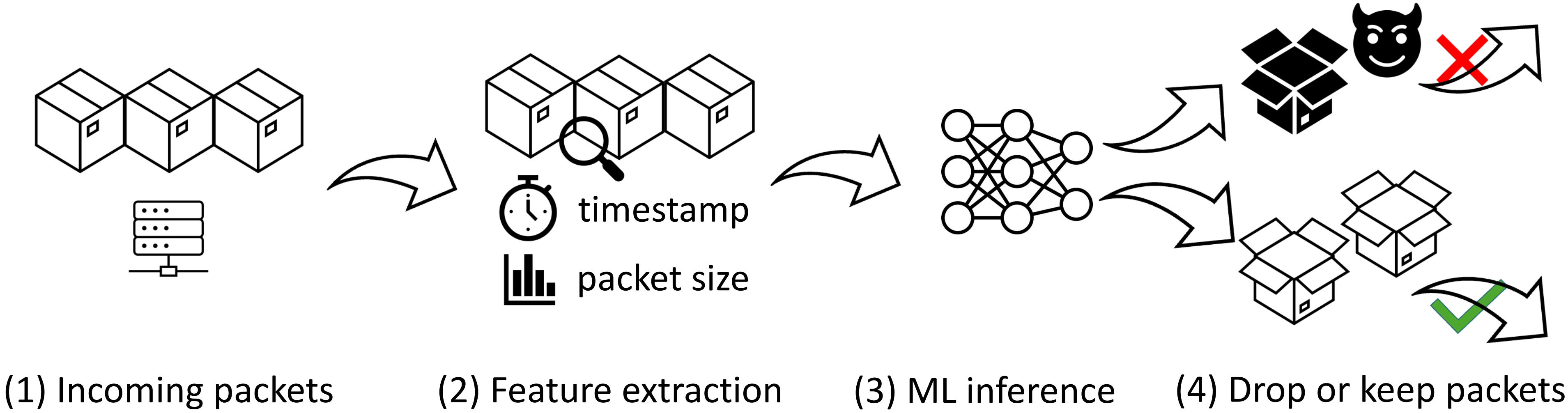
Caravan: Practical Online Learning of In-Network ML Models with Labeling Agents

Qizheng Zhang[1], Ali Imran[2], Enkeleda Bardhi[3], Tushar Swamy[1], Nathan Zhang[1], Muhammad Shahbaz[2,4], Kunle Olukotun[1]

1 Stanford University 2 PURDUE UNIVERSITY 3 SAPIENZA UNIVERSITÀ DI ROMA 4 UNIVERSITY OF MICHIGAN

Machine learning (ML) in online traffic analysis

- Motivating use case: Intrusion detection in a network



Why ML-based online traffic analysis?

- Diverse use cases
  - Enhancing infrastructure security
  - Improving application performance

- Growing incentive for adoption
  - Complexity of network traffic patterns
  - Encrypted network protocols

Machine Learning for Encrypted Malware Traffic Classification: Accounting for Noisy Labels and Non-Stationarity

Blake Anderson
Cisco Systems, Inc.
blake.anderson@cisco.com

CLOUDFLARE The Cloudflare Blog

Defensive AI: Cloudflare's framework for defending against next-gen threats

03/04/2024

Security Week | AI | Machine Learning | Phishing | Cloud Email Security | API Security | SASE

From identifying phishing attempts to protect applications and APIs, Cloudflare uses AI to improve the effectiveness of its security solutions to fight against new and more sophisticated attacks...

Estimating WebRTC Video QoE Metrics Without Using Application Headers

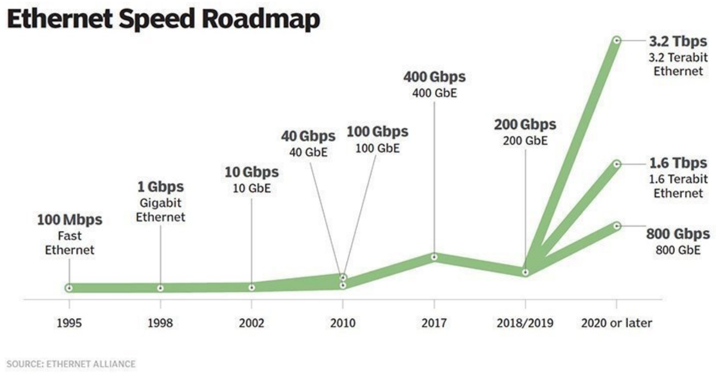
Taveesh Sharma, taveesh@uchicago.edu, University of Chicago, USA
Tarun Mangla, tmangla@iitd.ac.in, IIT Delhi, India
Arpit Gupta, arpitgupta@ucsb.edu, UCSB, USA
Junchen Jiang, junchenj@uchicago.edu, University of Chicago, USA
Nick Feamster, feamster@uchicago.edu, University of Chicago, USA

Challenge #1: Networks are getting faster

- More data in the network
  - Ethernet line-rate: 10 Gbps (2002) to 800 Gbps (2024)

- Lower response latency in the network
  - Datacenter RTT: 100μs (2008) to 5μs (2023)

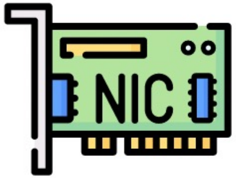
- Strict latency & throughput requirements
  - A need for small-batch or per-packet inference



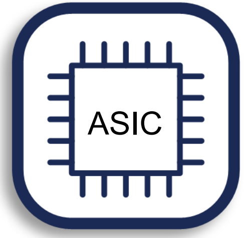
Small and specialized in-network models (fast)

- **In-network ML** in data plane devices for real-time, per-packet inference

Programmable switches
E.g. Leo [NSDI '24]



SmartNICs
E.g. N3IC [NSDI '22]

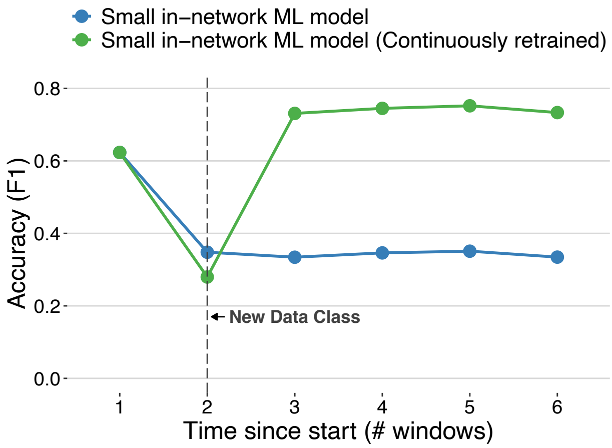


Hardware ASICs
E.g. Taurus [ASPLOS '22]

Why? Reduced *data movement* and *response latency*

Challenge #2: Networks are getting more complex

- Are specialized in-network ML models alone good enough? **No!**
- More complex traffic patterns
  - High-dimensional (thousands of features)
  - Long-context (millions of packets in a flow)
- More diverse deployment environments
  - Training & deployment environment can differ
  - Train-once-and-deploy for small models is insufficient



Large and versatile foundation models (accurate)

- Domain-specific **foundation models** for networking, security, etc.

Large and versatile foundation models (accurate)

- Domain-specific **foundation models** for networking, security, etc.

