Using Big Data for More Dependability: A Cellular Network Tale

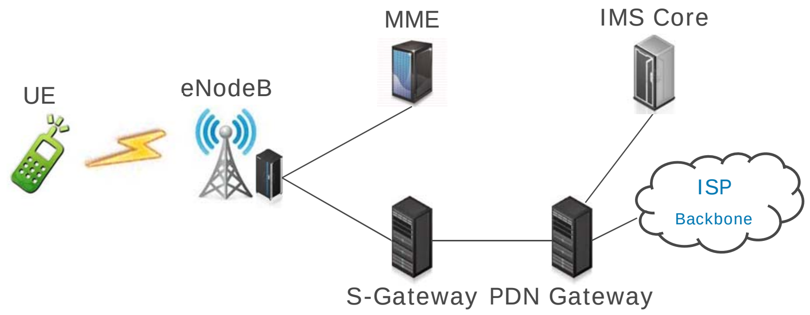
Nawanol Theera-Ampornpunt, Saurabh Bagchi
Purdue University
Kaustubh Joshi, Rajesh Panta
AT&T Labs Research

Motivation

- Large infrastructures instrument network performance and user activities
  - Web analytics record user clicks, page dwell time, etc., to understand page traffic
  - Cellular networks collect information about bandwidth usage, handovers, signal strength, etc., to analyze network performance
- The collection of such large quantities of analytics has been called “Big Data”
- These measurements are often used for offline analysis
- We explore how big data can be used in real time to improve dependability and user experience in the context of cellular networks

Background

- LTE architecture
  - User Equipment (UE) is connected to a cell sector (also referred to as “cell”) in a base station
  - A base station, called eNodeB, can have multiple sectors
  - Cellular traffic pass through Serving Gateway (S-Gateway) and Packet Data Network Gateway (PDN Gateway) to external network (the Internet)



Mechanisms that Improve Reliability

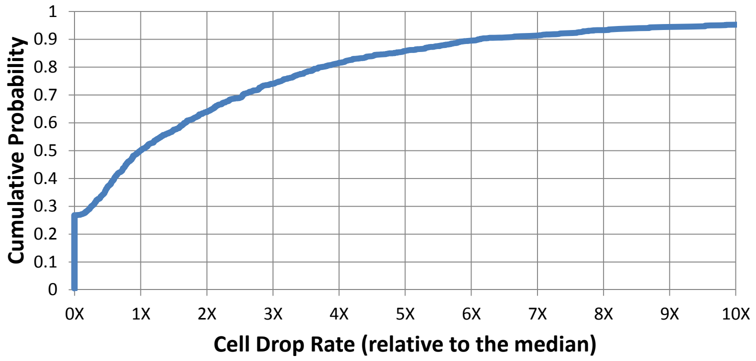
- There are methods that can improve reliability or user experience, but cannot be used *all the time* due to some cost:
- Switching to an older technology
  - Older technologies tend to be less congested than newer ones
  - Cost: lower bandwidth and/or missing features
- Prefetching
  - For applications such as web browsing and audio/video streaming
  - Cost: wasted bandwidth if prefetched content is not consumed
- Voice call auto-reconnecting
  - Makes reconnection more seamless for both parties
  - Cost: user inconvenience if reconnection takes long
- Need to identify conditions where benefits will outweigh the costs

