MobiCom: U: Broadcast LTE Data Reveals Application Type

Arjun Balasingam
Stanford University
arjunvb@stanford.edu

Manu Bansal
Uhana Inc.
manub@uhana.io

Rakesh Misra
Uhana Inc.
rakesh@uhana.io

Rahul Tandra
Uhana Inc.
rahul@uhana.io

Aaron Schulman
UC San Diego
schulman@cs.ucsd.edu

Sachin Katti
Stanford University
skatti@cs.stanford.edu

ABSTRACT

The rapid growth in mobile connectivity is enabling phones to support a wide range of societally-important applications. In this work, we show that broad classes of popular mobile applications have distinct radio resource allocation signatures. Using this insight, we design a mobile application classifier, and demonstrate that (1) an application can infer its own type solely from its resource allocation patterns, and (2) *anyone* can accurately infer the type of application being served by each session on a particular cell tower. We present our findings by showing the breakdown of applications being served by an LTE base station belonging to a Tier 1 US provider in downtown Palo Alto. Our work encourages an open discussion about LTE standards, and whether they might need to be enhanced to mask features that can be exploited to infer application type from signals broadcast over the air.

CCS CONCEPTS

• **Networks** → **Network resources allocation**; **Security protocols**; **Mobile and wireless security**; *Mobile networks*; *Packet classification*;

KEYWORDS

Application Classification; Cellular; LTE; PHY; Resource Allocation

1 INTRODUCTION

Internet-connected mobile devices play indispensable roles in our lives today. From online educational content, to video streams carrying sports, news, and entertainment, consumers are increasingly accessing cloud-based data on their wireless mobile devices. The amount of traffic on wireless mobile networks is expected to increase sevenfold over the next five years [1].

Conventional downlink scheduling algorithms at LTE base stations (e.g. round-robin, proportional-fair) [2] are agnostic to application type. In this paper, we show that surprisingly, resource allocation signatures repeatably capture the traffic arrival patterns demanded by different types of applications. We use this insight to demonstrate that it is possible to *infer the type of application being hosted by any LTE session* from only its radio resource allocation patterns. We present the design of an application classifier and quantify application usage on eNB in a network-congested region of Palo Alto.

Prior work, such as LTEye [3] and piStream [6], has analyzed broadcast LTE data to study network characteristics like spectrum utilization and inter-cell interference, and improve user quality of experience for video streaming and web browsing applications. Previous attempts at application classification from network data have employed deep packet inspection (DPI) to analyze data within packets to select application type [4]. In this work, we present a technique to classify applications from broadcast radio-layer (LTE) data.

We make two signficant contributions:

(1) **Local application inference.** We show through controlled experiments that an application can infer its *own* type from local resource allocation data that it receives at the client phone's modem. We introduce the `PROMINENCE` metric, and use it to characterize different types of applications. We then present a heuristic based only on `PROMINENCE`, that can accurately identify file download, video streaming, video conferencing, and web browsing applications from one another.

(2) **Cell-wide application inference.** We extend the findings in (1), and demonstrate that *anyone* can identify the type of application being served by each live radio session on a cell tower, by passively sniffing a broadcast LTE channel, without any dependency on a cellular carrier, application type, or eNB vendor.

Our work has important consequences. It encourages a discussion about the privacy implications of having open access to application type without any privileged information from the network or application provider. Such information can be used to understand a user's application preferences; for instance, a classifier like the one presented in this work allows anyone to draw conclusions about the mix of applications run by a user over some timespan. Additionally, hackers can use application type to target attacks during particular segments of a session; for example, after inferring that a user is engaged in a video conference call, one could predict that account authentication or payment details will be transmitted at the beginning of a call or after a re-connection, which can be used to disrupt the session. Thus, discussions around improving the security of data broadcast over LTE are necessary to prevent malicious attacks on applications carrying user data.