NetLLM: Adapting Large Language Models for Networking

Duo Wu[1], Xianda Wang[1], Yaqi Qiao[1], Zhi Wang[2], Junchen Jiang[3], Shuguang Cui[1], Fangxin Wang[1*]
[1]The Chinese University of Hong Kong, Shenzhen, [2]Tsinghua University, [3]The University of Chicago

NetLLM [SIGCOMM '24]

Large and versatile foundation models (accurate)

- Domain-specific **foundation models** for networking, security, etc.

netFound: Foundation Model for Network Security

Satyandra Guthula
UC Santa Barbara
satyandra@ucsb.edu

Navya Battula
UC Santa Barbara
navyabio12@gmail.com

Roman Beltiukov
UC Santa Barbara
rbeltiukov@ucsb.edu

Wenbo Guo
Purdue University
henrygwb@purdue.edu

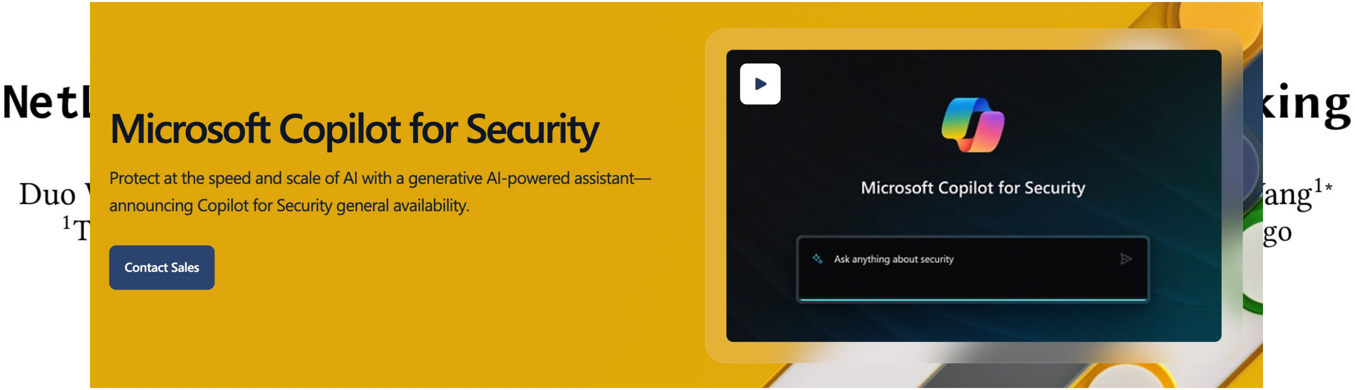
Arpit Gupta
UC Santa Barbara
arpitgupta@ucsb.edu

NetLL... ...etworking

Duo Wu[1] ... Fangxin Wang[1*]
[1]The C... ...ity of Chicago

Large and versatile foundation models (accurate)

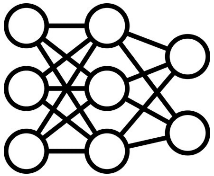
- Domain-specific **foundation models** for networking, security, etc.



Why? Better *in-depth analysis* and *generalization*

Two approaches: Small and large models

#1: **Small** and specialized **in-network** models (fast)

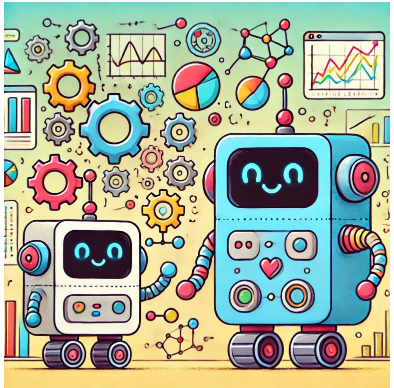


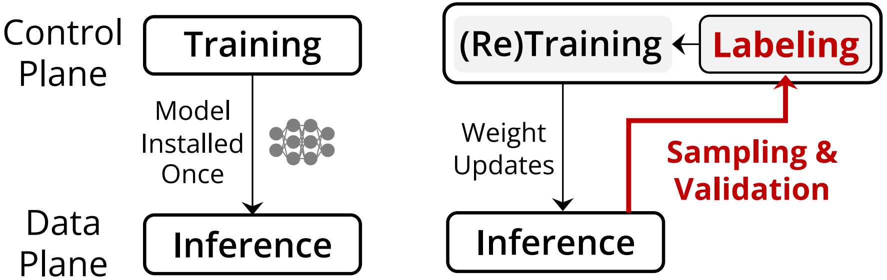
#2: **Large** and versatile **foundation** models (accurate)

Question: Can we be both, *fast and accurate*?

Our proposal: *Online learning* to the rescue

- Large and small models should work *jointly* online



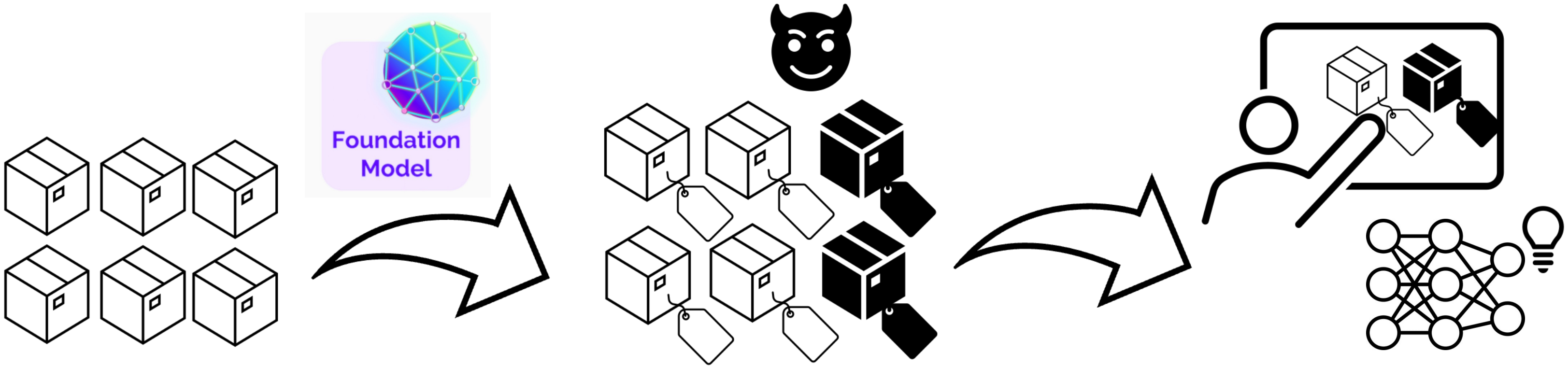


(a) Offline Learning **(b) Online Learning**

Large models can guide small models via online learning to achieve both *speed and accuracy*

Insight #1: Large models are good sources of *labeling*

- Large models can be used to generate **labeled** online data for training small models (online learning).



