netMosaic
Harnessing Public Code Repositories to Develop Production-Ready ML Artifacts for Networking

Punnal Ismail Khan
UC Santa Barbara
ML Model(s) for Networks

• Efforts in Past Decades
  1000+ research publications, multiple products/startups, billions of dollars invested
ML Model(s) for Networks

• **Efforts in Past Decades**
  1000+ research publications, multiple products/startups, billions of dollars invested

• **Expectations**
  – Easy to develop ML models for any given problem and target environment
  – Abundance of production-ready ML models---ready for high-stake decision-making
ML Model(s) for Networks

• **Efforts in Past Decades**
  1000+ research publications, multiple products/startups, billions of dollars invested

• **Expectations**
  – Easy to develop ML models for any given problem and target environment
  – Abundance of production-ready ML models---ready for high-stake decision-making

• **Reality**
  – Availability of **public datasets** dictates choice of learning problem and environment
  – Abundance of ML artifacts with high performance in **controlled “lab” settings**
Can we Deploy Existing ML Models in Production?

<table>
<thead>
<tr>
<th>Problem</th>
<th>Dataset(s)</th>
<th>Model(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detect VPN traffic</td>
<td>Public VPN dataset [20]</td>
<td>1-D CNN [61]</td>
</tr>
<tr>
<td>Detect Heartbleed traffic</td>
<td>CIC-IDS-2017 [54]</td>
<td>RF Classifier [54]</td>
</tr>
<tr>
<td>Detect Malicious traffic (IDS)</td>
<td>CIC-IDS-2017 [54], Campus dataset</td>
<td>nPrintML [32]</td>
</tr>
<tr>
<td>IoT Device Fingerprinting</td>
<td>UNSW-IoT [56]</td>
<td>Isy [63]</td>
</tr>
<tr>
<td>Adaptive Bit-rate</td>
<td>HSDPA Norway [49]</td>
<td>Pensieve [42]</td>
</tr>
</tbody>
</table>
Can we Deploy Existing ML Models in Production?

<table>
<thead>
<tr>
<th>Problem</th>
<th>Dataset(s)</th>
<th>Model(s)</th>
<th>Model Generalizability Issues</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detect Malicious traffic (IDS)</td>
<td>CIC-IDS-2017 [54], Campus dataset</td>
<td>nPrintML [32]</td>
<td>Spurious correlations</td>
</tr>
<tr>
<td>IoT Device Fingerprinting</td>
<td>UNSW-IoT [56]</td>
<td>Iisy [63]</td>
<td>Likely shortcut learning</td>
</tr>
</tbody>
</table>

Most existing ML models fail to generalize; not ready for production deployments
How to Develop Generalizable ML Models for Networks?

Preprocessing + Model selection → Training → Evaluation

Standard ML Pipeline
How to Develop Generalizable ML Models for Networks?

Is this the right data?

Standard ML Pipeline
How to Develop Generalizable ML Models for Networks?

Is this the right data?

CIC-IDS → Preprocessing + Model selection → Training → Evaluation

Is this model underspecified?

F1-score: 0.99

Standard ML Pipeline
How to Develop Generalizable ML Models for Networks?

- How to collect better data at scale?

CIC-IDS → Preprocessing + Model selection → Training and Evaluation

Learning shortcut

- Is this the right data?

Is this model underspecified?

F1-score: 0.99

Standard ML Pipeline
How to Develop Generalizable ML Models for Networks?

How to *collect better data* at scale?

Is this the *right* data?

CIC-IDS → Preprocessing + Model selection → Training → Evaluation

Learning shortcut

Is this model *underspecified*?

F1-score: 0.99

Standard ML Pipeline

Answering these questions is critical for developing generalizable ML artifacts for networking
Progress in Past Years
Progress in Past Years

- **netUnicorn** [CCS’23]
  Iteratively collect data for any problem and environment
- **Trustee** [CCS’22]
  Explain and analyze ML model’s decision making
- **PINOT** [ANRW’23]
  Transform your production network for data collection
Progress in Past Years

- **netUnicorn [CCS’23]**
  Iteratively collect data for any problem and environment

- **Trustee [CCS’22]**
  Explain and analyze ML model’s decision making

- **PINOT [ANRW’23]**
  Transform your production network for data collection
Progress in Past Years