MobiCom '17, October 16–20, 2017, Snowbird, UT, USA

ACM ISBN 978-1-4503-4916-1/17/10.
https://doi.org/10.1145/3117811.3131254

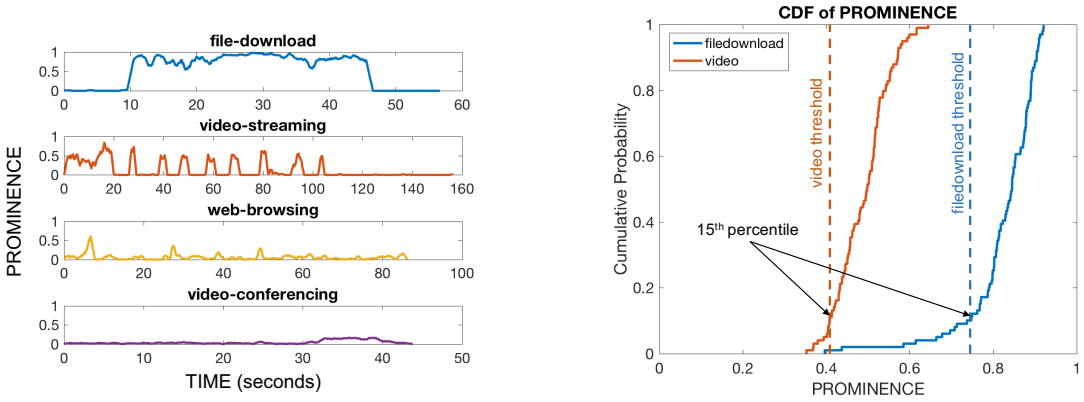

Figure 1: Different applications have unique PROMINENCE signatures. [left] PROMINENCE time series signatures for 4 different application types. [right] CDF of PROMINENCE showing reliability of metric.

2 DESIGN OF APPLICATION CLASSIFIER

We designed our application classifier using radio-layer data supplied by QXDM [5], a software package that logs messages received by a phone's modem. Our experimental setup consisted of an LG android phone (running one of the applications under investigation) connected to a PC running QXDM.

2.1 The PROMINENCE Metric

Our classifier is based on data transmitted on LTE's Physical Downlink Control Channel (PDCCH), a broadcast channel that carries Downlink Control Information (DCI) messages, specifying information about the encoding rate, modulation, and allocation of resource blocks scheduled in each millisecond of transmission.

We construct the following simple metrics for each session from the raw data exposed by the DCIs logged in QXDM:

- SCHEDTIME: the total number of milliseconds where that session was scheduled at least one resource
- SESSDUR: the total duration of that session in milliseconds
- PROMINENCE: schedule count normalized by session duration (i.e. SCHEDTIME/SESSDUR)

The PROMINENCE metric—a very simple quantity to compute—is surprisingly sufficient to classify broad categories of applications being run on mobile phones. It leverages the insight that different applications have distinct traffic arrival patterns (e.g. files are downloaded in a continuous stream, while videos are downloaded in segments), and abstracts out other session-specific quantities like the volume and modulation of the transmitted data. External factors that might affect the quality of experience a user sees at the application level, such as poor RF channel conditions, will not impact the overall PROMINENCE of a session[1]. In the rest of this section, we characterize different applications in terms of their PROMINENCE and present the design of our classifier.

2.2 Classifier Design

As we see in Figure 1 on the left, different classes of applications have unique PROMINENCE signatures. A prominence signature is a timeseries of the PROMINENCE metric, computed in a moving window of 1 second.

We can characterize each application type in terms of their prominence signatures.

- **File download** sessions are full-buffer and consequently highly prominent—they are scheduled resources in on average 80% of the transmission intervals in a second.
- The PROMINENCE signature of a **video playback** session captures the fairly regular periodicity of video streaming algorithms, characterized by segment downloads (buffer building) followed by idle periods.
- **Web browsing** (loading a sequence of web pages) can be viewed as a stream of very brief file downloads.
- **Video conferencing** sessions have consistently low (nonzero) overall PROMINENCE, since data is transmitted as needed per millisecond (instead of as larger chunks).