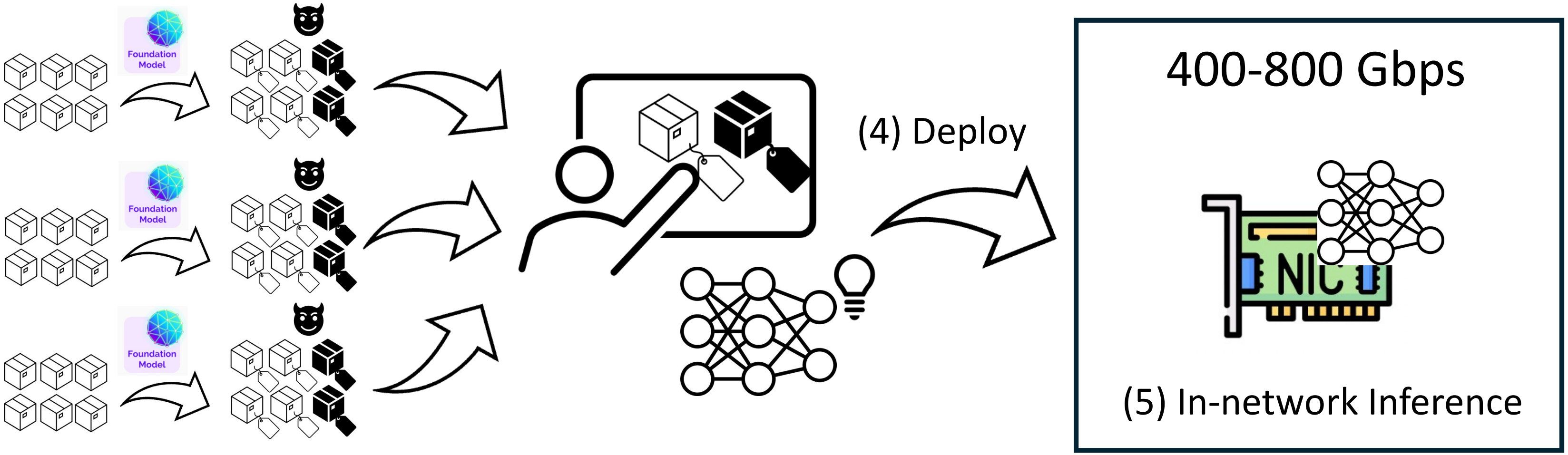
(1) Incoming packets

(2) Labeled packets

(3) Training data for small models

Insight #1: Large models are good sources of *labeling*

- Data labeling & online learning **do not** need to happen in real-time.
  - Further acceleration through large-batch inference, parallelization, etc.



Large models can be good sources of *labeling* in online scenarios

Example: Adapting GPT-4 as a labeling source

- We adapted GPT-4 for data labeling in the intrusion detection use case.

Instruction following prompts

In-context learning examples

Labeling request prompts

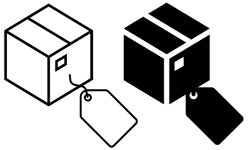
API

Labeling response

Parsed labels

Data Labeling Prompt (P2): Please give me a label for each of these unlabeled flows. No explanation or analysis needed, label only; one flow on each line. Format for each line: (flow number) label. `[Flows and their features go here]`.

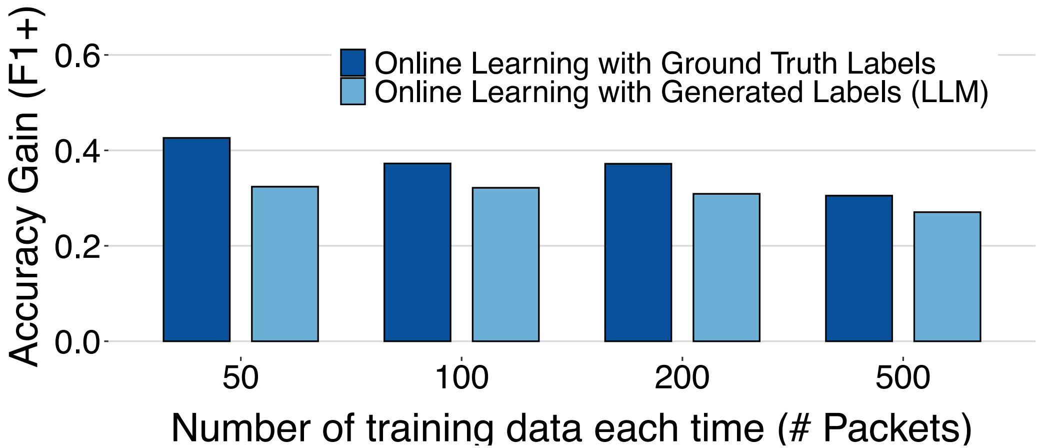
```python
def label_flow_with_rule_cache(flow):
    """
        Labels a network flow as benign (0) or malicious (1) based on certain heuristics.
        This function is designed to be conservative and only labels a flow as malicious when it is highly confident.

        :param flow: A tuple containing features of the flow.
        :return: An integer label, 0 for benign and 1 for malicious.
    """

    # Unpack the flow features
    dur, proto, sbytes, dbytes, sttl, dttl, sload, dload, spkts, dpkts, smean, dmean, sinpkt, dinpkt, tcprtt, synack,
    ackdat, ct_src_ltm, ct_dst_ltm, ct_dst_src_ltm = flow

    # Define thresholds and conditions for a flow to be considered malicious
    # These thresholds are derived from cybersecurity expertise and unlabeled data provided
    conditions = [
```



Off-the-shelf foundation models can be adapted to be labeling sources

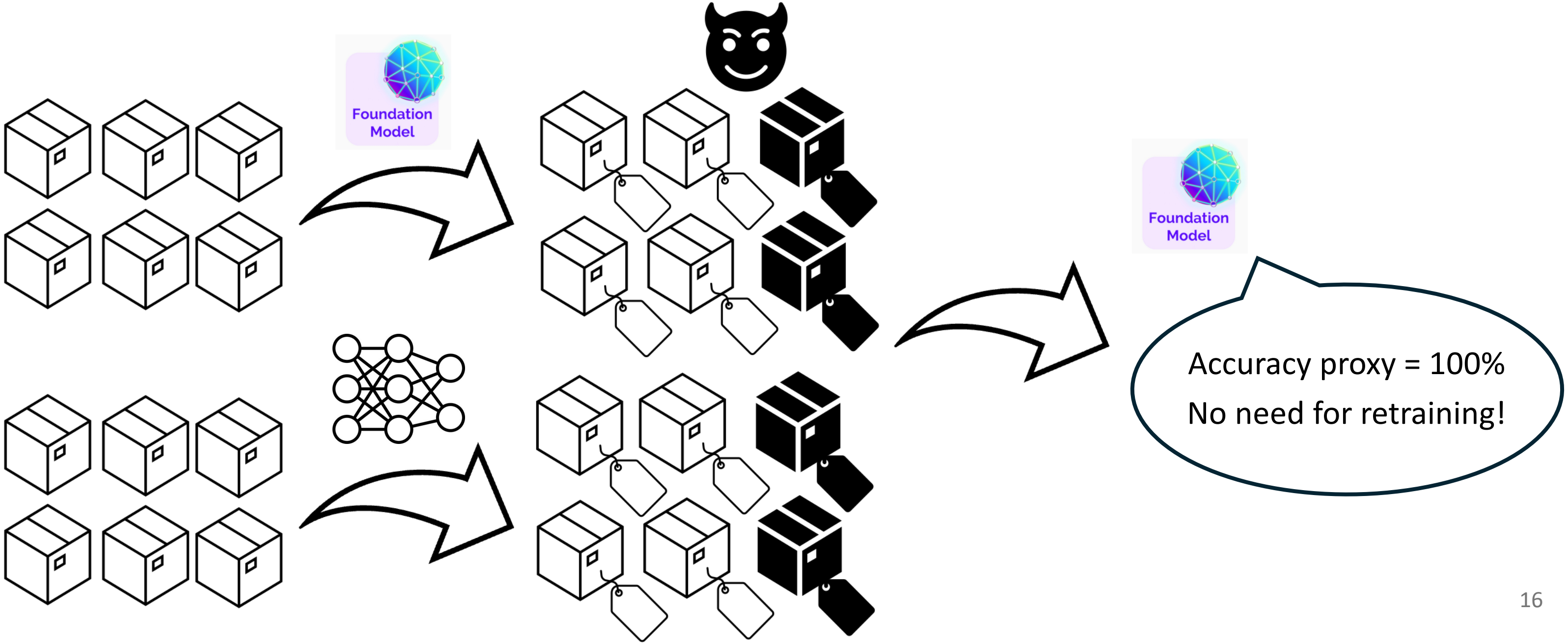
Generated labels from GPT-4 for online learning

- We use generated labels from GPT-4, as well as ground truth labels, for online learning.
- Result: The accuracy gains from online learning are similar.

