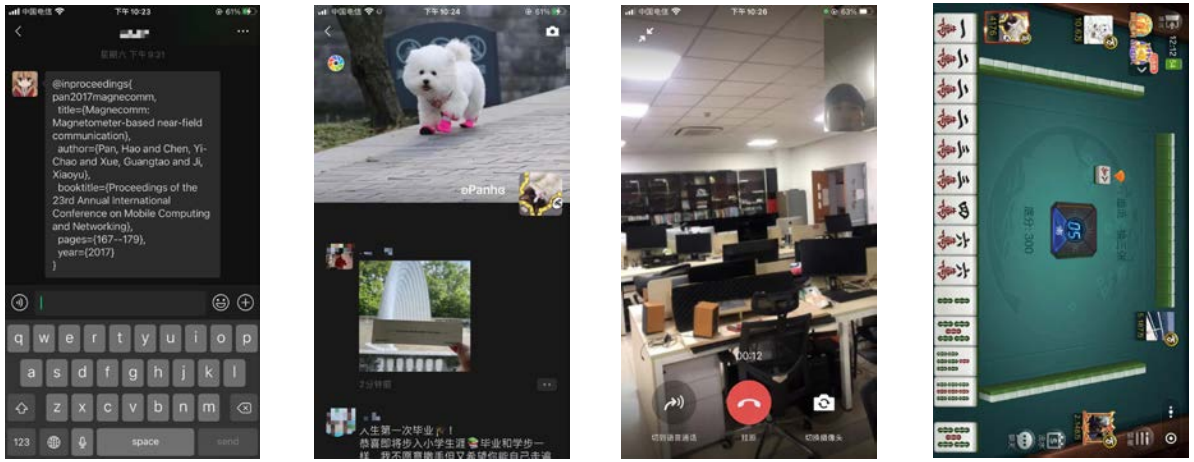
Evaluation

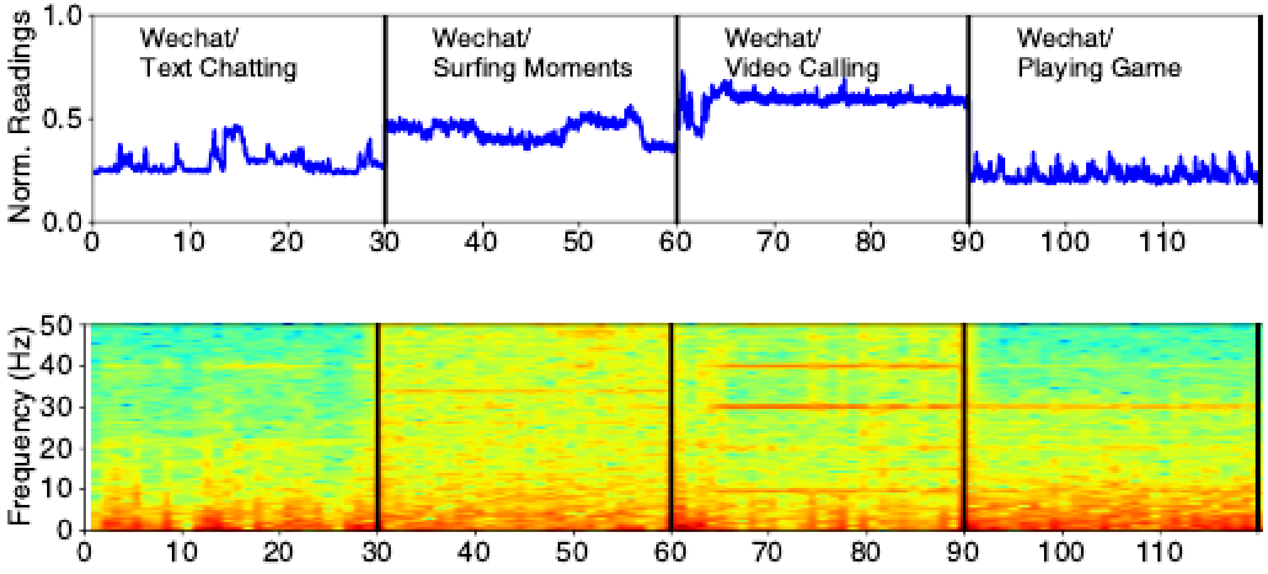
Conclusion

Preliminary experiment I —— app & in-app service

App 1: Wechat

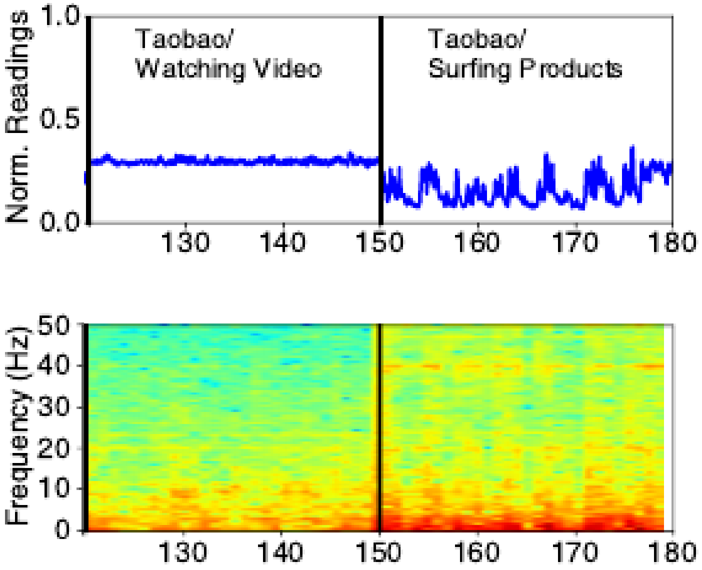
Text chatting | *Surfing moments* | *Video calling* | *Playing games*



