

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

*This project was published at **OSDI 2024**.*

*A shorter version was also presented at the **PACMI'24** workshop.*

Roadmap

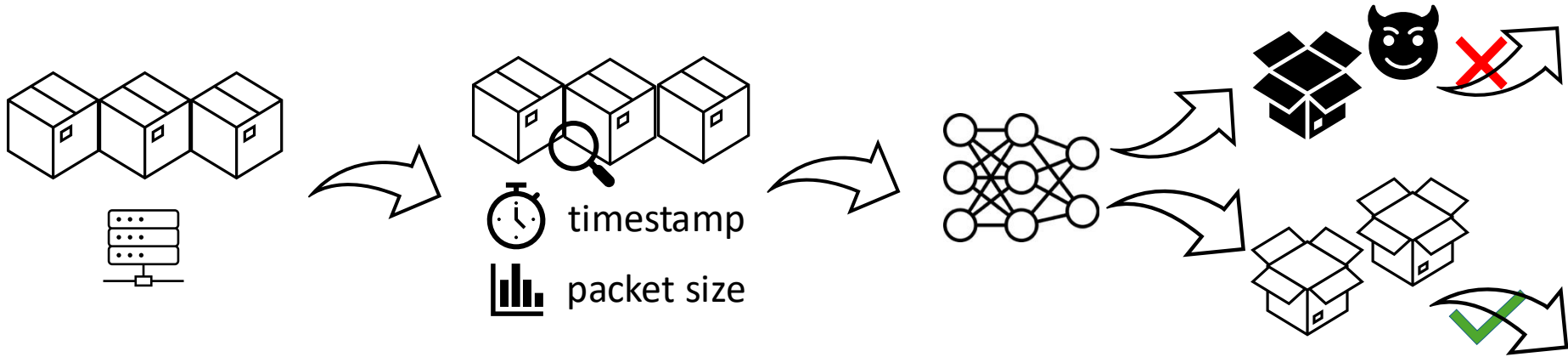
- ML + Online traffic analysis in networking
- Two approaches: Small models (fast) v.s. Large models (accurate)
- How to achieve the benefits of both? [REDACTED]
 - Three challenges and two insights to enable [REDACTED]
- Putting insights together: Caravan
- Limitations and future work

Roadmap

- ML + Online traffic analysis in networking
- Two approaches: Small models (fast) v.s. Large models (accurate)
- How to achieve the benefits of both? [REDACTED]
 - Three challenges and two insights to enable [REDACTED]
- Putting insights together: Caravan
- Limitations and future work

Machine learning (ML) in online traffic analysis

- Motivating use case: Intrusion detection in a network



(1) Incoming packets

(2) Feature extraction

(3) ML inference

(4) Drop or keep packets

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



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

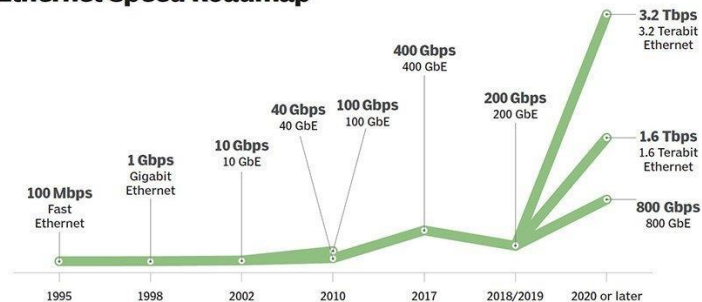
Roadmap

- ML + Online traffic analysis in networking
- Two approaches: Small models (fast) v.s. Large models (accurate)
- How to achieve the benefits of both? [REDACTED]
 - Three challenges and two insights to enable [REDACTED]
- Putting insights together: Caravan
- Limitations and future work

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: 100 μ s (2008) to 5 μ s (2023)
- Strict latency & throughput requirements
 - A need for small-batch or per-packet inference

Ethernet Speed Roadmap



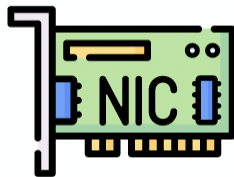
SOURCE: ETHERNET ALLIANCE

Small and specialized in-network models (fast)

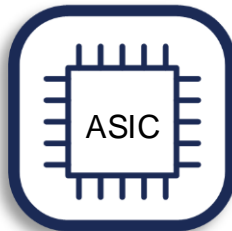
- **In-network ML** in data plane devices for real-time, per-packet inference



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



SmartNICs
E.g. N3IC [NSDI '22]



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

References

[1] Leo: Online ML-based Traffic Classification at Multi-Terabit Line Rate (NSDI '24)

[2] Re-architecting Traffic Analysis with Neural Network Interface Cards (NSDI '22)

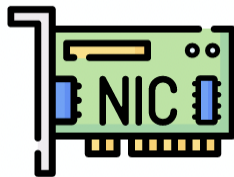
[3] Taurus: a data plane architecture for per-packet ML (ASPLOS '22)

Small and specialized in-network models (fast)

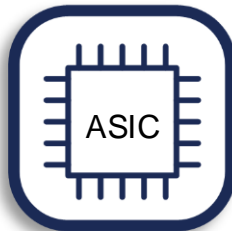
- **In-network ML** in data plane devices for real-time, per-packet inference



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*

References

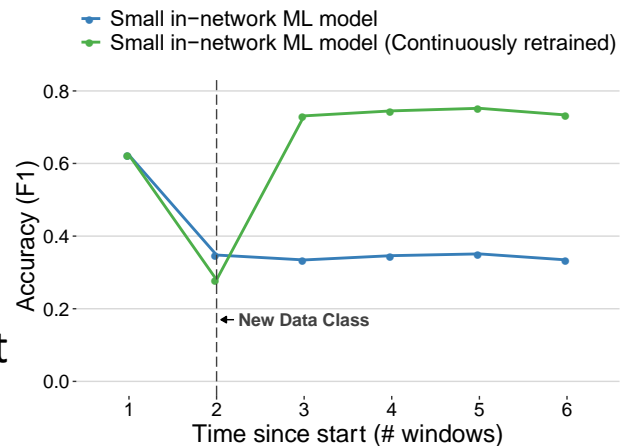
[1] Leo: Online

[2] Re-architecting Traffic Analysis with Neural Network Interface Cards (NSDI '22)

[3] Taurus: a data plane architecture for per-packet ML (ASPLOS '22)

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



In this work, we assume that all packet payload contents are unavailable, so payload-based approaches like DPI cannot be used.