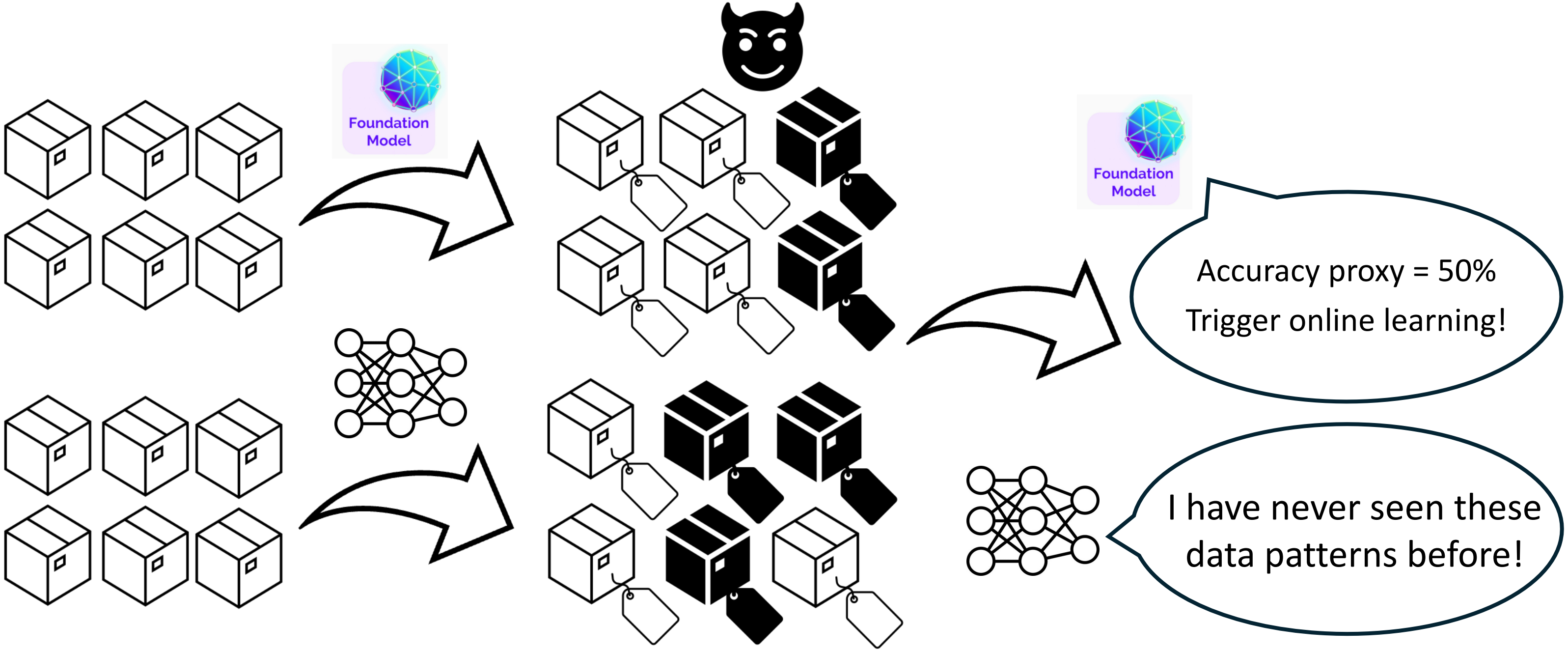


Insight #2: Online learning can be *triggered* sparsely

- Generated labels from large models can be used to **approximate** the online accuracy of small models (which we call accuracy proxy).



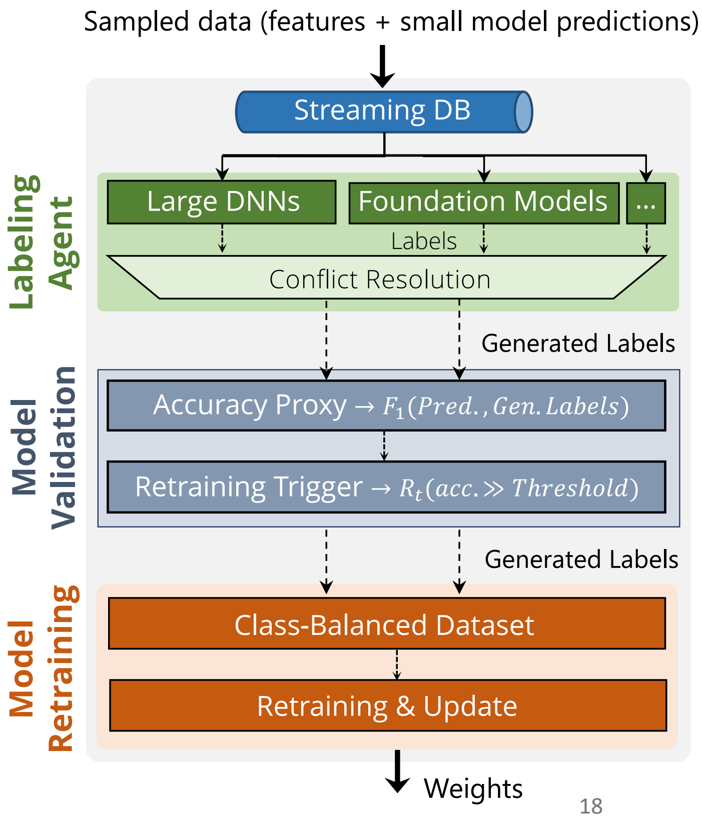
Insight #2: Online learning can be *triggered* sparsely



Sparse online learning via *accuracy proxy* avoids excessive retraining

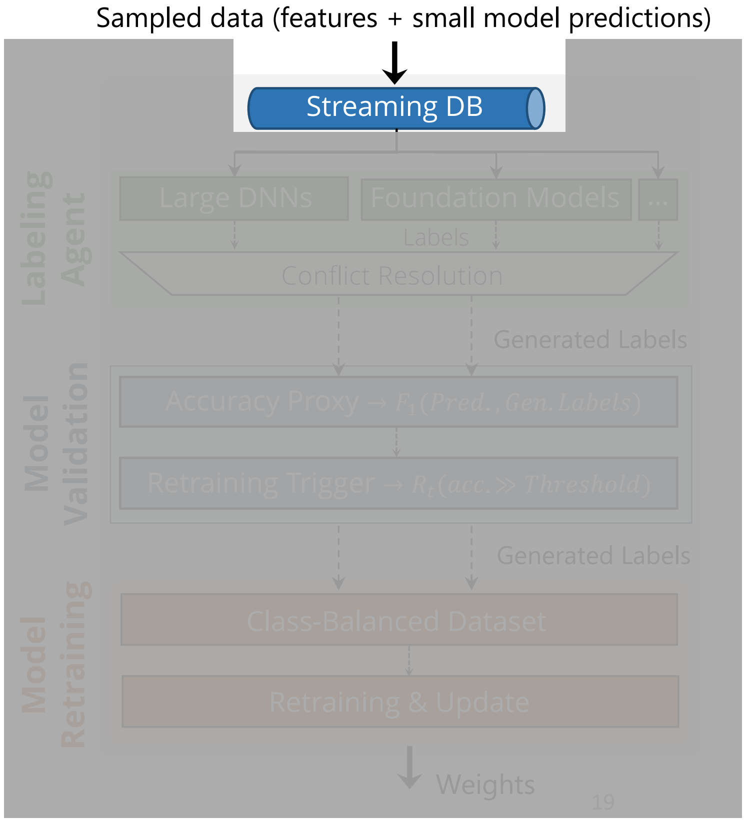
Putting them together (Caravan)