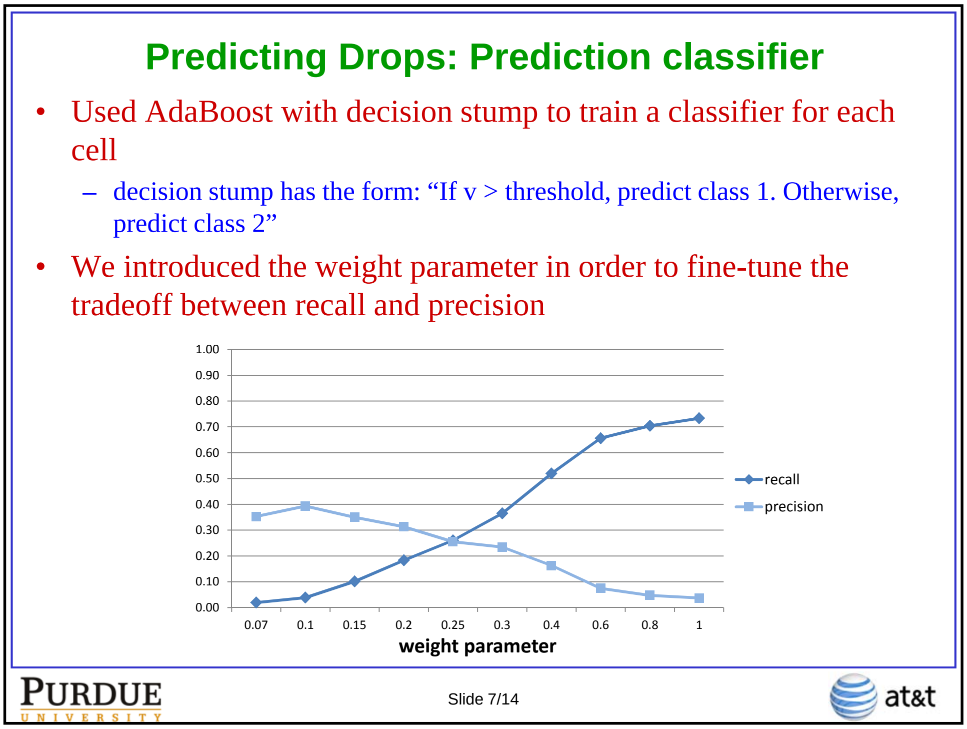
Case Studies

- Data source
  - Real data collected from in a major U.S. 3G cellular network
    - Data are collected at the base stations and aggregated at the Radio Network Controller (RNC), which manages multiple base stations
  - Device and user identifiers are anonymized
  - Contains low-level performance events
    - connections/disconnections
    - cell ID
    - current download/upload throughput
    - cell load
  - 250 metrics total
- We divide data into windows and take the aggregate functions (e.g., avg, count, max, etc.) of each metric

Predicting Drops: Call Drop Data

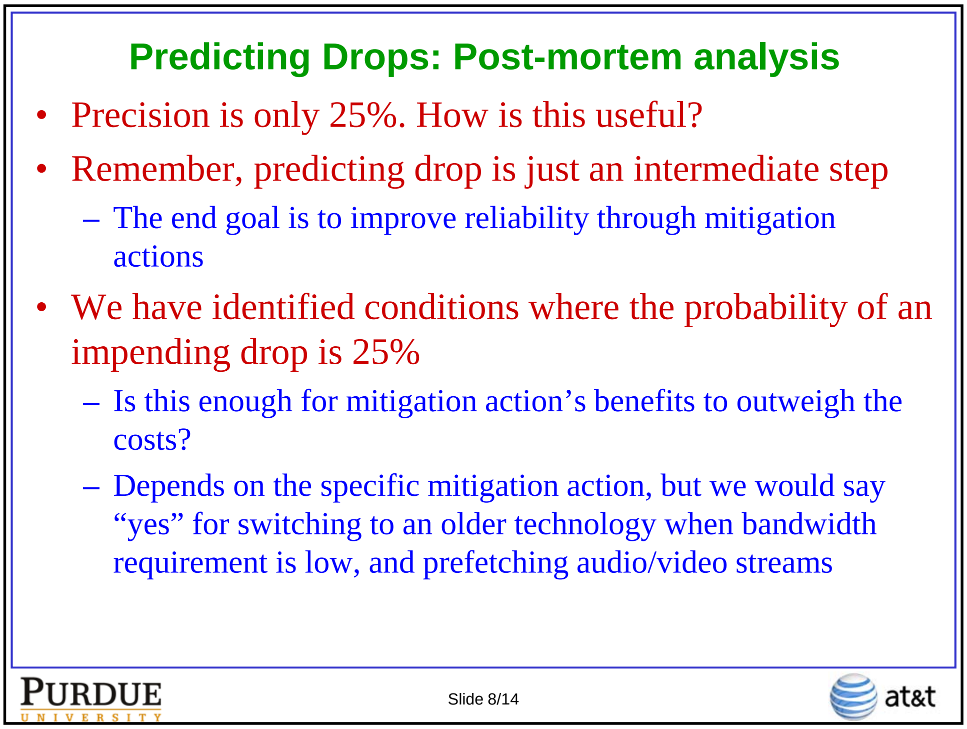
- Predict disconnections (both voice and data)
- Initiate mitigation actions such as switching to older technology and prefetching
- How to partition the models?
- Drop rate of 1,095 cells in the same RNC





