Attacking Mobile Traffic Analytics and Backhaul Utility Maximisation with Deep Learning

Paul Patras
Work with Chaoyun Zhang, Rui Li, John Thompson (U Edinburgh), Xi Ouyang (Huazhong UST), Pan Cao (U Hertfordshire)
The Goal of Mobile Traffic Analysis

Mobile traffic consumption continues to grow.

> 4-fold increase in the next 5 years
The Goal of Mobile Traffic Analysis

Mobile traffic consumption continues to grow

> 4-fold increase in the next 5 years

Mining exabytes of data may offer valuable insights
The Goal of Mobile Traffic Analysis

Mobile traffic consumption continues to grow.

Mobile traffic by application category

Unit: GB/month
Video | Social Networking | Audio | Web Browsing | Software Update | File Sharing | Other
All devices
Year: 2018 - 2024

Source: Ericsson (Feb 2019)
Forecasting future mobile traffic consumption
1. Precision traffic engineering
   - On demand allocation of resources
   - Building Intelligent 5G networks

2. Energy saving
   – Green cellular networks

3. Mobility analysis
   - Movement prediction
   - Transportation planning
The Goal of Mobile Traffic Analysis

1. Precision traffic engineering
   - On demand allocation of resources
   - Building Intelligent 5G networks

2. Energy saving
   – Green cellular networks

3. Mobility analysis
   - Movement prediction
   - Transportation planning

Importance of Precision Mobile Traffic Forecasting

Long-term mobile traffic forecasting is key!
Continuous measurements are expensive

1. Rely on dedicated infrastructure

• Packet Gateway (PGW) or Radio Network Controller (RNC) probes
• Data storage
Continuous measurements are expensive

1. Rely on dedicated infrastructure
   • Packet Gateway (PGW) or Radio Network Controller (RNC) probes
   • Data storage

2. Data post-processing overheads
   • Call detail record reports transfer
   • Spatial aggregation
Exponential Smoothing (ES) and Autoregressive Integrated Moving Average model (ARIMA):

1. Operate on individual time series, while ignoring spatial correlations.

2. Performance degenerates considerably over time.

3. Do not generalise well in different locations.

Traditional Approaches
The Goal of Mobile Traffic Analysis

Solving forecasting at city-scale

Mobile traffic measurements

(Different color corresponds to different traffic volume)
• Convolutional neural networks (ConvNets) work particularly well in handling spatial data.
The Potential of Deep Learning

- Convolutional neural networks (ConvNets) work particularly well in handling spatial data.

- Recurrent neural networks (e.g. LSTM) can capture temporal dependencies.
The Potential of Deep Learning

- Convolutional neural networks (ConvNets) work particularly well in handling spatial data.

- Recurrent neural networks (e.g. LSTM) can capture temporal dependencies.

- Advanced GPU computing enables to train NN architectures fast and deliver real-time inference.
STN: Spatio-Temporal Neural Network

Decoder

Fully-connected layers

ConvLSTM

Fusion

ConvLSTM

Fusion

3D-Conv. layer * 3

3D-Conv. layer * 3

Inputs

Outputs
STN: Spatio-Temporal Neural Network

Inputs

3D-Conv. layer * 3  3D-Conv. layer * 3

Fusion  Fusion

ConvLSTM  ConvLSTM

Encoder
Convolutional Long Short-Term Memory (ConvLSTM)

• Advanced version of LSTM.

• Replaces inner dense connections with convolution operations.

• Works remarkably well in modelling spatio-temporal data.
Proposed Solution: Spatio-Temporal Neural Network
3D Convolutional Neural Network (3D-ConvNet)
3D Convolutional Neural Network (3D-ConvNet)
STN: Spatio-Temporal Neural Network

Inputs → 3D-Conv. layer * 3 → Fusion → 3D-Conv. layer * 3 → Fusion → ConvLSTM → Encoder
STN: Spatio-Temporal Neural Network
Input: A local $r \times r \times (s+1)$ tensor ($s = 11$, 2 hours)
One-step predictions

**Input:** A local \( r \times r \times (s+1) \) tensor

**Output:** The central point in the traffic snapshot \( t+1 \)
Use the model to scan the entire input map and obtain the predicted traffic snapshot at t+1.
One-step predictions

- The traffic consumption at a certain location largely depends on that in its neighbouring cells.
Extending to Long-Term Forecasting

Feeding the model with previous forecasts: Error accumulates.
Extending to Long-Term Forecasting

Feeding the model with previous forecasts: Error accumulates.
Feeding the model with previous forecasts: Error accumulates.
Ouroboros Training Scheme (OTS)

OTS: Fine-tuning the model with earlier predictions.
Ouroboros Training Scheme (OTS)

OTS: Fine-tuning the model with earlier predictions.