- **netUnicorn** [CCS’23]
  - Iteratively collect data for any problem and environment
- **Trustee** [CCS’22]
  - Explain and analyze ML model’s decision making
- **PINOT** [ANRW’23]
  - Transform your production network for data collection
netUnicorn: A Flexible Data Collection Platform

Fragmented

Learning Problems

Data Collection

Network environments

Physical/virtual network infrastructures
netUnicorn: A Flexible Data Collection Platform

Fragmented

Learning Problems
Data Collection
Network environments
Physical/virtual network infrastructures

Thin Waist

netUnicorn
netUnicorn: A Flexible Data Collection Platform

Simplifies collecting data for any learning problem and target network environment
Limitation of netUnicorn

- Application Fingerprinting
- Flow completion time prediction

- Learning Problems
- Application Logic
- Network environments
- Physical/virtual Network infrastructures
Limitation of netUnicorn

Explanation:

- Writing application logic is **manual effort**
  - Collecting data for new application is hard
  - Easily breaks over time

Network environments

Physical/virtual Network infrastructures

Application Fingerprinting
Flow completion time prediction
Limitation of netUnicorn

Writing application logic is manual effort
- Collecting data for new application is hard
- Easily breaks over time

How do we scale data collection for new applications?
Opportunity: Publicly Accessible Code Repositories

• Millions of publicly accessible code repositories capture diverse application logic
  • GitHub, Bitbucket, etc.

• Prior work showed around 70k GitHub repositories with containerized applications that can generate diverse network traffic.

• We refer to these repositories as Big Code
Opportunity: Publicly Accessible Code Repositories

- Millions of publicly accessible code repositories capture diverse application logic
  - GitHub, Bitbucket, etc.

- Prior work showed around 70k GitHub repositories with containerized applications that can generate diverse network traffic.

Can we use Big Code to address netUnicorn’s limitation?
Proposed Solution

netUnicorn

Learning Problems

Network environments

Physical/virtual Network infrastructures
Proposed Solution

netMosaic

Big Code

netUnicorn

Application Logic

Learning Problems

Network environments

Physical/virtual Network infrastructures
Proposed Solution

Subsumes netUnicorn to leverage Big Code’s diverse application logic
Does it enable curating “better” datasets?

- **Learning problem**
  Traffic Classification: identify traffic classes based on encrypted packets in a flow

- **Data Source**
  - 16k GitHub repositories
  - Labeled data using port numbers

- **Curated Dataset**
  - 1.7 million flows, 54 million packets, 264 unique services
  - Top six services: HTTPS, Redis, PostgreSQL, Eforward, MongoDB, MySQL.
Does it enable curating “better” datasets?

- **Learning problem**
  Traffic Classification: identify traffic classes based on encrypted packets in a flow

- **Data Source**
  - 16k GitHub repositories
  - Labeled data using port numbers

- **Curated Dataset**
  - 1.7 million flows, 54 million packets, 264 unique services
  - Top six services: HTTPS, Redis, PostgreSQL, Eforward, MongoDB, MySQL.

<table>
<thead>
<tr>
<th></th>
<th>netMosaic</th>
<th>CrossMarkets</th>
<th>ISCXVPN2016</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Flows</td>
<td>1.7 Million</td>
<td>46,179</td>
<td>9,536</td>
</tr>
</tbody>
</table>
Does it enable curating “better” datasets?

- **Learning problem**
  Traffic Classification: identify traffic classes based on encrypted packets in a flow

- **Data Source**
  - 16k GitHub repositories
  - Labeled data using port numbers

- **Curated Dataset**
  - 1.7 million flows, 54 million packets, 264 unique services
  - Top six services: HTTPS, Redis, PostgreSQL, Eforward, MongoDB, MySQL.

*netMosaic is able to curate “better” datasets, i.e., more diverse and less sparse*
Does it enable developing “generalizable” model?

• **Data Source**
  • 256 GitHub repositories

• **Datasets**
  • **Source Datasets:** Labeled datasets used for model training
    Dataset A: Default setting → Model A
    Dataset B: Low congestion setting → Model B
  • **Target Dataset:** Unlabeled dataset used for assessing generalizability
    Dataset C: High-congestion setting

• **Learning Models**
  • Random Forest, Decision Trees, Logistic Regression, MLP
# Results

Performance of models trained on Dataset A (Model A) and Dataset B (Model B) and tested on unseen Dataset C.