Large and versatile foundation models (accurate)

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

References

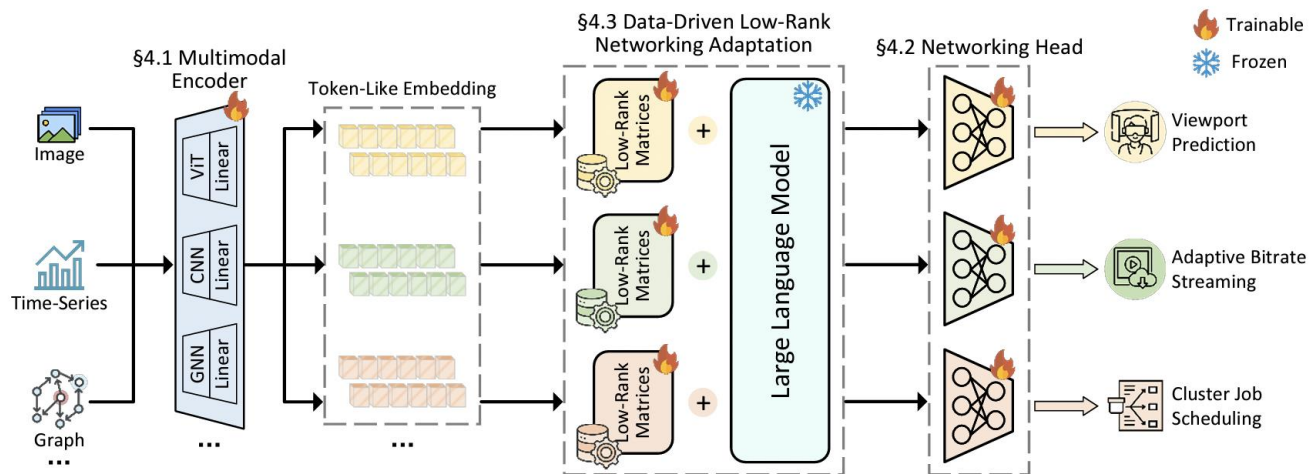
[1] NetLLM: Adapting Large Language Models for Networking (SIGCOMM '24)

[2] netFound: Foundation Model for Network Security (arXiv)

[3] Microsoft Copilot for Security. <https://www.microsoft.com/en-us/security/business/ai-machine-learning/microsoft-copilot-security>.

Large and versatile foundation models (accurate)

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



NetLLM [SIGCOMM '24]

References

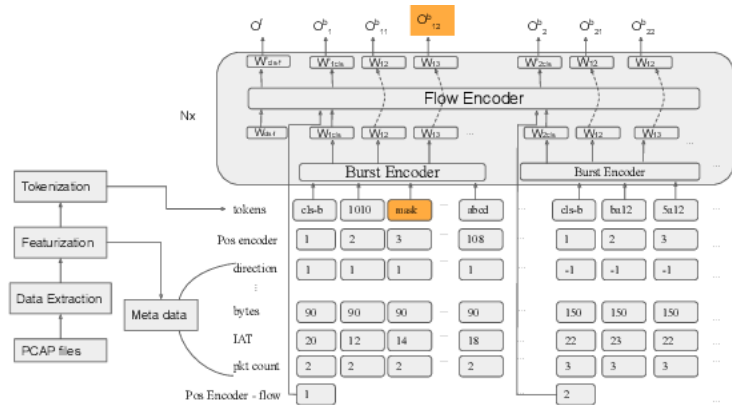
[1] NetLLM: Adapting Large Language Models for Networking (SIGCOMM '24)

[2] netFound: Foundation Model for Network Security (arXiv)

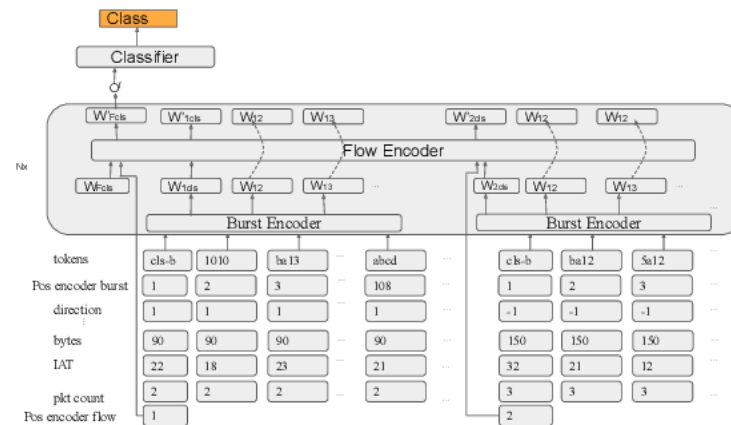
[3] Microsoft Copilot for Security. <https://www.microsoft.com/en-us/security/business/ai-machine-learning/microsoft-copilot-security>.