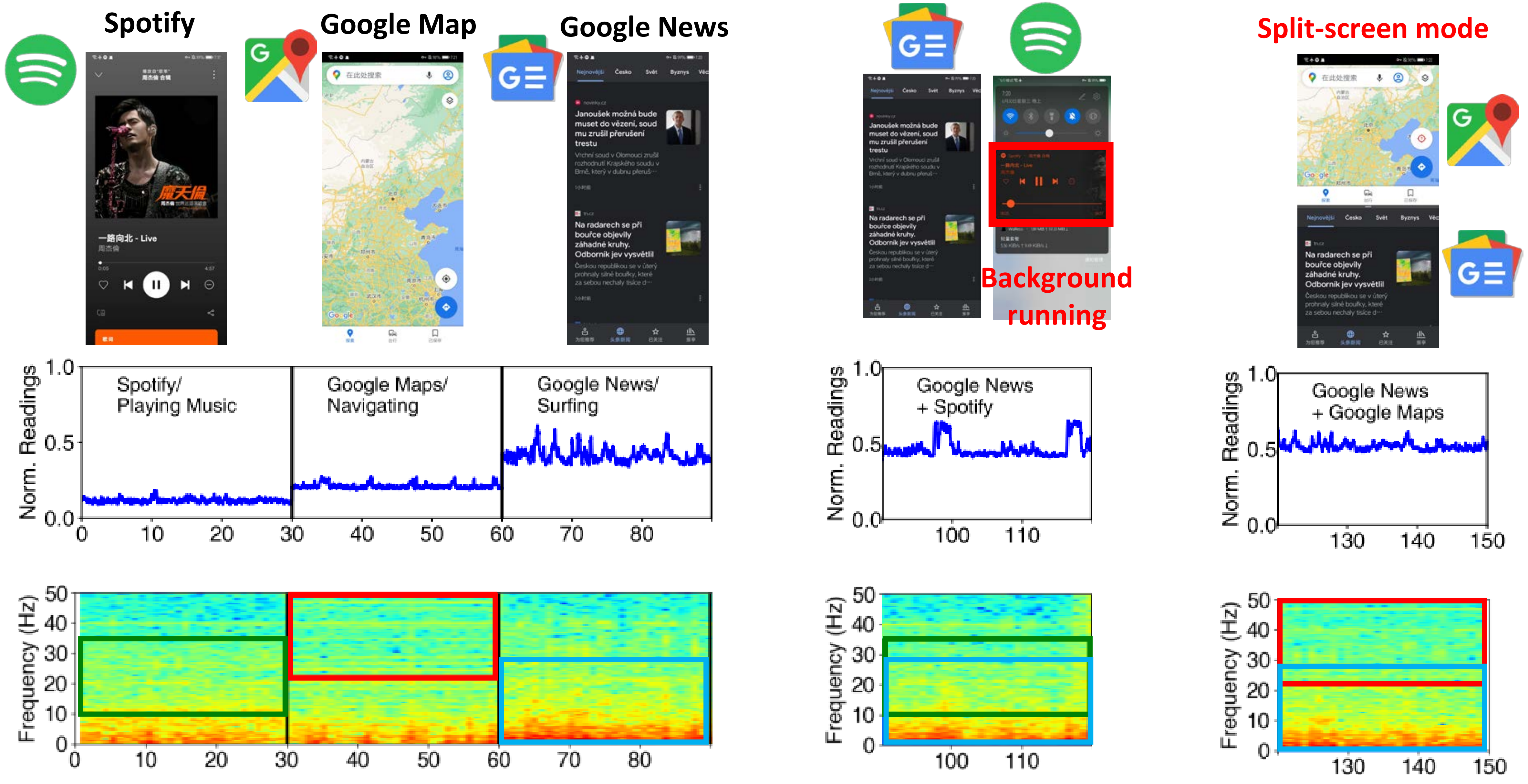


App 2: Taobao

Watching video | *Surfing products*



Preliminary experiment II —— multiple running apps



Outline

Background and Motivation

Related Works and Limitations

Preliminary Analysis

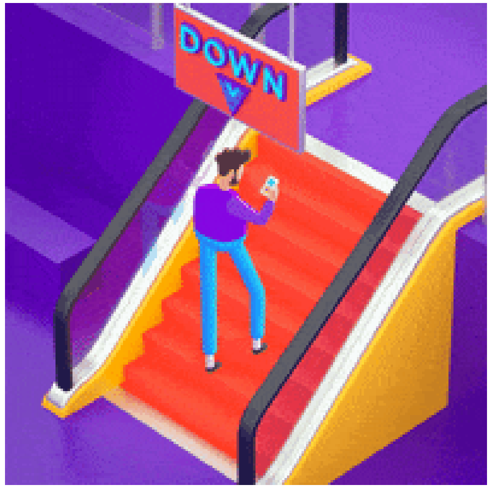
System Design

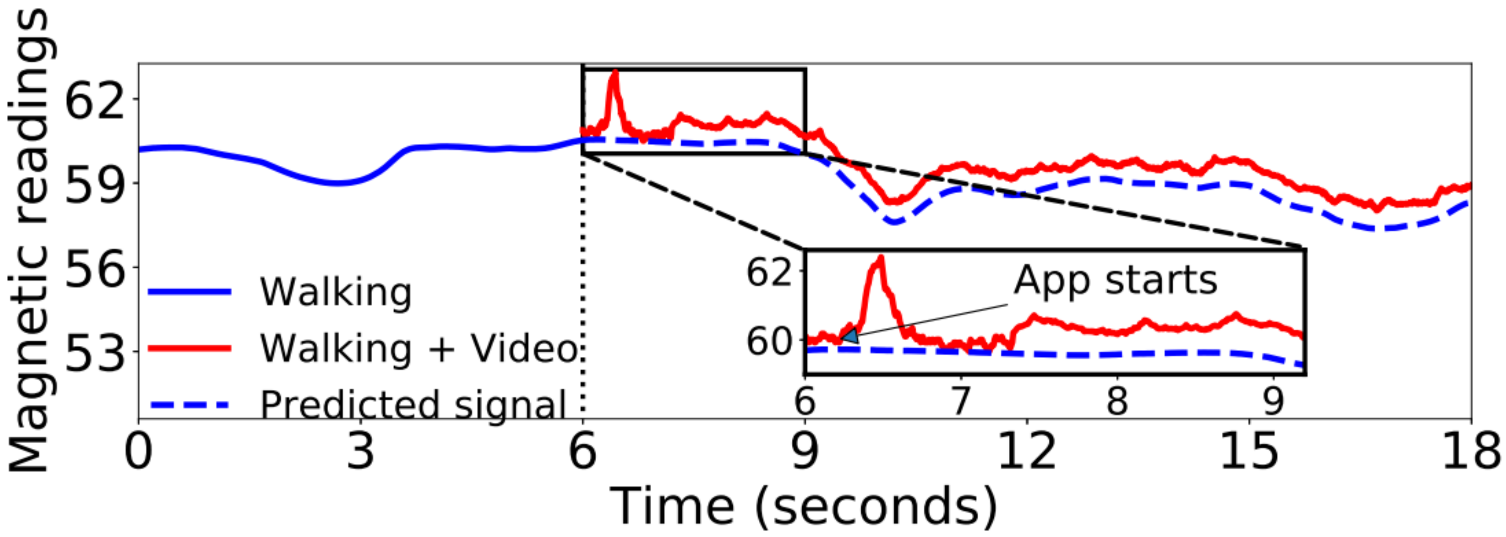
Evaluation