1. Reuse predictions as input of the model.
Ouroboros Training Scheme (OTS)

OTS: Fine-tuning the model with earlier predictions.

1. Reuse predictions as input of the model.
2. **Train the model with predictions and ground truth.**
Ouroboros Training Scheme (OTS)

OTS: Fine-tuning the model with earlier predictions.

1. Reuse predictions as input of the model.
2. Train the model with predictions and ground truth.
OTS: Fine-tuning the model with earlier predictions.

1. Reuse predictions as input of the model.
2. Train the model with predictions and ground truth.

Ouroboros Training Scheme (OTS)
Ouroboros Training Scheme (OTS)

OTS: Fine-tuning the model with earlier predictions.

1. Reuse predictions as input of the model.
2. Train the model with predictions and ground truth.
Ouroboros Training Scheme (OTS)

OTS: Fine-tuning the model with earlier predictions.

1. Reuse predictions as input of the model.
2. Train the model with predictions and ground truth.
Problem: Uncertainty still grows with the number of prediction steps.
Embedding Historical Statistics

**Problem:** Uncertainty still grows with the number of prediction steps.

**Observation:** Individual traffic series are relatively close to their mean values calculated over historical data.
Problem: Uncertainty still grows with the number of prediction steps.

Figure: Data traffic over a week with empirical mean, mean ± standard deviation, and sampled traffic.
Problem: Uncertainty still grows with the number of prediction steps.

Observation: Individual traffic series are relatively close to their mean values calculated over historical data.

Solution: Mix model predictions with empirical mean by a decaying weight.

\[
\text{Forecast}(t) = (1 - w(t)) \cdot \text{STN} + w(t) \cdot \text{Mean}
\]

\[
w(t) = \frac{1}{1 + e^{-a \cdot t + b}}
\]
D-STN - Long-Term Forecasting

- OTS + mixing data with empirical mean allows feeding earlier predictions as input, while achieving precise forecasting:

- **Double STN (D-STN)**
Dataset

- Telecom Italia's Big Data Challenge
• Telecom Italia's Big Data Challenge

• Measurements of mobile traffic volume between 1 Nov 2013 and 1 Jan 2014 in the city of Milan and province of Trentino.
• Telecom Italia's Big Data Challenge

• Measurements of mobile traffic volume between 1 Nov 2013 and 1 Jan 2014 in the city of Milan and province of Trentino.

• Aggregated every 10 minutes.
Dataset

• Telecom Italia's Big Data Challenge

• Measurements of mobile traffic volume between 1 Nov 2013 and 1 Jan 2014 in the city of Milan and province of Trentino.

• Aggregated every 10 minutes.

• Partitioned in 100×100 grids for Milan and 117×98 grids for Trentino.
1. Traditional Forecasting Tools (Trained on both Milan and Trentino)

- Holt-Winters Exponential Smoothing (HW-ExpS)
- Autoregressive Integrated Moving Average Model (ARIMA)
Methods used for comparison

1. Traditional Forecasting Tools (Trained on both Milan and Tretino)
   • Holt-Winters Exponential Smoothing (HW-ExpS)
   • Autoregressive Integrated Moving Average Model (ARIMA)

2. Machine Learning Approaches (Trained on Milan ONLY)
   • Support Vector Machine (SVM)
   • Auto-Encoder + Long Short-Term Memory (AE+LSTM)
Methods used for comparison

1. Traditional Forecasting Tools (Trained on both Milan and Tretino)
   • Holt-Winters Exponential Smoothing (HW-ExpS)
   • Autoregressive Integrated Moving Average Model (ARIMA)

2. Machine Learning Approaches (Trained on Milan ONLY)
   • Support Vector Machine (SVM)
   • Auto-Encoder + Long Short-Term Memory (AE+LSTM)

3. Components of D-STN (Trained on Milan ONLY)
   • STN (D-STN without OTS),
   • ConvLSTM, 3D-ConvNet, MLP
Evaluation

Normalised Root Mean Square Error (NRMSE):
The prediction accuracy.