Predicting Drops: Prediction classifier

- Used AdaBoost with decision stump to train a classifier for each cell
  - decision stump has the form: “If v > threshold, predict class 1. Otherwise, predict class 2”
- We introduced the weight parameter in order to fine-tune the tradeoff between recall and precision



Predicting Drops: Post-mortem analysis

- Precision is only 25%. How is this useful?
- Remember, predicting drop is just an intermediate step
  - The end goal is to improve reliability through mitigation actions
- We have identified conditions where the probability of an impending drop is 25%
  - Is this enough for mitigation action’s benefits to outweigh the costs?
  - Depends on the specific mitigation action, but we would say “yes” for switching to an older technology when bandwidth requirement is low, and prefetching audio/video streams

Predicting Drops

- Top two metrics that influence the classification the most:
  - Number of records for a device where the upload throughput is zero, referred to as A1
  - Sum of cell’s transmit power within a time window, referred to as A2

A1	A2	Fraction of failure data points
≤	≤	0.51X
≤	>	2.04X
>	≤	1.82X
>	>	40X
Any	Any	X

Predicting Drops

- Upload throughput is reported every 2 seconds, even if it is zero
- Cell’s transmit power is related to the current load on the cell, which is correlated with drops

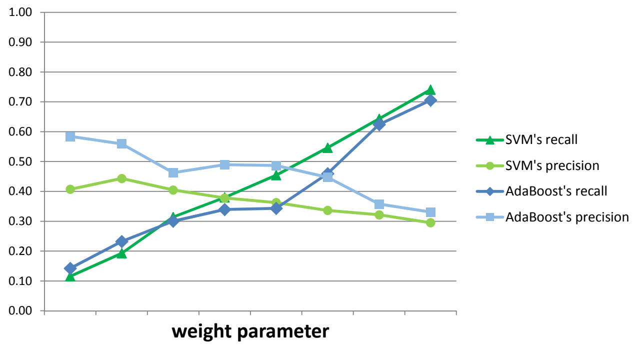
A1	A2	Fraction of failure data points
≤	≤	0.51X
≤	>	2.04X
>	≤	1.82X
>	>	40X
Any	Any	X

Predicting Drop Duration

- Given that a disconnection has occurred, what is the earliest time the connection can be reestablished
  - We refer to the duration between these two events as “drop duration”
- If user was in a voice call and drop duration is short (e.g., <10 seconds), the call can be paused instead of dropped
  - Both the disconnected party and the call server need to agree on the same action
  - Online predictor sends prediction to both, and send them an update when the prediction changes
- It is not possible to determine the earliest time the connection can be reestablished from collected data
  - The network is not aware of unsuccessful reconnection attempts
  - Devices do not always attempt to reconnect immediately after a drop
  - We use first successful reconnection after a drop to compute drop duration

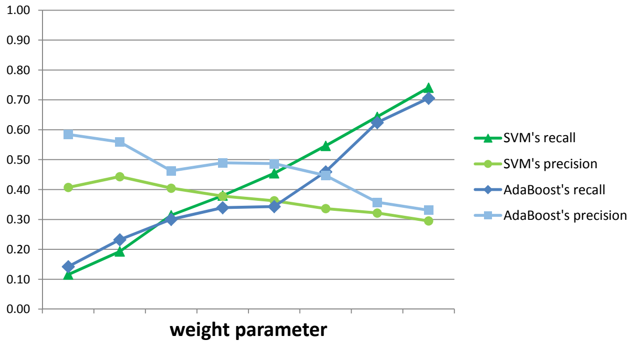
Predicting Drop Duration

- For each drop, we use data from the 1-minute window leading up to, but not including the drop
- We compare accuracy of support vector machine (SVM) and AdaBoost with decision stump
  - one model is trained for the whole RNC



Predicting Drop Duration

- AdaBoost performs slightly better than SVM
  - it achieves both recall and precision of 45%
- Top metric: download throughput
  - high throughput correlate with short drops



Avenues for Further Work

- Real-time data access still not available
  - Need to protect user privacy
- Data volume
  - Need efficient data streaming
  - Data processing needs to be close to data source
- Lack of unified framework
  - Some data analyses and predictions are useful for many applications
  - Need a standard way for applications to express interests in receiving such notifications, and for the network to send notifications to the device

Thanks!

Questions?