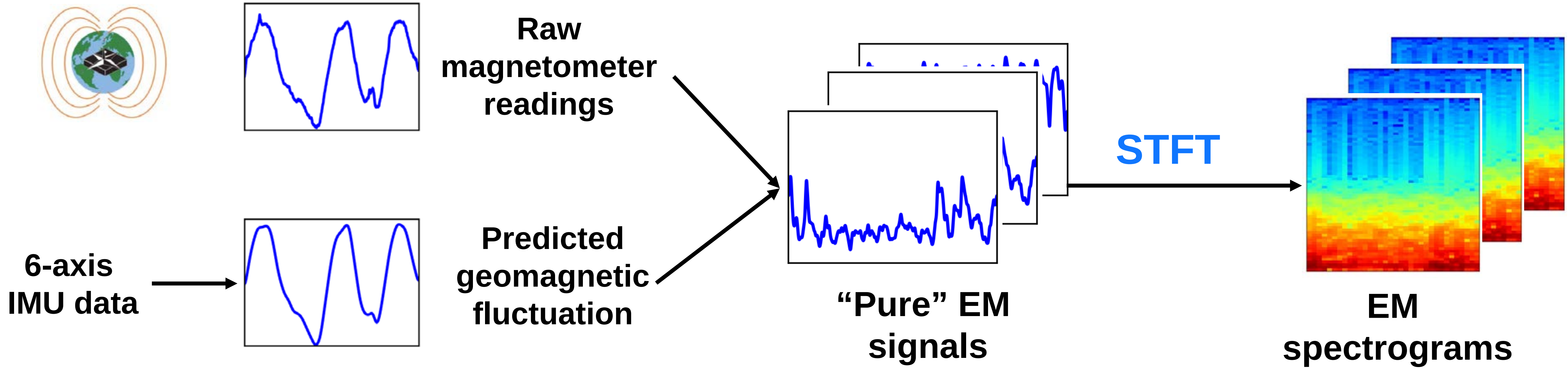
Conclusion

Cancel out the geomagnetic filed signals

Using phones when walking

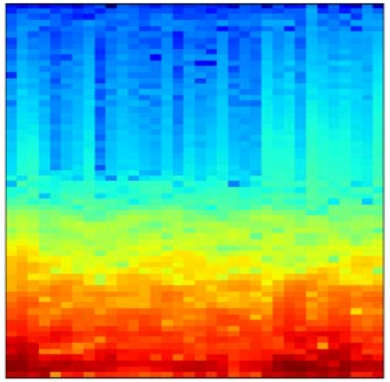




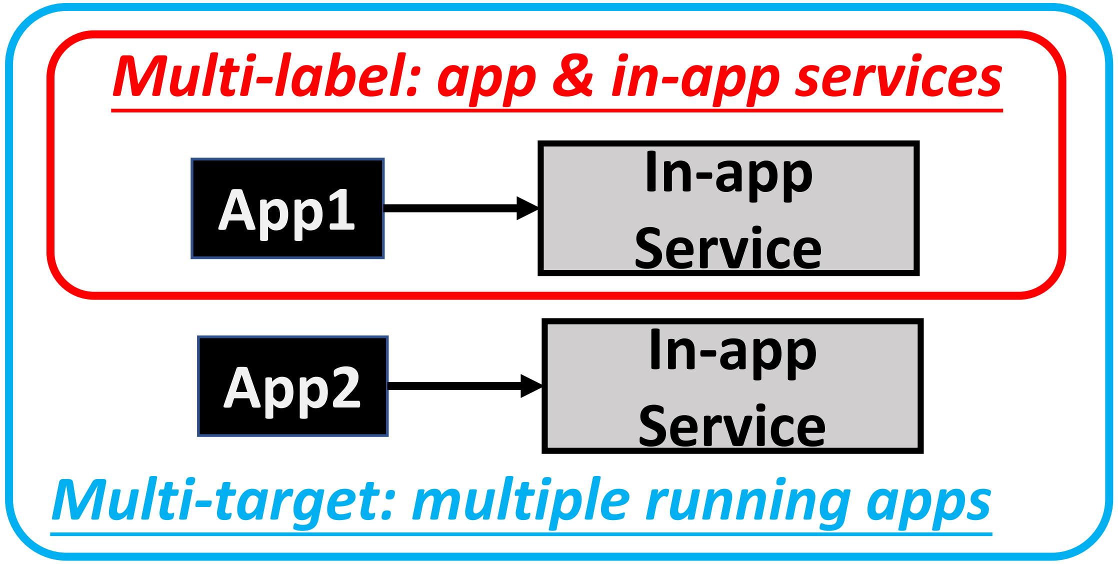


Dataset collection

EM spectrograms



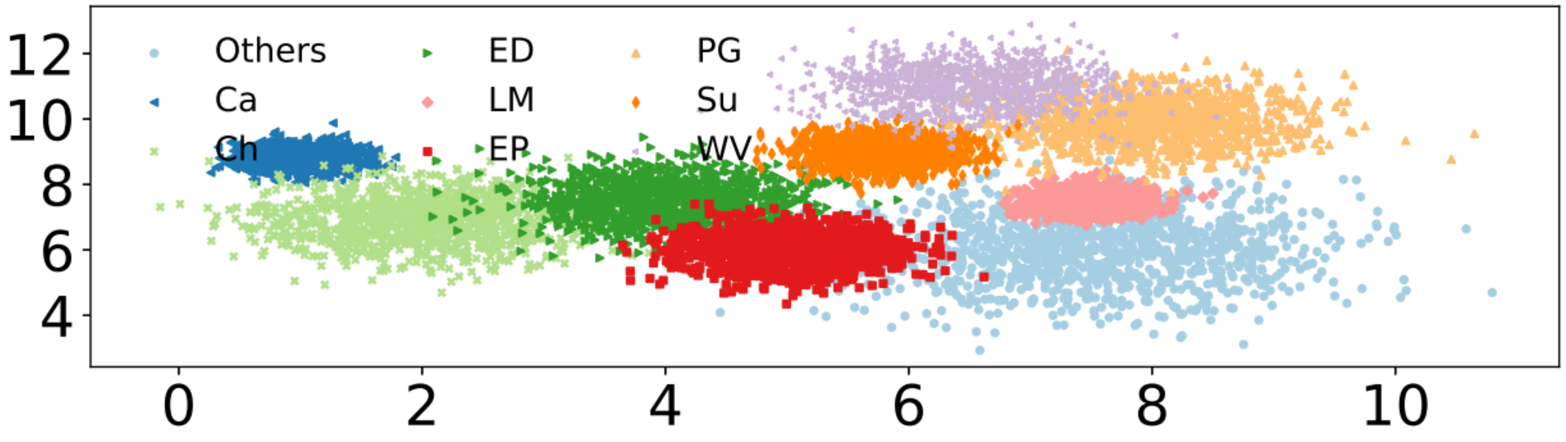
labeling



App types are known!

How about labels of in-app services?

EM signal clusters related to nine types of in-app services



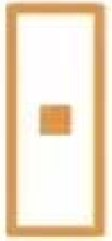
In-app service labels:

- ✓ Ca: video/voice calls
- ✓ Ch: text chatting/typing
- ✓ ED: editing documents
- ✓ LM: listening to music
- ✓ EP: editing photos
- ✓ PG: playing games
- ✓ SU: surfing/reading
- ✓ WV: watching videos
- ✓ Others

How to define the region of each running app?