Large and versatile foundation models (accurate)

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



(a) Pre-training



(b) Fine-tuning

netFound [arXiv]

References

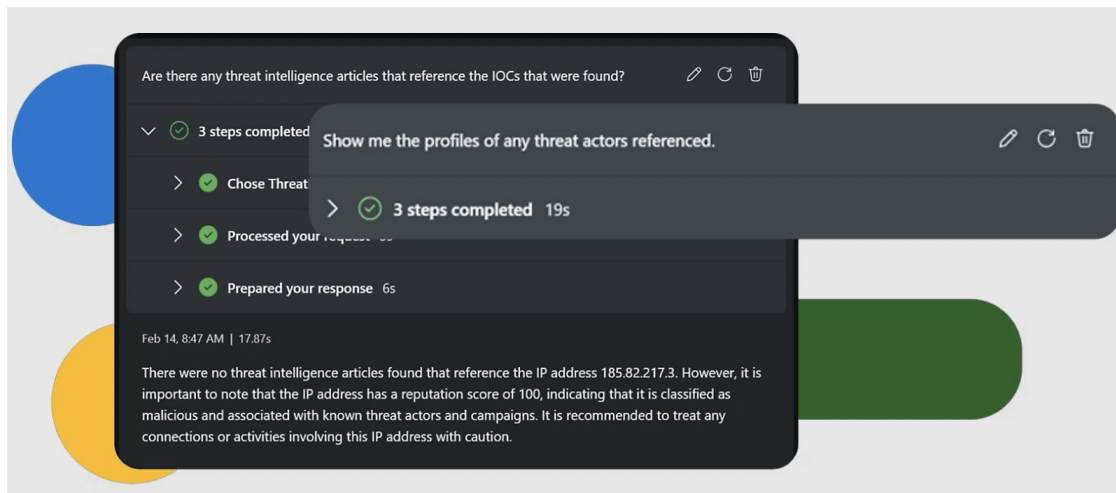
[1] NetLLM: Adapting Large Language Models for Networking (SIGCOMM '24)

[2] netFound: Foundation Model for Network Security (arXiv)

[3] Microsoft Copilot for Security. <https://www.microsoft.com/en-us/security/business/ai-machine-learning/microsoft-copilot-security>.

Large and versatile foundation models (accurate)

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



Microsoft Copilot for Security

References

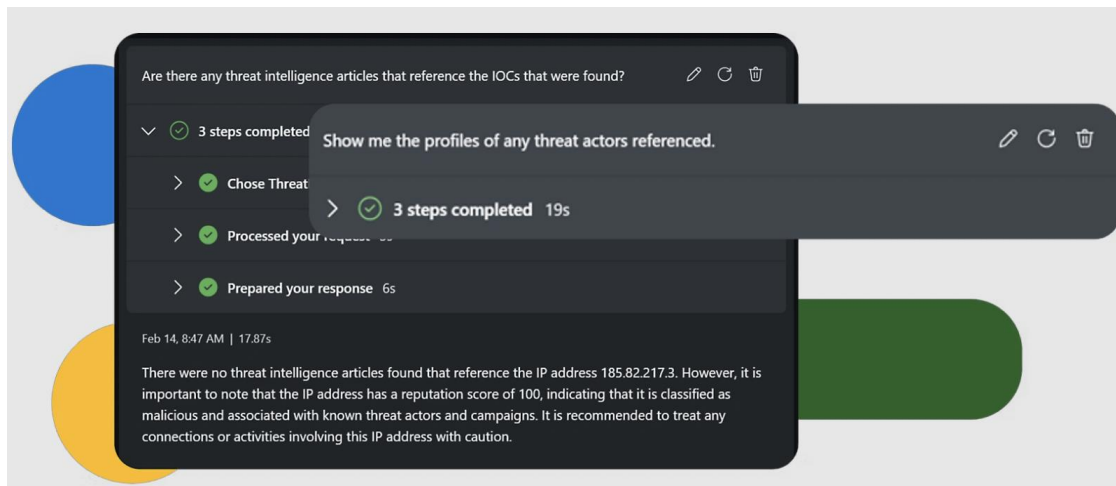
[1] NetLLM: Adapting Large Language Models for Networking (SIGCOMM '24)

[2] netFound: Foundation Model for Network Security (arXiv)

[3] Microsoft Copilot for Security. <https://www.microsoft.com/en-us/security/business/ai-machine-learning/microsoft-copilot-security>.

Large and versatile foundation models (accurate)

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



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

References

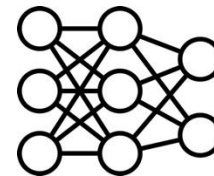
[1] NetLLM: Ad

[2] netFound: Foundation Model for Network Security (arXiv)

[3] Microsoft Copilot for Security. <https://www.microsoft.com/en-us/security/business/ai-machine-learning/microsoft-copilot-security>.

Two approaches: Small and large models

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



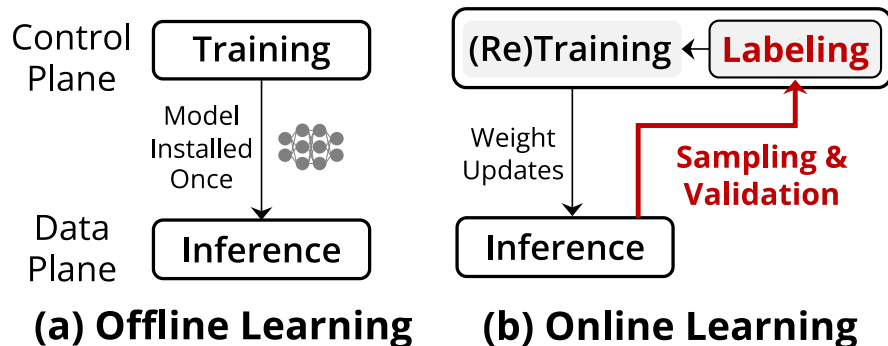
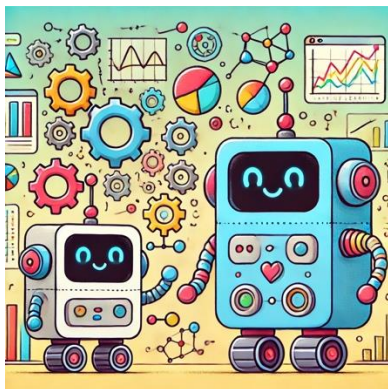
#2: **Large** and versatile **foundation** models (accurate)