<table>
<thead>
<tr>
<th>Model Type</th>
<th>Model A Source Dataset</th>
<th>Model A Target Dataset</th>
<th>Model B Source Dataset</th>
<th>Model B Target Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Decision Trees</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Logistic Regression</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Results

Performance of models trained on Dataset A (Model A) and Dataset B (Model B) and tested on unseen Dataset C.

<table>
<thead>
<tr>
<th>Model</th>
<th>Source Dataset</th>
<th>Target Dataset</th>
<th>Source Dataset</th>
<th>Target Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest</td>
<td>0.83</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Decision Trees</td>
<td>0.81</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>0.23</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLP</td>
<td>0.76</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Results

Performance of models trained on Dataset A (Model A) and Dataset B (Model B) and tested on unseen Dataset C.

<table>
<thead>
<tr>
<th></th>
<th>Model A</th>
<th>Model B</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Source Dataset</td>
<td>Target Dataset</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.83</td>
<td></td>
</tr>
<tr>
<td>Decision Trees</td>
<td>0.81</td>
<td></td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>0.23</td>
<td></td>
</tr>
<tr>
<td>MLP</td>
<td>0.76</td>
<td></td>
</tr>
</tbody>
</table>
## Results

Performance of models trained on Dataset A (Model A) and Dataset B (Model B) and tested on unseen Dataset C.

<table>
<thead>
<tr>
<th></th>
<th>Source Dataset</th>
<th>Target Dataset</th>
<th>Source Dataset</th>
<th>Target Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest</td>
<td>0.83</td>
<td><strong>0.24</strong></td>
<td>0.81</td>
<td></td>
</tr>
<tr>
<td>Decision Trees</td>
<td>0.81</td>
<td>0.10</td>
<td>0.80</td>
<td></td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>0.23</td>
<td>0.06</td>
<td>0.15</td>
<td></td>
</tr>
<tr>
<td>MLP</td>
<td>0.76</td>
<td><strong>0.07</strong></td>
<td>0.73</td>
<td></td>
</tr>
</tbody>
</table>
## Results

Performance of models trained on Dataset A (Model A) and Dataset B (Model B) and tested on unseen Dataset C.

<table>
<thead>
<tr>
<th>Model</th>
<th>Source Dataset</th>
<th>Target Dataset</th>
<th>Source Dataset</th>
<th>Target Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest</td>
<td>0.83</td>
<td><strong>0.24</strong></td>
<td>0.81</td>
<td><strong>0.52</strong></td>
</tr>
<tr>
<td>Decision Trees</td>
<td>0.81</td>
<td>0.10</td>
<td>0.80</td>
<td>0.28</td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>0.23</td>
<td>0.06</td>
<td>0.15</td>
<td>0.14</td>
</tr>
<tr>
<td>MLP</td>
<td>0.76</td>
<td><strong>0.07</strong></td>
<td>0.73</td>
<td>0.37</td>
</tr>
</tbody>
</table>
## Results

Performance of models trained on Dataset A (Model A) and Dataset B (Model B) and tested on unseen Dataset C.

<table>
<thead>
<tr>
<th>Model</th>
<th>Source Dataset</th>
<th>Target Dataset</th>
<th>Source Dataset</th>
<th>Target Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest</td>
<td>0.83</td>
<td><strong>0.24</strong></td>
<td>0.81</td>
<td><strong>0.52</strong></td>
</tr>
<tr>
<td>Decision Trees</td>
<td>0.81</td>
<td>0.10</td>
<td>0.80</td>
<td>0.28</td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>0.23</td>
<td>0.06</td>
<td>0.15</td>
<td>0.14</td>
</tr>
<tr>
<td>MLP</td>
<td>0.76</td>
<td><strong>0.07</strong></td>
<td>0.73</td>
<td>0.37</td>
</tr>
</tbody>
</table>

Using training data collected under more realistic network conditions could **improve model generalizability**.
Summary and Outlook

• Lessons learned
  • Our system simplifies collecting data for disparate applications under different network conditions leveraging Big Code and netUnicorn
  • Prototype implementation demonstrates ability to curate better datasets and generalizable ML models
Summary and Outlook

• **Lessons learned**
  - Our system simplifies collecting data for disparate applications under different network conditions leveraging *Big Code* and *netUnicorn*
  - Prototype implementation demonstrates ability to curate better datasets and generalizable ML models

• **What’s next?**
  - Leverage model explainability tools (e.g., Trustee)
  - Scale data collection for more repositories
  - Improve data quality: address class imbalance issues, filter noisy samples