Our idea:
Region Proposal Network

Ground truth of bounding box (manual labeling)

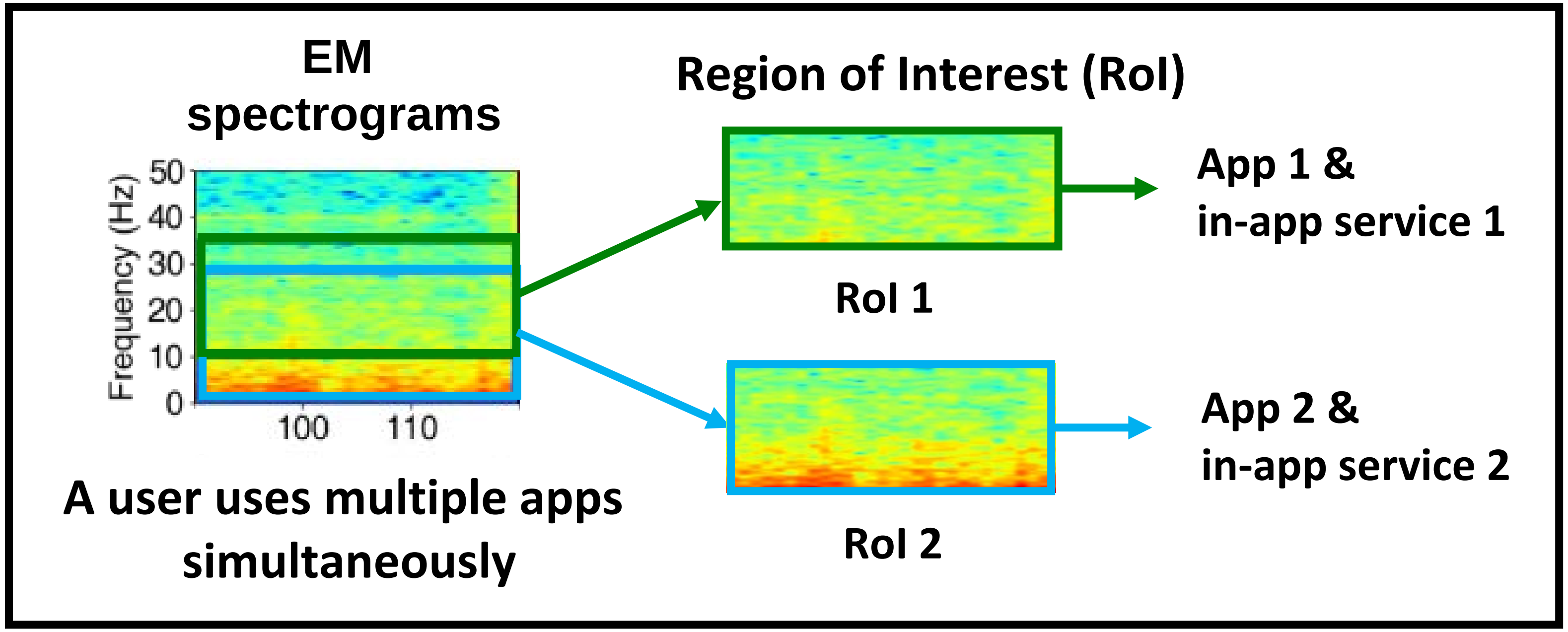


w: *Box width*

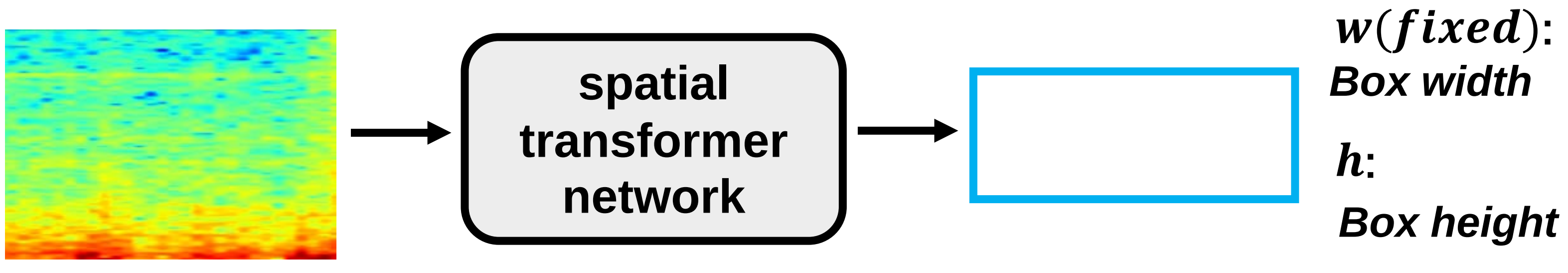
h: *Box height*

x, y: *Box center*

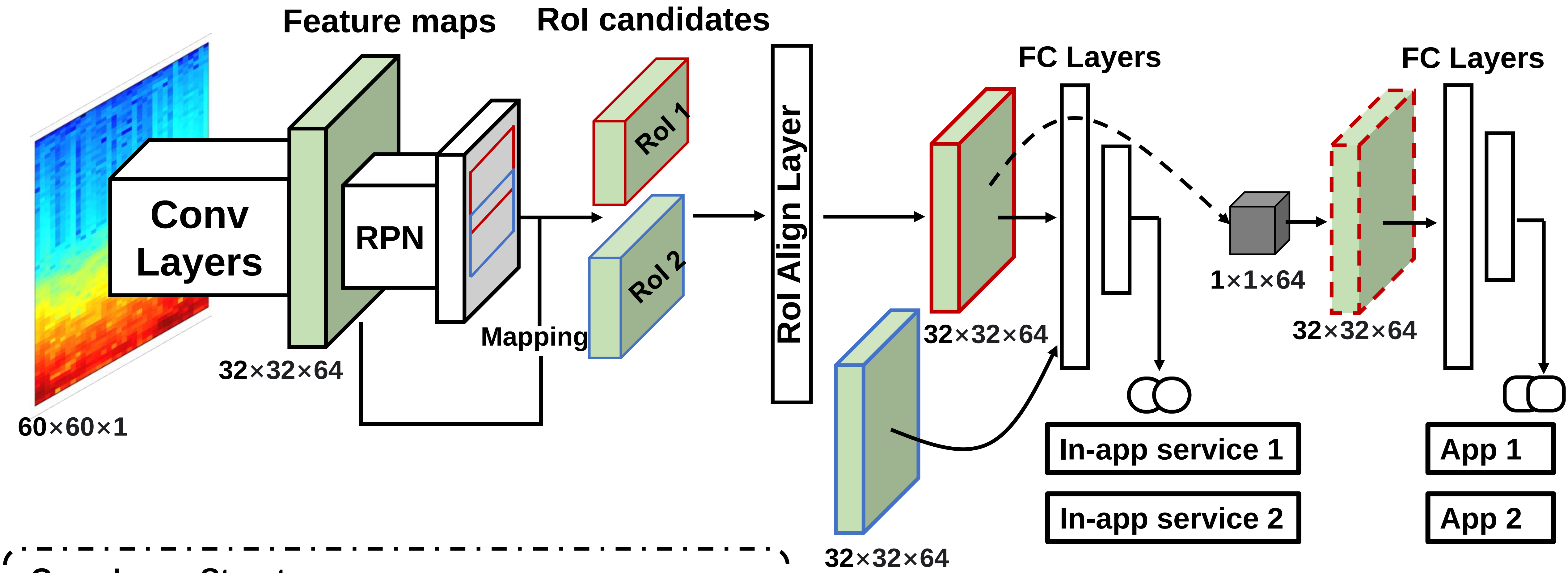
Design the app/in-app classification model:



Determine the bounding box of each single running app with STN