- Caravan: A system for practical online learning of in-network ML models



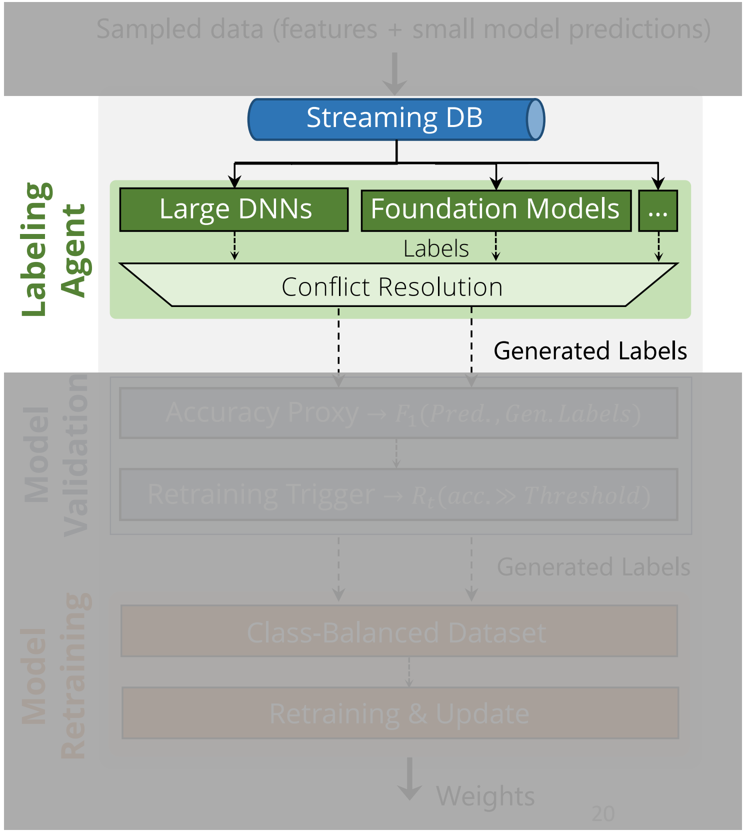
Putting them together (Caravan)

- Online data is collected and sampled.
- Samples are stored in a streaming DB.



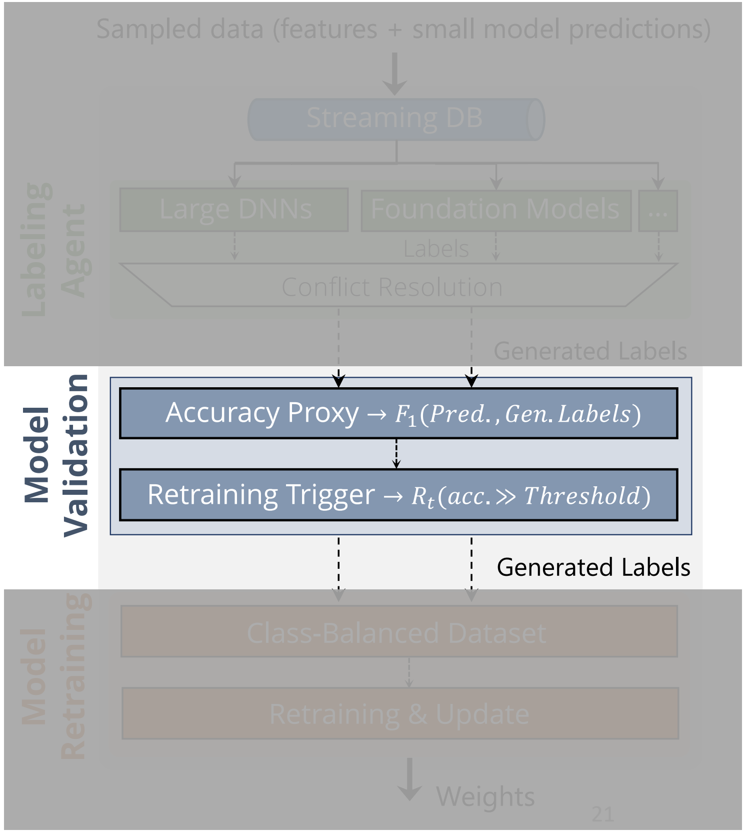
Putting them together (Caravan)

- Labeling agent
  - Retrieves batched data from streaming DB
  - Generates labels for these data via user-defined large models



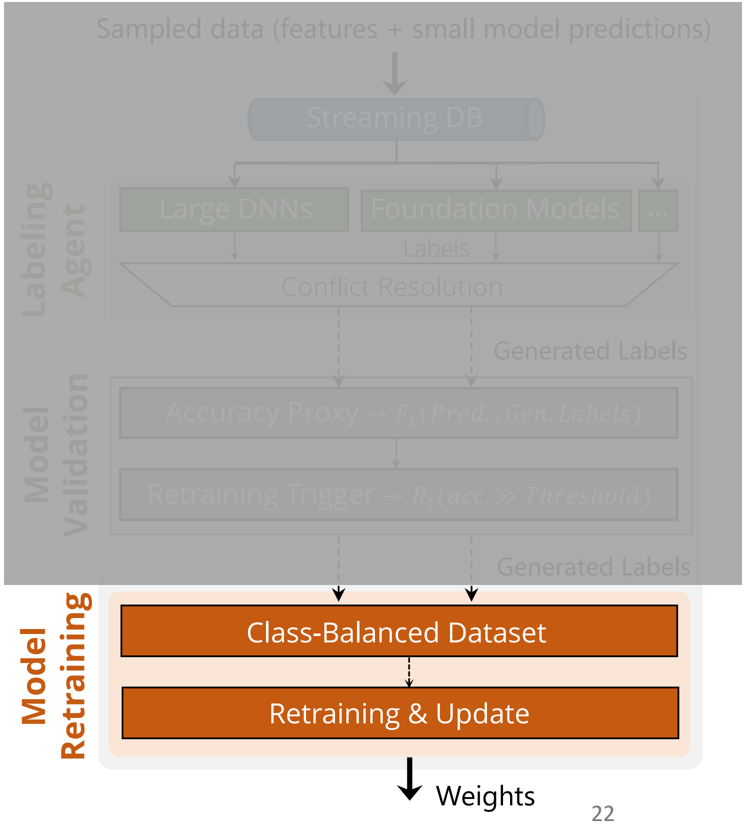
Putting them together (Caravan)

- Model validation
  - Computes accuracy proxy
  - Decides if online learning is necessary