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

Our proposal: *Online learning* to the rescue

- Large and small models should work *jointly* online



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

Roadmap

- ML + Online traffic analysis in networking
- Two approaches: Small models (fast) v.s. Large models (accurate)
- How to achieve the benefits of both? Online Learning
 - Three challenges and two insights to enable practical online learning
- Putting insights together: Caravan
- Limitations and future work

Three challenges of using large models (e.g. FMs) online

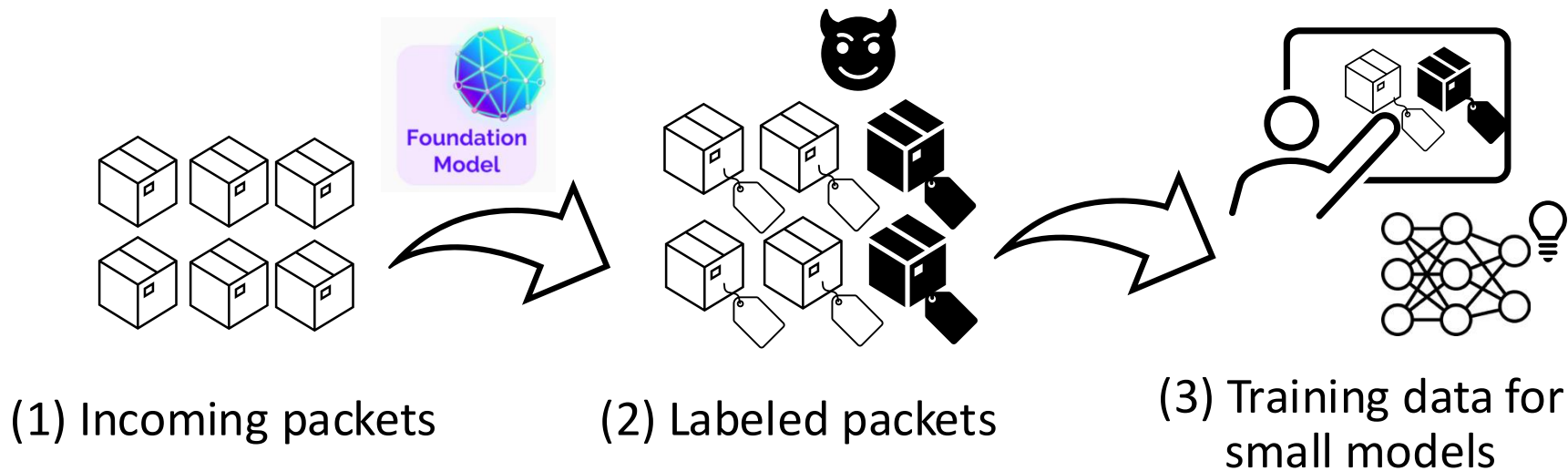
High cost

Low speed

Poor control over output

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).



Labeling can be cheap, fast, and controllable online

- Why *labeling* in an online system?
- We start with the naïve way of using a foundation model, e.g. GPT-4:



Help me analyze which of the following network flows are malicious. [Data here]



Of course, I'd be happy to help! I'll do my best to provide a benign or malicious classification based on the provided information and my knowledge.

Network flow 1:

- Decision: FTP-Patator
- Explanation: The dst port is 21 (FTP). The pattern is very similar to FTP-Patator from the labeled data, with many SYN and ACK flags and a low average packet length.

.....

Labeling can be cheap, fast, and controllable online

- Let's adapt this chat to be a *labeling* request:



Help me analyze which of the following network flows are malicious. **I only need a binary label for each flow.** [Data here]



1
0
1
0
0
1
.....

Labeling can be cheap, fast, and controllable online

- Let's adapt this chat to be a *labeling* request:



Help me analyze which of the following network flows are malicious. **I only need a binary label for each flow.** [Data here]



1
0
1
0
0
1
.....

Better control over output

Labeling can be cheap, fast, and controllable online

- Let's adapt this chat to be a *labeling* request:



Help me analyze which of the following network flows are malicious. **I only need a binary label for each flow.** [Data here]



1
0
1
0
0
1
.....

Better control over output

Labeling can be cheap, fast, and controllable online

- Instead of repeatedly querying an LLM to label data, we ask it to generate a (distilled) *labeling program* that can be cached:



Help me analyze which of the following network flows are malicious. **Generate a Python program for this.** [Data here]



```
def label_network_flow (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.
```

```
    """
```

Labeling can be cheap, fast, and controllable online

- Instead of repeatedly querying an LLM to label data, we ask it to generate a (distilled) *labeling program* that can be cached:



Help me analyze which of the following network flows are malicious. **Generate a Python program for this.** [Data here]