DRCNN: multiple apps/in-app services classification



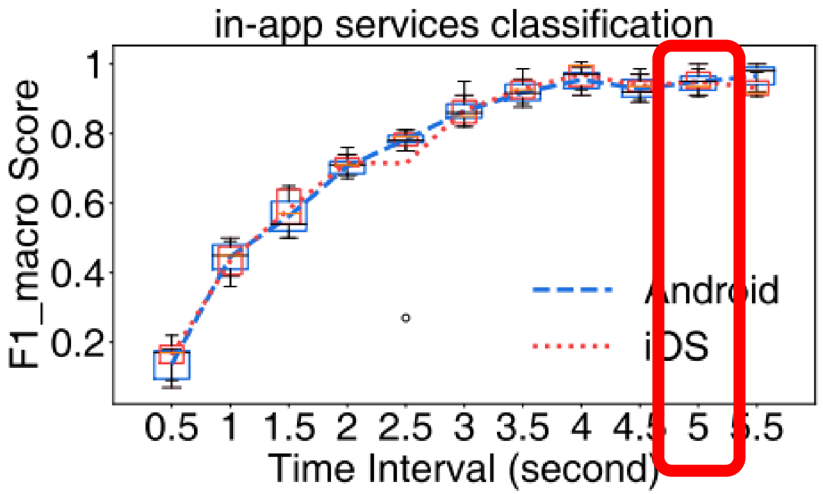
Conv Layer Structure：
Conv2D(32) – BN – ReLU – Conv2D(64) – BN – ReLU

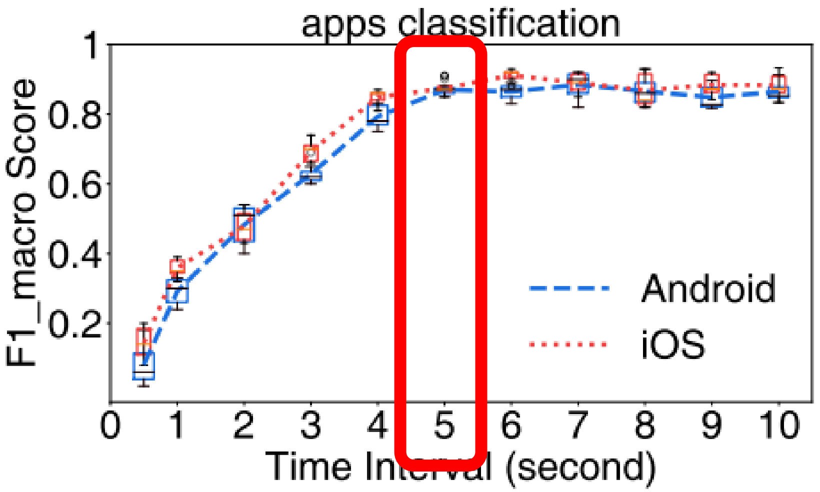
Outline

- Background and Motivation
- Related Works and Limitations
- Preliminary Analysis
- System Design
- **Evaluation**
- Conclusion