<table>
<thead>
<tr>
<th>Method</th>
<th>Milan (5 mins)</th>
<th>Milan (100 mins)</th>
<th>Milan (5h)</th>
<th>Milan (10h)</th>
</tr>
</thead>
<tbody>
<tr>
<td>STN</td>
<td>0.19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D-STN</td>
<td>0.19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HW-ExpS</td>
<td>0.33</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARIMA</td>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLP</td>
<td>0.23</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ConvLSTM</td>
<td>0.23</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3D-CNN</td>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVM</td>
<td>0.39</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AE+LSTM</td>
<td>0.24</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Evaluation

Normalised Root Mean Square Error (NRMSE):
The prediction accuracy.

<table>
<thead>
<tr>
<th>Method</th>
<th>5 mins</th>
<th>100 mins</th>
<th>5h</th>
<th>10h</th>
</tr>
</thead>
<tbody>
<tr>
<td>STN</td>
<td>0.19</td>
<td>0.29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D-STN</td>
<td>0.19</td>
<td>0.28</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HW-ExpS</td>
<td>0.33</td>
<td>0.51</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARIMA</td>
<td>0.2</td>
<td>0.39</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLP</td>
<td>0.23</td>
<td>0.38</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ConvLSTM</td>
<td>0.23</td>
<td>0.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3D-CNN</td>
<td>0.2</td>
<td>0.37</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVM</td>
<td>0.39</td>
<td>0.46</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AE+LSTM</td>
<td>0.24</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Milan
Evaluation

Normalised Root Mean Square Error (NRMSE):
The prediction accuracy.
Evaluation

Normalised Root Mean Square Error (NRMSE):
The prediction accuracy.
Evaluation

Normalised Root Mean Square Error (NRMSE):
The prediction accuracy.
Evaluation

Normalised Root Mean Square Error (NRMSE): The prediction accuracy.
(D-)STN can deliver reliable 10-hour forecasting, given 2 hours of observation.

Achieve up to 61% lower prediction error.
Example (10 hours)
Example (10 hours)
Example (10 hours)
Example (10 hours)
Generalisation

Trentino

Generalises well to a different deployment without retraining.
Spatio-Temporal Neural Network (STN) to perform mobile traffic forecasting

D-STN: OTS + mixing predictions with empirical mean

Reliable long-term forecasting, outperforming other methods; generalise well to different deployments.

Summary

Maximising the utility of virtualised backhauls
Increasingly diverse services

• 5G networks need to accommodate services with distinct performance requirements
  • Bandwidth: UHD video streaming and AR/VR
  • Delay: Autonomous vehicles and remote medical care

• Network slicing
  • Partitioning physical infrastructure into logically isolated networks

• Network densification
  • High speed wireless backhauling tangible (mm-wave, free space optics, etc.)
Small Cell Backhauling
• Resource Allocation: Rate $r_{i,j}$ for flow $f_{i,j}$
• To meet the **service requirements** and to maximise **resource utilisation**
Utility Functions for Different Applications

QoS (AR/VR)

Sigmoid: $U_{\text{sig}}(r) = \frac{1}{1 + e^{-\alpha_1 (r - \beta_1)}}$

Best-effort (IoT)

Logarithmic: $U_{\text{log}}(r) = \log(\alpha_2 r + \beta_2)$
Utility Functions for Different Applications

Delay sensitive (Tele-Operation)

Polynomial: \( U_{\text{ply}}(r) = \alpha_3 (r \beta_3) \)

Revenue (other)

Linear: \( U_{\text{lnr}}(r) = \alpha_4 r + \beta_4 \)
Combining some of these in a “simple” scenario...
General utility framework

Arbitrary combinations of all know types of utilities

Sigmoid: \( U_{\text{sig}}(r) = \frac{1}{1 + e^{-\alpha_1(r - \beta_1)}} \)

Polynomial: \( U_{\text{ply}}(r) = \alpha_2(r^{\beta_2}) \)

Logarithmic: \( U_{\text{log}}(r) = \log(\alpha_3r + \beta_3) \)