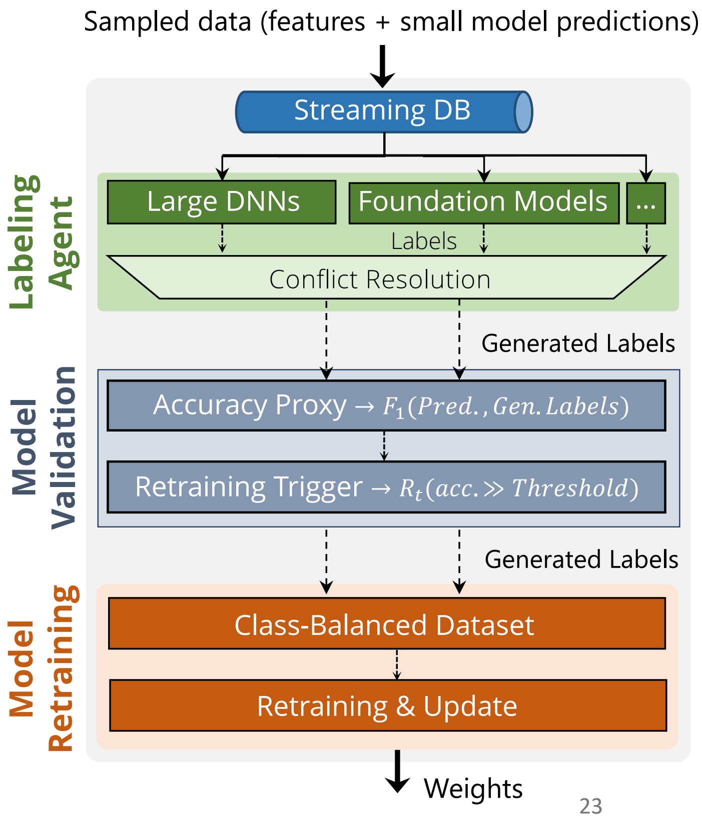
Putting them together (Caravan)

- Model retraining
  - Forms a retraining dataset
  - Retrains the model
  - Sends updated weights to the small model



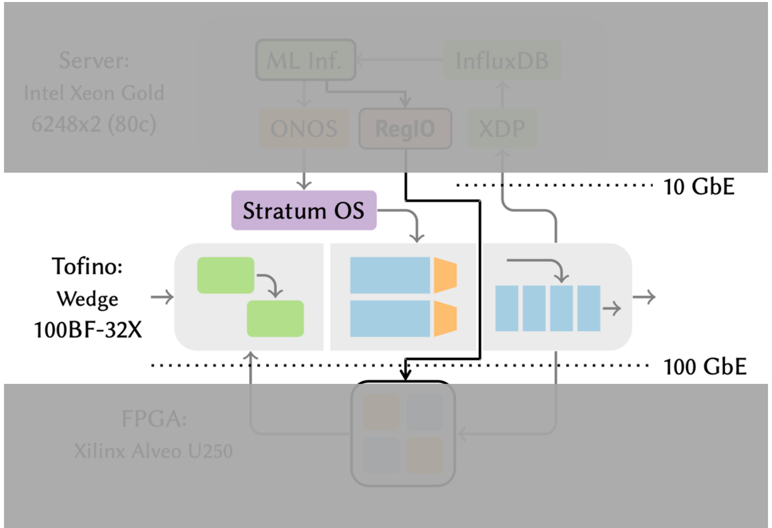
Putting them together (Caravan)

- In summary, three collaborating pieces!



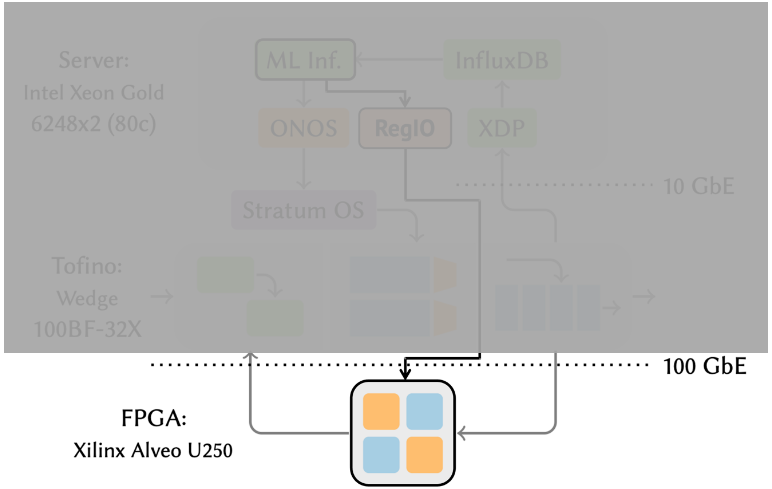
Implementation

- Prototype with three major pieces
  - A Tofino switch for packet parsing/deparsing
  - An FPGA for running in-network ML model
  - A server for the Caravan software



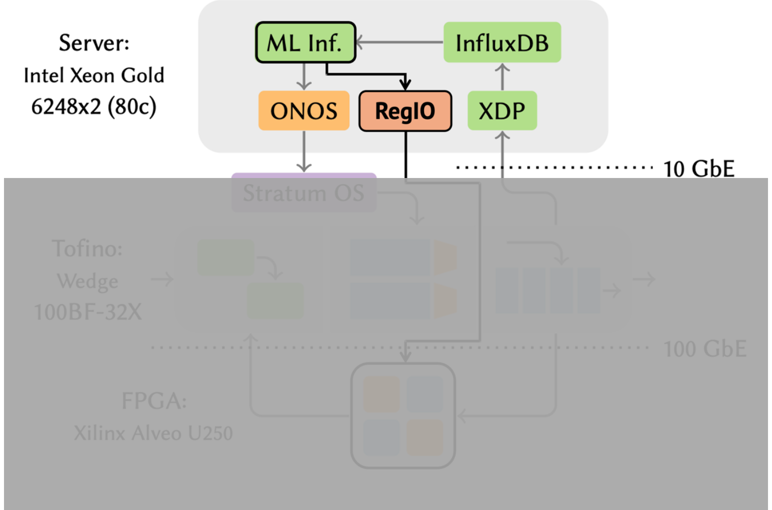
Implementation

- Prototype with three major pieces
  - A Tofino switch for packet parsing/deparsing
  - An FPGA for running in-network ML model
  - A server for the Caravan software



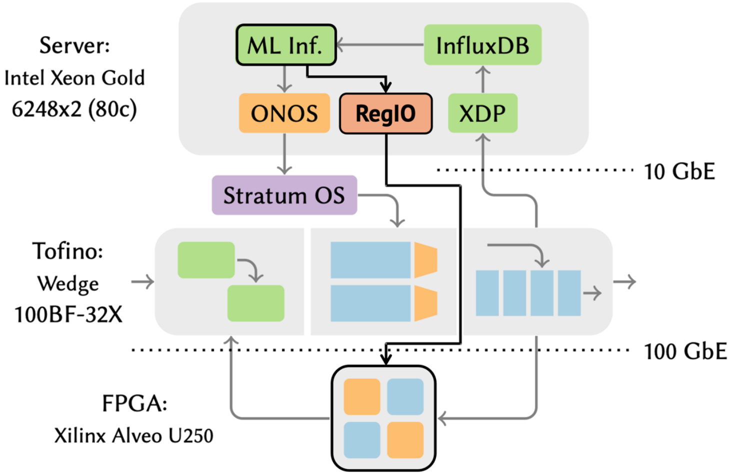
Implementation

- Prototype with three major pieces
  - A Tofino switch for packet parsing/deparsing
  - An FPGA for running in-network ML model
  - A server for the Caravan software



Implementation

- Prototype with three major pieces
  - A Tofino switch for packet parsing/deparsing
  - An FPGA for running in-network ML model
  - A server for the Caravan software



Evaluation setup

- 2 applications and 3 datasets
  - Intrusion detection (security)
  - IoT traffic classification (performance)
- 2 evaluation metrics
  - ML model accuracy: F1 score
  - System cost of online learning: CPU/GPU and memory usage