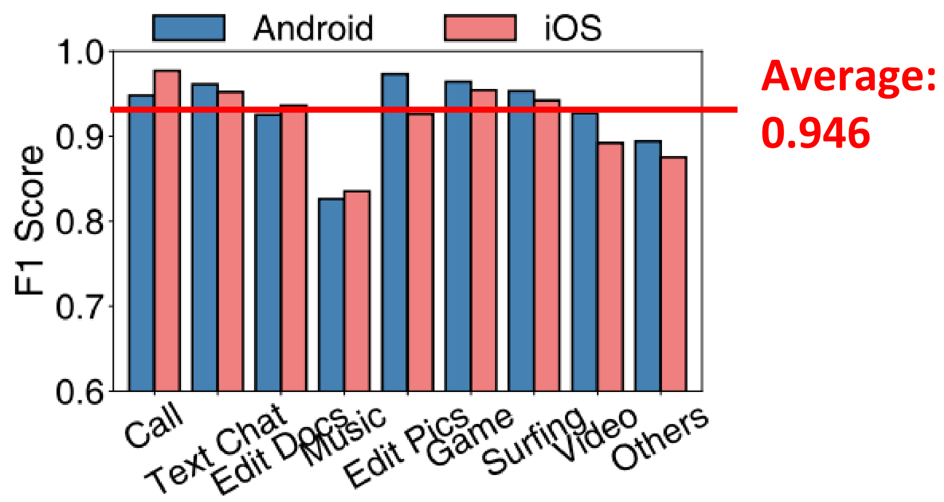
Experiment Results

Determine the time interval length of EM signals:

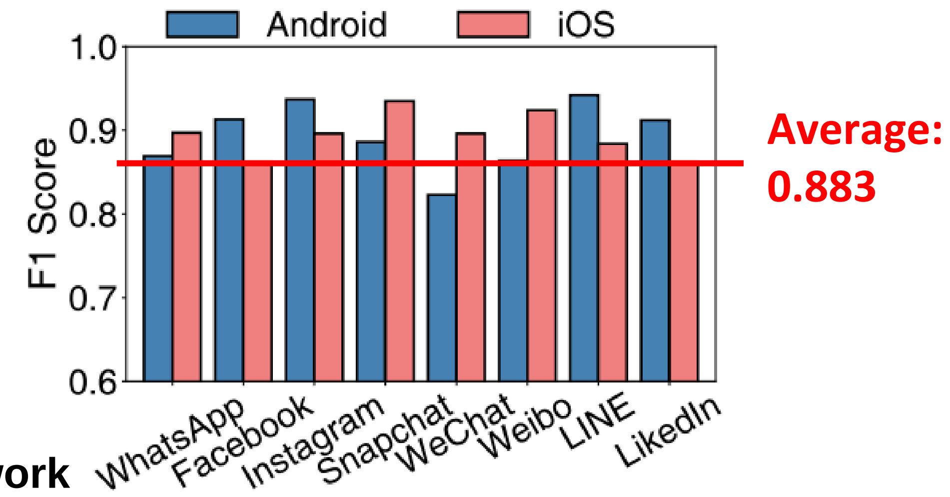




Multiple in-app service classification:



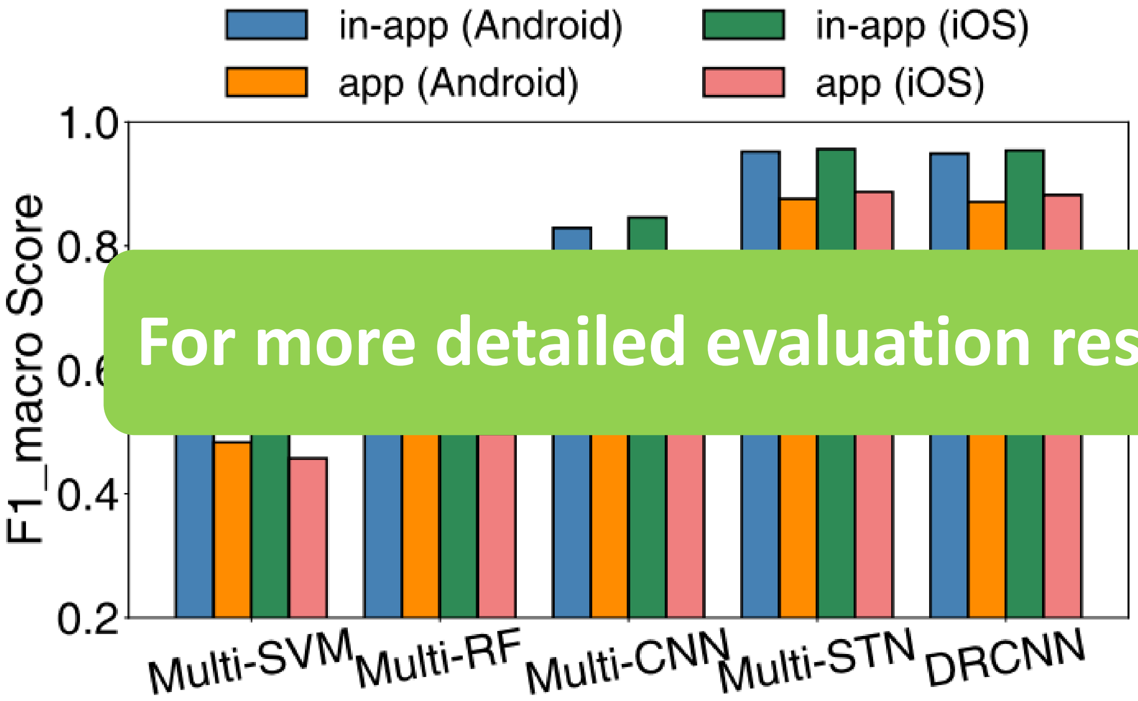
Multiple app classification:



Partly:
Social network

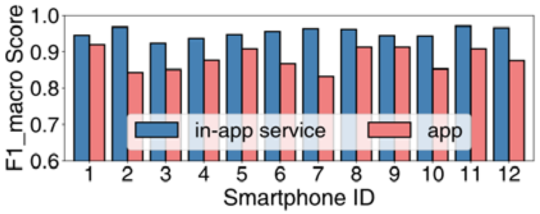
Experiment Results

Comparison of multi-label classification models



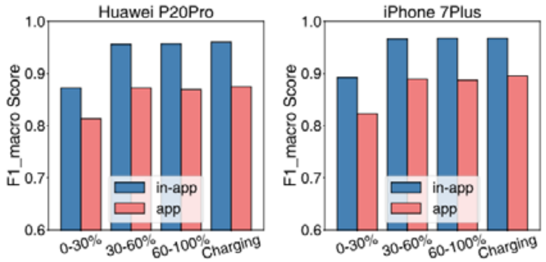
For more detailed evaluation results, please read our paper ☺

Performance on different smartphones：



Against different environments：

Smartphone settings (e.g., battery)：



Outline

Conclusion

- **MagThief** can steal fine-grained sensitive app usage info with the built-in magnetometer readings:

  ✓ We developed a Deep Region CNN (DRCNN) to facilitate the ***multi-target*** and ***multi-label*** classification of multiple running **apps** as well as corresponding **in-app services**.

  ✓ Extensive experiments demonstrated the efficacy of the MagThief, and it achieves high average macro F1 scores of 0.87/0.95 when identifying multiple apps/in-app services respectively.

Questions

- **Q1: The deep learning model you proposed in your paper need to be trained for each unseen device? In other words, how about the performance against untrained devices?**

Thanks for your good question. In fact, there are thousands of mobile phones from different manufacturers on the market, some smartphone manufacturers optimize the app starting process, to let users have a better experience, such as pre-loading/pre-launching the app to short the time of cold starting. So identifying the apps with the launching process should consider the differences of EMI patterns from different smartphones.

However, our work is to identify the EMI signal generated when using the app. When different smartphones perform the same tasks in the same app, the codes they execute are always the same. We consider the two mainstream mobile operating systems, Android and iOS. We trained a classification model for Android phones and another classification model for iPhones. And the Operating System Type information is very easy to obtain. In this way, we can deploy our model on unknown devices, with the operating system type of the target device.

Questions

- **Q2: There are thousands of apps on the market, and your model seems to be able to recognize only a limited number of apps in the training data set. How to identify those apps that are not in the data set?**

Yeah, thanks for your good question. Our proposed model is a supervised learning method, so we indeed need to collect electromagnetic radiation signals when users use various types of apps in advance, and collect this data as training dataset. In this work, we collected the mainstream apps, the top 50 most popular apps from the Google play and Apple store, as our training dataset. So our model currently supports the identification of these 50 mainstream apps.

We also design our model as a multi-label classification model (to identify the running apps and corresponding in-app services), and we defined the limited categories of in-app services, so when we face untrained apps, we can still identify the current using in-app service information from users. In the future work, we will hire more volunteers to collect EMI training dataset of more apps, so that our model supports the identification of more apps.

Questions

☐ Q3: How spatial transform network works? And why Region Proposal Network can find out the multiple running apps?

Spatial transform network can help us find the sub frequency band that is most conducive to identifying the current running app/in-app service from the entire EM spectrogram. Similar to the object detection problem in computer vision, for example, there is a cat in an image, but this cat only occupies a small area in the entire image, so STN can be used to find the smallest area that contains the cat, in this way we can focus on this small area instead of the entire image to identify objects.

When we use STN to find out the bounding box of each running app or in-app service in the EM spectrogram. We can train an Region Proposal Network. This Region Proposal Network is used to find the whole sub frequency bands that most likely to contain the running app or in-app service. If multiple boxes are found, it means that the current device is running multiple apps. In this way, we can realize the multi-target classification.