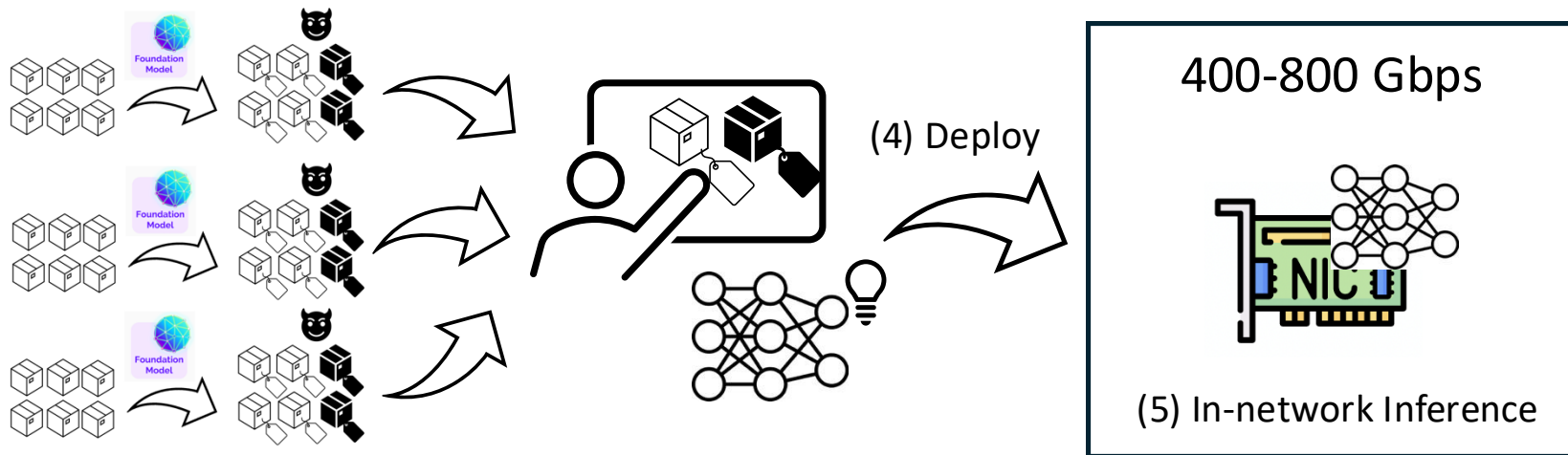
```
def label_network_flow (flow):  
    """  
    Labels a network flow as benign (0) or malicious (1) based on certain heuristics.  
    This function is designed to be conservative and  
    is highly confident.  
  
    :param flow: A tuple containing features of the  
    :return: An integer label, 0 for benign and 1 for  
    """
```

Lower cost

Higher speed

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

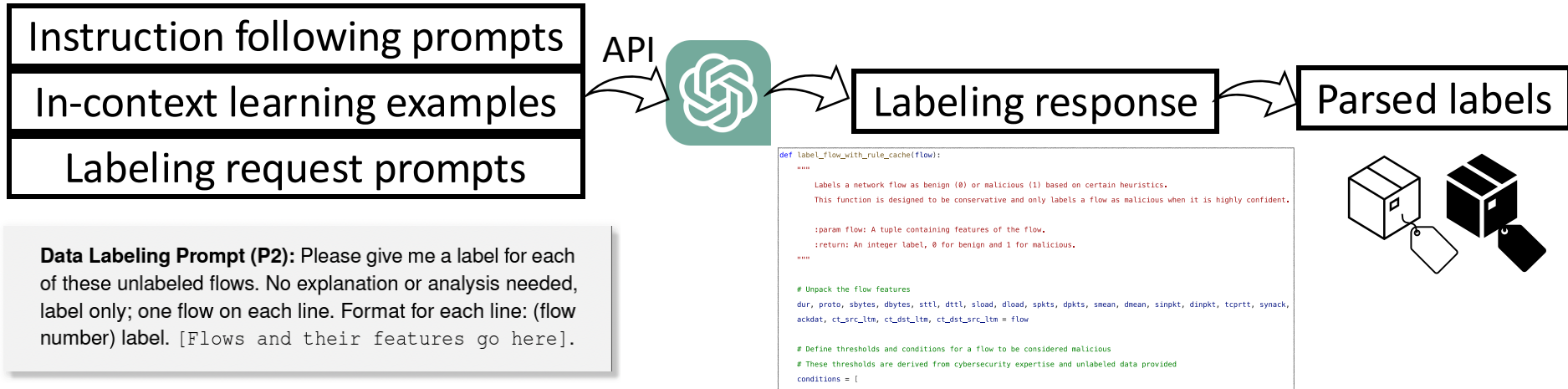
- Data labeling & online learning **do not** need to happen in real-time.
 - Further acceleration through large-batch inference, parallelization, etc.



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

Example: Adapting GPT-4 as a labeling source

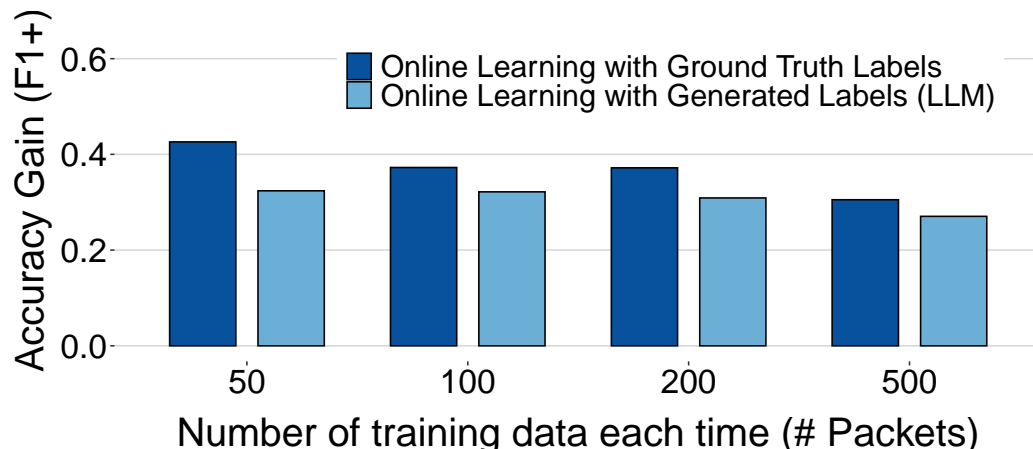
- We adapted GPT-4 for data labeling in the intrusion detection use case.