The analysis of prominence signatures suggests that a simple thresholding of the PROMINENCE metric for sessions is sufficient to distinguish between file download and video sessions. We corroborate this observation by running 100 file download and 100 video sessions at different times of day and under different channel conditions. CDFs of the PROMINENCE metric are shown in Figure 1 on the right.

First, we see that both distributions are tight—the spread in PROMINENCE for each application type is less than 25%—indicating that the PROMINENCE score is a highly repeatable metric that can be used reliably to identify file download from video. Second, video and file downloads have clearly different distributions, suggesting that a very simple thresholding by the PROMINENCE metric is sufficient to distinguish between file downloads and videos. Even a highly conservative choice (e.g. 15th percentile) of PROMINENCE threshold allows us to separate file download traces from video traces accurately.

Expanding on this insight from the PROMINENCE metric, we can design an application classifier to identify file download, video streaming, video conferencing, and web browsing sessions. File downloads can be filtered based on high PROMINENCE scores (e.g. PROMINENCE > 0.4) and a large fraction of 1-second windows with

[1]We experimentally validate this claim, as discussed in Section 2.2.

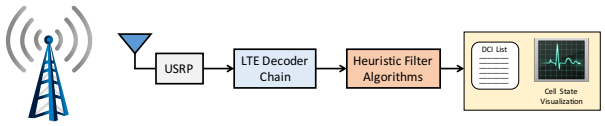

Figure 2: eNBsniffer block diagram. To sniff the PDCCH of any cell tower, we tune eNBsniffer's USRP to the trasmit frequency of that cell tower, collect raw IQ samples, process with a standard LTE decoder, and prune out false positives using a series of heuristic filter algorithms.

nonzero PROMINENCE. Web browsing sessions also have a large fraction of 1-second windows with nonzero PROMINENCE, but have low overall PROMINENCE (e.g. PROMINENCE < 0.15). Video streaming has low overall prominence, but high prominence in 1-second windows (due to segment downloads), while video conferencing has low overall PROMINENCE as well as low PROMINENCE in 1-second windows (since data is encoded on a per-millisecond frame basis).

3 CELL-WIDE APPLICATION INFERENCE

In this section, we leverage the classifier presented in Section 2 and show that it is possible to tag each user connected to a particular cell tower with the category of application being run on their phone. We then show a breakdown of the types of applications being served by an eNB in downtown Palo Alto.

3.1 eNBsniffer

In order to get a cell-wide view of resource allocation, we built eNBsniffer, an LTE PHY-layer sniffer, similar to LTEye [3]. The eNBsniffer system consists of an off-the-shelf software-defined radio (USRP), a standard LTE decoder chain from MATLAB's LTE toolbox, and a series of heuristic filters that we designed. One of its advantages is that it is completely passive—it does not require any special handling or access to privileged information that only carriers can sanction. Further, it listens *only* on the PDCCH, which broadcasts metadata to all users, and does not sniff on any data sent to specific users.

From eNBsniffer, we process DCI messages sent to *all* users connected to the LTE band to which we tune the USRP. So, it provides us with the same information about the PDCCH that QXDM supplies, but for every user connected to the eNB, instead of for just our test user.

eNBsniffer's block diagram is shown in Figure 2. First, we tune the USRP to the trasmit frequency of the eNB of interest. We then collect the IQ samples and extract the DCIs using some standard modules in MATLAB's LTE toolbox. The DCIs don't explicitly include the identifier for each user. A user's device decodes DCIs addressed to itself by scrambling a 16-bit key with its own identifier (RNTI). The challenge is that eNBsniffer does not have access to the users' RNTIs. We overcome this problem by exhaustively searching over all 2^{16} possible RNTIs. While this approach guarantees that all transmitted DCIs are decoded, on occasion, it can match the scrambled key with an RNTI by coincidence, yielding false positives.