Linear: \( U_{\text{lnr}}(r) = \alpha_4r + \beta_4 \)

\[ \text{arg max } \sum U(r_{i,j}) \]
Utility Maximisation

- High-dimensional problem, highly non-convex
- Global search is time consuming
- Heuristic methods can solve but sub-optimal
Utility Maximisation

• High-dimensional problem, highly non-convex
• Global search is time consuming
• Heuristic methods can solve but sub-optimal

• Deep Learning Approach:
  Learn the correlation between flow demands and optimal allocations
Deep Learning approach to Maximising network Utility (DELMU)
• 4 multi-hop topologies
• Flows $i$ with all types of utilities over each path $j$
• A range of
  • Flow demands ($d_{i,j}$) and minimum service rates ($\delta_{i,j}$)
  • Optimal solutions to utility maximization problem obtained using Global Search (GS) algorithm*

• 10,000 (“ground truth”) data points in total
  • 4/5 used for training, 1/5 for testing

• NN training performed using GPU (minutes) and SGD algorithm; inference using CPU

• “Sanity check” routine to ensure flows do not violate capacity constraints after allocation
• NN training performed using GPU (minutes) and SGD algorithm; inference using CPU

• “Sanity check” routine to ensure flows do not violate capacity constraints after allocation

• Benchmarks
  • Global Search (optimal but slow)
  • Greedy – recursively increase flow rates so as to improve utility (should work fast)
Evaluation: topologies

Topology 1

1
2
3
4
5
6
7
8
9

Path 1: 2772 Mbps
Path 2: 6756 Mbps
Path 3: 6756 Mbps

Topology 2

0
1
2
3
4

Path 1: 1386 Mbps
Path 2: 693 Mbps
Path 3: 2079 Mbps

Topology 3

0
1
2
3
4

Path 1: 2772 Mbps
Path 2: 693 Mbps
Path 3: 5197 Mbps

Topology 4

0
1
2
3
4

Path 1: 3465 Mbps
Path 2: 866 Mbps
Path 3: 866 Mbps
Results: Total Utility Distribution

Topology 1

Topology 2

Topology 3

Topology 4
Results: Total Utility Distribution

- **Topology 1**
  - Greedy
  - Delmu
  - GS

- **Topology 2**
  - Greedy
  - Delmu
  - GS

- **Topology 3**
  - Greedy
  - Delmu
  - GS
  - 62% arrow from Greedy to GS

- **Topology 4**
  - Greedy
  - Delmu
  - GS
Results: Total Utility Distribution
Results: Utility per Traffic Type

Topologies 1-4 show the utility for different traffic types under varying algorithms and function types. The x-axis represents the algorithms (Greedy, Delmu, GS), and the y-axis represents the utility. Each bar represents a different function type (Linear, Sigmoid, Polynomial, Logarithmic). The utility values are labeled at the top of each bar.
Results: Utility per Traffic Type
Results: Utility per Traffic Type

- Topology 1
  - Greedy: 0.97
  - Delmu: 1.04
  - GS: 1.99

- Topology 2
  - Greedy: 0.27
  - Delmu: 1.18
  - GS: 1.66

- Topology 3
  - Greedy: 1.12
  - Delmu: 1.96
  - GS: 1.54

- Topology 4
  - Greedy: 0.53
  - Delmu: 1.45
  - GS: 1.97

Utility represented for Linear, Sigmoid, Polynomial, and Logarithmic models.
Results: Utility per Traffic Type
Computation time

- Dell workstation
- Average computation time over 2k instances
Computation time

- Dell workstation
- Average computation time over 2k instances

![Runtime Gain Chart]

- Topology 1: GS/Delmu = 2343, Greedy/Delmu = 42
- Topology 2: GS/Delmu = 1316, Greedy/Delmu = 45
- Topology 3: GS/Delmu = 1380, Greedy/Delmu = 47
- Topology 4: GS/Delmu = 1858, Greedy/Delmu = 52
Dell workstation
Average computation time over 2k instances

Inference up to ~2300x faster than GS runtime
At least 42x faster than Greedy
A general utility framework that encompasses all known types of utility functions

Delmu achieves close-to-optimal utility solutions, and makes rapid inferences

Suitable for 5G backhauls with real-time and dynamic requirements