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

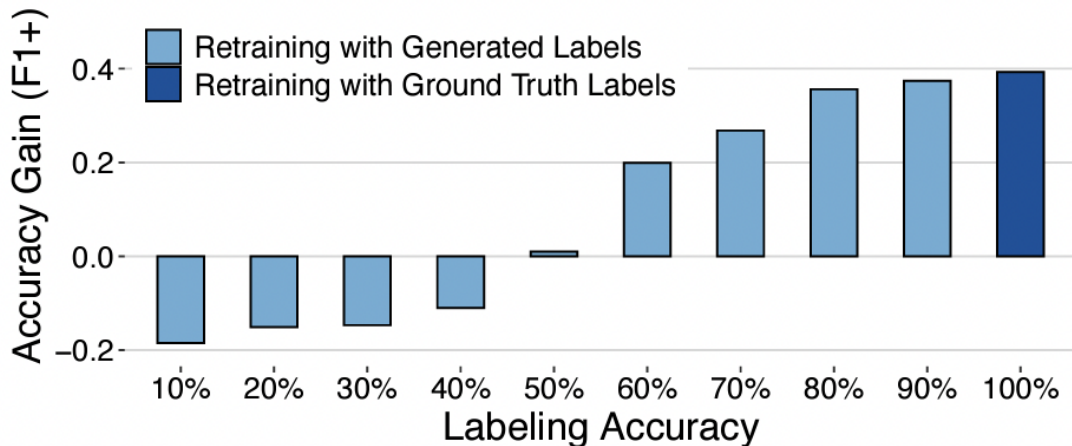
Generated labels from GPT-4 for online learning

- We use generated labels from GPT-4, as well as ground truth labels (from dataset publishers), for online learning.
- Result: The accuracy gains from online learning are comparable.



Caveat: *Accurate* labels are essential

- Foundation models can produce noisy labels (partially incorrect).
 - And highly inaccurate generated labels can backfire.
- If ground truth labels are available online in the network, use them!



Caveat: *Accurate* labels are essential

- We discuss potential solutions to this problem in the OSDI paper.
 - Offline
 - *Benchmarking* the performance of foundation models on domain-specific tasks

Category: **Math** #models: 127 (93%) #votes: 228,144 (13%)

Rank* (UB)	Delta	Model	Arena Score	95% CI	Votes	Organization	License	Knowledge Cutoff
1 ↑	5	Claude 3.5 Sonnet	1274	+7/-8	7138	Anthropic	Proprietary	2024/4
1 ↑	1	Gemini-1.5-Pro-Exp-0827	1271	+9/-8	3431	Google	Proprietary	2023/11
1	0	ChatGPT-4o-latest (2024-08-08)	1269	+9/-10	4029	OpenAI	Proprietary	2023/10
1 ↑	1	Gemini-1.5-Pro-Exp-0801	1258	+9/-8	3723	Google	Proprietary	2023/11
2 ↑	3	GPT-4o-2024-05-13	1257	+7/-6	10817	OpenAI	Proprietary	2023/10
2	0	Grok-2-08-13	1252	+14/-13	1258	xAI	Proprietary	2024/3

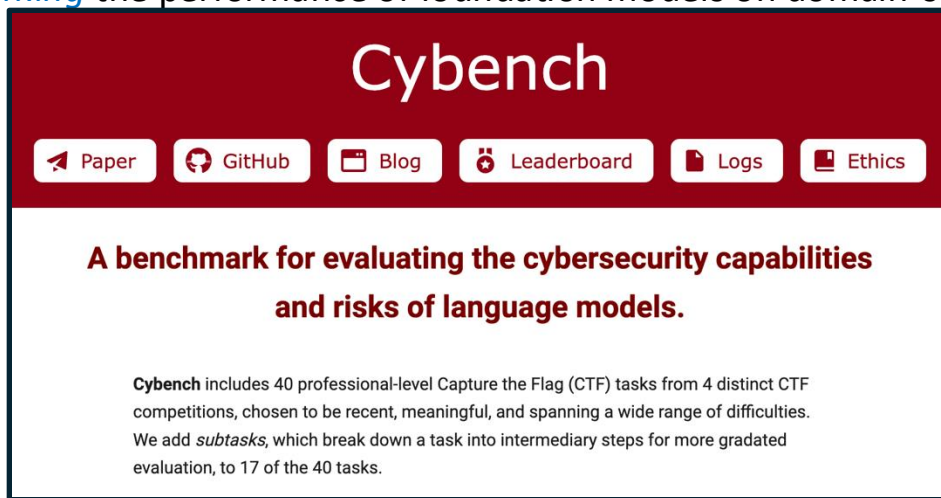
References

[1] Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena (NeurIPS '23)

[2] Cybench: A Framework for Evaluating Cybersecurity Capabilities and Risk of Language Models (arXiv)

Caveat: *Accurate* labels are essential

- We discuss potential solutions to this problem in the OSDI paper.
 - Offline
 - *Benchmarking* the performance of foundation models on domain-specific tasks



References


[1] Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena (NeurIPS '23)

[2] Cybench: A Framework for Evaluating Cybersecurity Capabilities and Risk of Language Models (arXiv)

Caveat: *Accurate* labels are essential

- We discuss potential solutions to this problem in the OSDI paper.
 - Offline
 - *Benchmarking* the performance of foundation models on domain-specific tasks
 - Online
 - *Validating* outputs from a foundation model

Unlocking LLM Confidence Through Logprobs

 Gautam Chutani · Follow
12 min read · Feb 18, 2024

30

In the realm of AI-driven responses, understanding the confidence level of generated text is crucial for evaluating model performance and enhancing user trust. The use of log probabilities (or logprobs) serves as a beacon of insight into the decision-making process of language models.