Therefore, we add another pruning module—a series of filter algorithms—to our eNBsniffer system. The pseudocode for this filter block is shown in Algorithm 1. There are 4 key filters that we apply to prune out the false positive DCIs:

Algorithm 1: eNBsniffer DCI Pruning Heuristics

```
collect samples, perform exhaustive search, and decode DCIs;
Function FilterChain():
    apply duplicate, prominence, power, and conflict filters;
Function DuplicateFilter():
    discard duplicates, i.e. identical (RNTI, TTI, alloc);
Function ProminenceFilter():
    for u ∈ user_list do
        if PROMINENCE (u) < prom_threshold then discard ;
    end
Function PowerFilter():
    for d ∈ dci_list do
        if PDSCHpower(d) < pdsch_threshold then discard ;
        if CCEpower(d) < cce_threshold then discard ;
    end
Function ConflictFilter():
    for t ∈ tti_list do
        if length(dci_list[tti = t]) > 1 then
            sort dci_list by PROMINENCE;
            keep most prominent DCI;
    end
```

- **Duplicate filter.** This filter flags DCIs with duplicates if they occur in they belong to the same user, occur in the same time slot (TTI), and have identical resource allocations in that TTI. Duplicate DCIs are discarded so each DCI in the final list is unique.
- **Prominence filter.** This algorithm identifies all unique users from the list of DCIs, and prunes out those with a low PROMINENCE score.
- **Power filter.** In this filter, we inspect the raw signal power of the time-frequency elements carrying the data payload (PDSCH) being referenced by each DCI[2]. DCIs pointing to elements with low power are discarded.
- **Conflict filter.** This final filter considers resource allocation conflicts between all DCIs belonging to the same TTI. In particular, a conflict is flagged if mutliple DCIs point to the same resource block in the same TTI. We resolve conflicts by keeping the DCI belonging to the user with highest PROMINENCE.

Because our filter algorithms are heuristics, we needed to assess eNBsniffer's accuracy, and specifically quantify its false positive and false negative rate. We ran several field experiments to verify the accuracy of DCIs recovered by eNBsniffer against the ground-truth DCI contents reported for our own users by QXDM. First, we added our own traffic on an LTE picocell that we deployed in our lab space; our false negative error was close to 0% and our false positive error was less than 1%. On multiple production LTE base-stations, we saw a false negative error of 5% and false positive error < 1% for moderately high SNR RF conditions. Given our performance on

[2]Note that we only look at the power characteristics of the signal; the PDSCH channel is encrypted, so we cannot decode it.

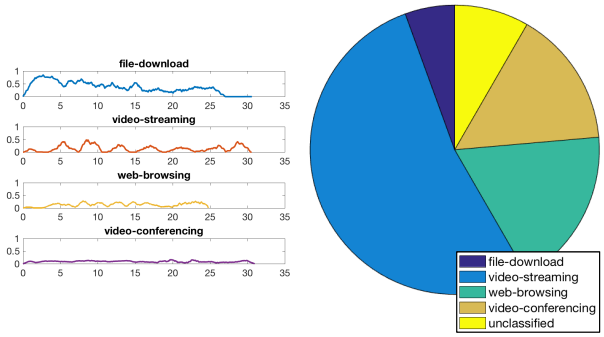

Figure 3: PROMINENCE analysis conducted using eNBsniffer. [left] PROMINENCE signatures for applications classified from eNBsniffer data. [right] Breakdown of application usage on a congested eNB in downtown Palo Alto.

the picocell and from our analysis of the raw PDCCH symbols, we believe that most of the 5% false negative error can be made up for with better frontend hardware. Additionally, a 5% error in volume recovered will not affect the classification accuracy, for the broad class of applications discussed in Section 2.

3.2 Application Usage in Palo Alto

With eNBsniffer, we have a means to passively sniff the LTE PDCCH channel, and collect resource allocation patterns for all users connected to a cell tower—without any dependency on the network carrier. We can apply our application classifier to tag each session with its application type. We now show the results of this exercise on a base station in downtown Palo Alto.