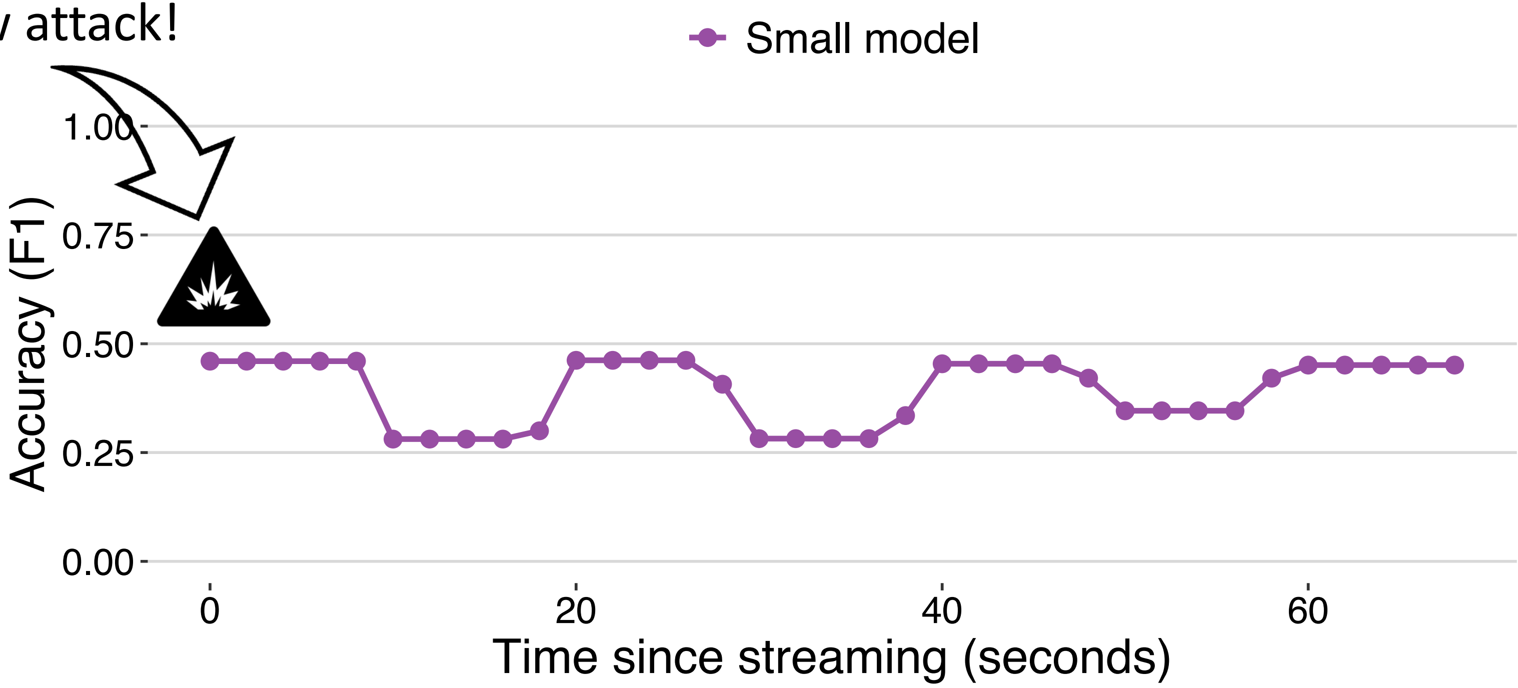
Example: End-to end intrusion detection

- A dataset with 35 million packets
- 7 different types of attacks
- A 7-layer DNN that runs at line-rate in FPGA
- Classify each packet as malicious or benign

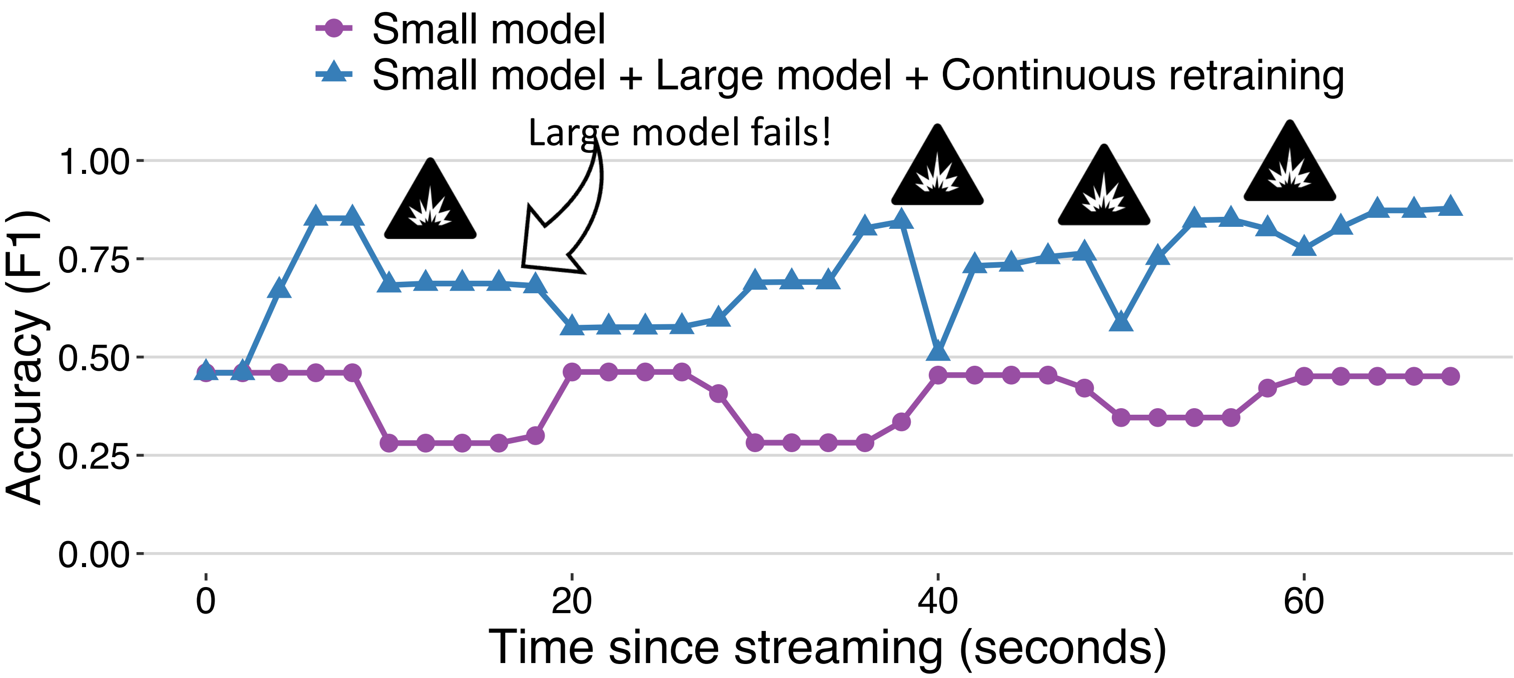
- Packet rate: 0.5 million packets/sec
- Run inference + compute accuracy on **every** packet
- Sample rate for the control plane: 0.1%