LLM Toolkit: Validation is all you need

MAY 20, 2024
BY JEFF SCHOMAY

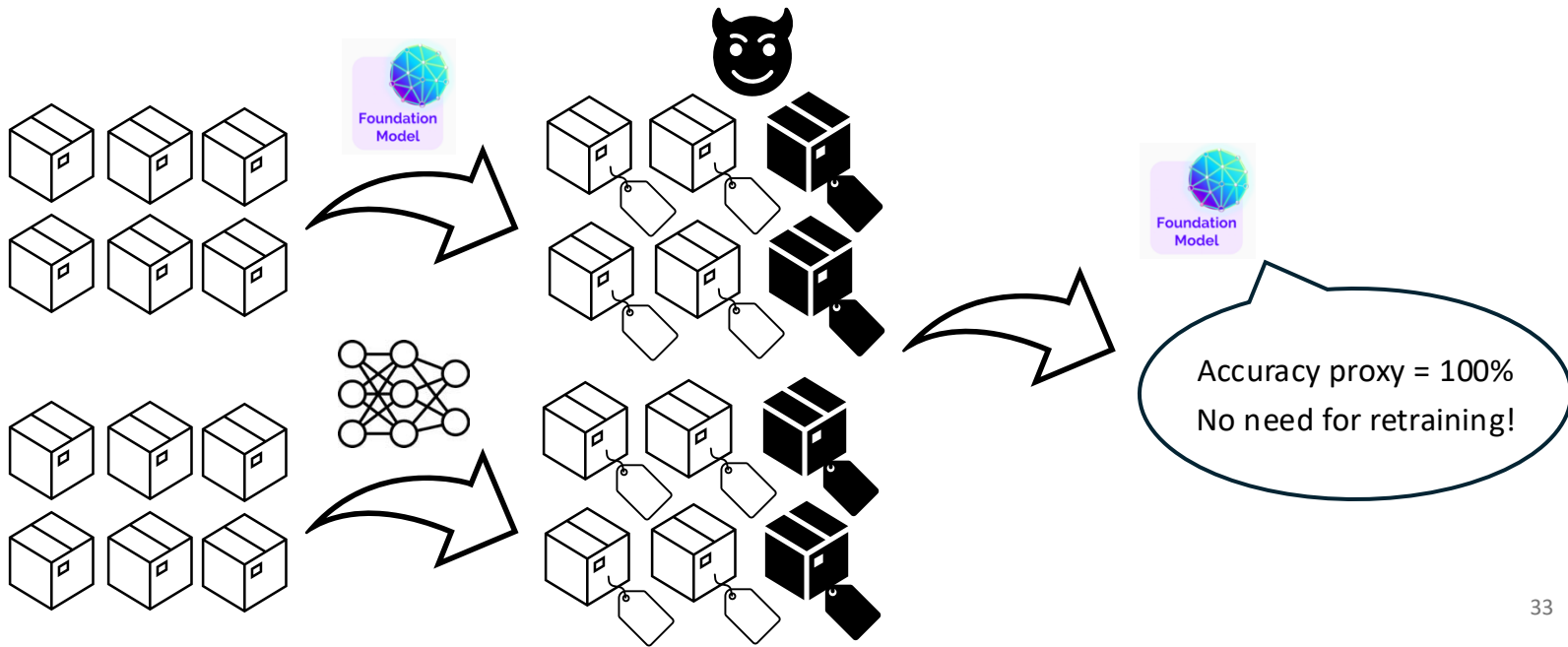
BLOG

Forget chains—structured output and validation are all you need.

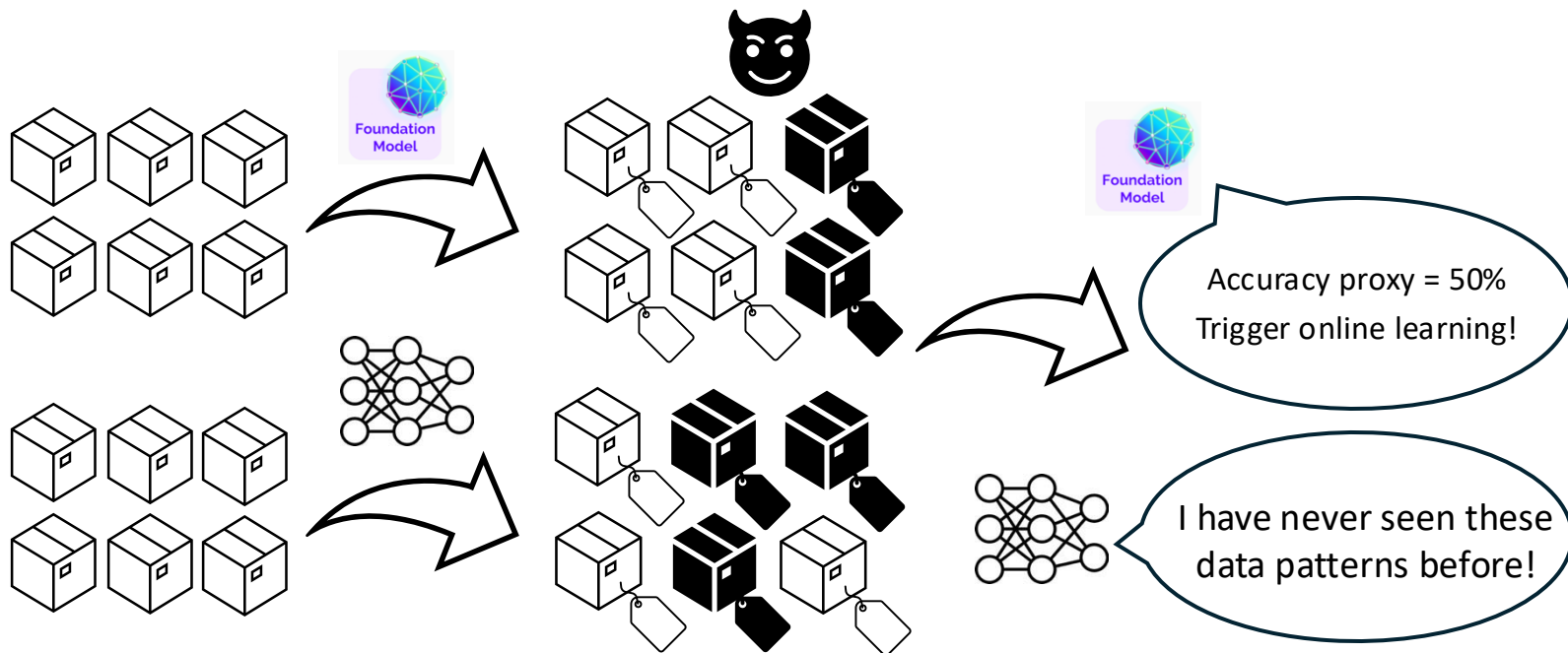
At Mechanical Orchard, we're building complex, bespoke, explainable AI agents to interrogate legacy mainframe systems autonomously. We've used a wide range of popular AI libraries and frameworks to bring LLM best practices into our codebase. The one I've been absolutely loving is [Instructor](#), because modeling data is so much more powerful than modeling prompts. In this technical article, I'll show you why.