Using eNBsniffer, we obtained the resource allocation signatures of all sessions being served by an eNB in Palo Alto for a few minutes during peak hours. We applied the classifier discussed in Section 2, and the results are shown in Figure 3. About 68% of the traffic was video (53% video streaming, 15% video conferencing). Six percent were file downloads and 18% were web browsing sessions. 8% of sessions could not be classified by our heuristic. This breakdown in usage on this one eNB is consistent with the application usage projected in Cisco's Virtual Networking Index (VNI) Forecast [1].

4 DESIGNING FOR SECURITY

In the preceding sections, we a proposed PROMINENCE-based classifier of application type, and presented the implementation of a system that categorizes applications from broadcast data. The implications of a passive system enabled by eNBsniffer, as presented in Section 3 are significant: without any privileged network information, one can see exactly how the eNB scheduler allocates resources to different flows, and draw inferences about particular sessions or users. Hackers could now use such a system to isolate particular applications to attack.

The first step to devising secure LTE protocols is to identify features that might compromise user identity. We can then develop schemes to mask these features. For instance, in this paper, we showed that the PROMINENCE metric, which is a function of resource allocation frequency, is a highly indicative feature of application type.

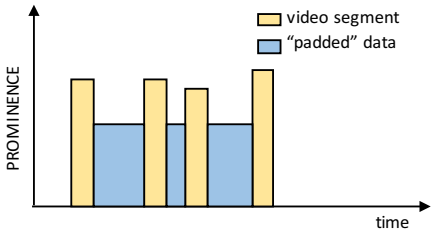

Figure 4: Application padding. Shown here for video streaming.

One solution is to "pad" sessions, as depicted in Figure 4 for a video streaming, application, so they all have similar PROMINENCE. This would entail filling idle periods in sessions with data. Ultimately, all sessions would end up looking like file downloads, with a time series similar to the one in Figure 1. A concern with this design is that radio resoures—which are becoming increasingly scarce with the growth in mobile connectivity—would be squandered. To mitigate the effects, we can find an optimal padding frequency that minimizes overhead while masking distinct classes emerging from PROMINENCE time series.

5 CONCLUSIONS

This initial work shows us that a simple metric derived only from broadcast resource allocation data reveals the type of application supported by a radio session. We introduced the PROMINENCE metric, used it to describe popular classes of applications, and applied it on data sniffed over the air from a congested eNB in downtown Palo Alto.

As future work, we would like to further validate the accuracy of our heuristic on a broader class of applications, on different types of video and file download clients, and on eNBs with different schedulers. As we expand the application space for this heuristic, we might also consider machine learning-based approaches to identifying hidden patterns in PROMINENCE signatures. Additionally, we hope to more extensively discuss the privacy and security implications of our findings, and propose more modifications to LTE standards that conceal features of application type from unencrypted signals broadcast over the air.

REFERENCES

[1] Cisco. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016–2021 White Paper. http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html, 2016],.

[2] M. Kawser, H. Farid, A. Hasin, A. Sadik, and I. Razu. Performance comparison between round robin and proportional fair scheduling methods for lte. In *International Journal of Information and Electronics Engineering*, 2012.

[3] S. Kumar, E. Hamed, D. Katabi, and L. E. Li. LTE Radio analytics made easy and accessible. In *Proc. ACM SIGCOMM*, 2014.

[4] S. Lee, H.-c. Kim, D. Barman, S. Lee, C.-k. Kim, and T. T. Kwon. Netramark: A network traffic classification benchmark. In *ACM SIGCOMM Computer Communication Review*, 2011.

[5] QUALCOMM. Qualcomm extensible diagnostic monitor. https://www.qualcomm.com/documents/qxdm-professional-qualcomm-extensible-diagnostic-monitor.

[6] X. Xie, X. Zhang, S. Kumar, and L. E. Li. piStream: Physical layer informed adaptive video streaming over lte. In *Proc. ACM Conference on Mobile Computing and Networking (MobiCom)*, 2015.