We start with the small in-network model

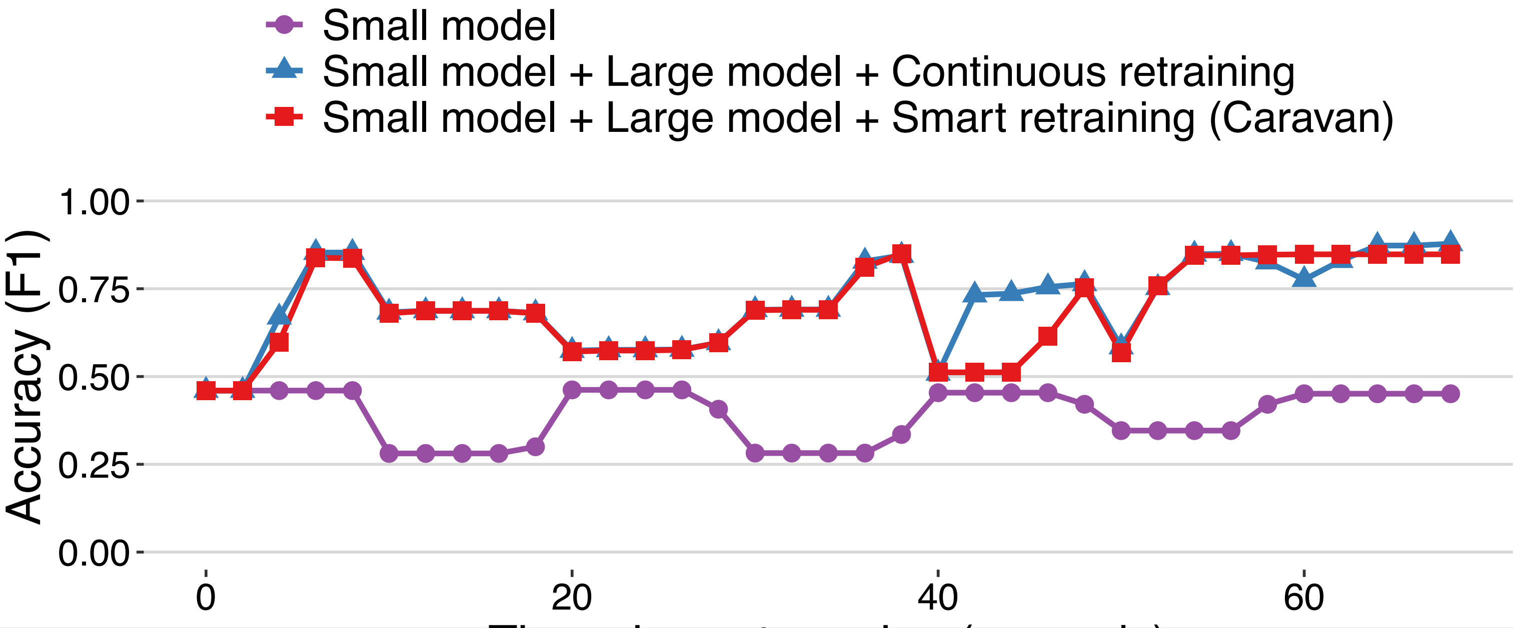
New attack!



What if the large model guides the small model (via online learning)?

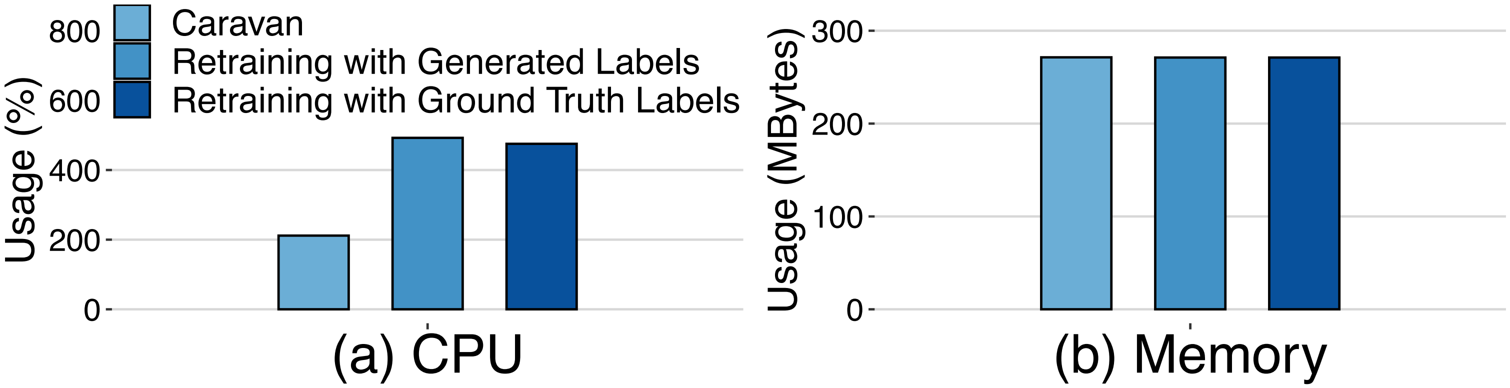


What if we introduce selective retraining via accuracy proxy (Caravan)?



Caravan keeps in-network ML models up-to-date with changing traffic dynamics

Caravan saves backend computation from excessive retraining



Caravan reduces backend CPU usage by an average of 56.23%

Scope and limitations

When to use Caravan

✓ ML inference on streaming data in real-time (e.g. edge, near-data)

✓ Complex and dynamic data patterns (e.g. data drifts, concept drifts)

✓ No ground truth labels available (e.g. no human intervention)

When *not* to use Caravan

✕ ML inference on offline data (e.g. analytics of batch or historical data)

✕ Simple and static data patterns (e.g. small local area networks)

✕ Ground truth labels readily available (e.g. human-in-the-loop)

More details in our paper

paper

code

- User interface of Caravan
- Effectiveness of weak supervision labels
- GPT-4 labeling prompts
- Example of GPT-4 generations
- System cost and latency analysis
- Artifact (software + hardware)

…

CARAVAN: Practical Online Learning of In-Network ML Models with Labeling Agents

Qizheng Zhang, Ali Imran[†], Enkeleda Bardhi[‡], Tushar Swamy, Nathan Zhang, Muhammad Shahbaz[†*], Kunle Olukotun

Stanford University [†]*Purdue University* [‡]*Sapienza University of Rome* [*]*University of Michigan*

Abstract

Recent work on in-network machine learning (ML) anticipates offline models to operate well in modern networking environments. However, upon deployment, these models struggle to cope with fluctuating traffic patterns and network conditions and, therefore, must be validated and updated frequently in an online fashion.

This paper presents CARAVAN, a practical online learning system for in-network ML models. We tackle two primary challenges in facilitating online learning for networking: (a) the automatic labeling of evolving traffic and (b) the efficient monitoring and detection of model performance degradation to trigger retraining. CARAVAN repurposes existing systems (e.g., heuristics, access control lists, and foundation models)—not directly suitable for such dynamic environments—into high-quality labeling sources for generating labeled data for online learning. CARAVAN also introduces a new metric, ac-

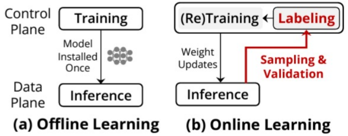


Figure 1: Comparison of in-network model learning. (a) Offline learning: trained and deployed once; (b) Online learning: trained and updated over time—requires iterative sampling, labeling, and validation.

Unlike conventional approaches (e.g., hand-crafted heuristics and static rulesets), ML models are better at revealing hidden patterns and characteristics in vast amounts of high-dimensional data—such as network traffic [35, 54, 71, 104, 111, 116, 117]. However, most efforts on replacing traditional

Conclusion

paper code

- Large models, e.g. GPT-4, can guide small in-network ML models via *online learning* since they can be good sources of *labeling*
- Sparse online learning via *accuracy proxy* saves system resources from excessive retraining
- We present **Caravan** for practical online learning for in-network ML
  - Caravan keeps in-network ML models up-to-date with changing traffic dynamics
  - Caravan reduces backend CPU usage by an average of 56.23% from excessive retraining
- Questions? Email me: qizhengz@stanford.edu