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



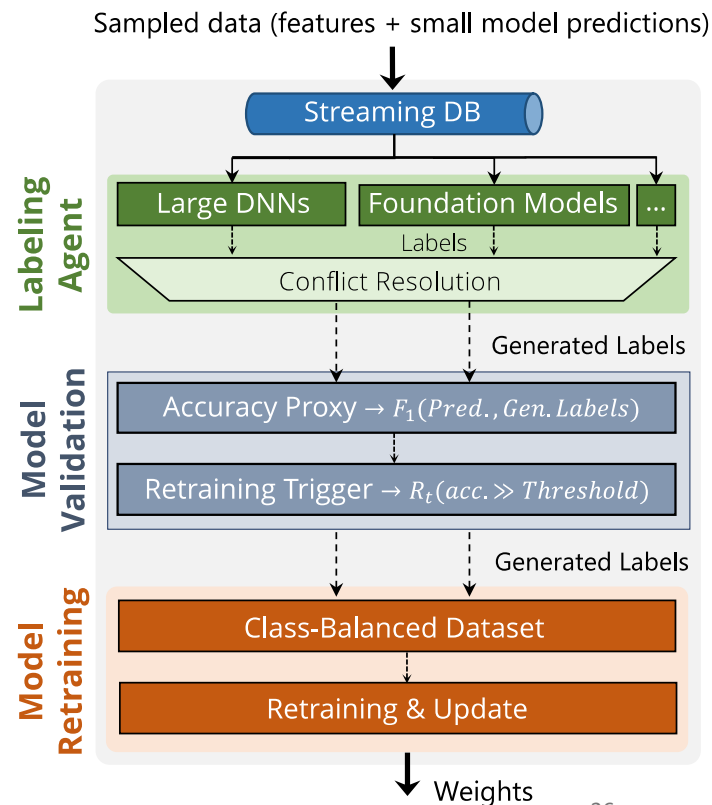
Sparse online learning via *accuracy proxy* avoids excessive retraining

Roadmap

- ML + Online traffic analysis in networking
- Two approaches: Small models (fast) v.s. Large models (accurate)
- How to achieve the benefits of both? Online Learning
 - Three challenges and two insights to enable practical online learning
- Putting insights together: Caravan
- Limitations and future work

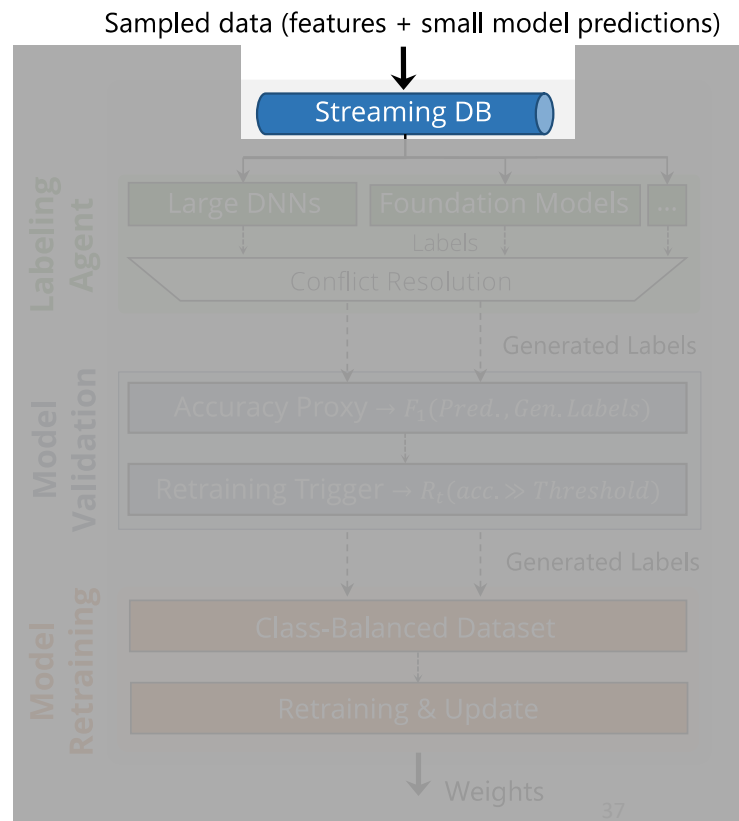
Putting them together (Caravan)

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



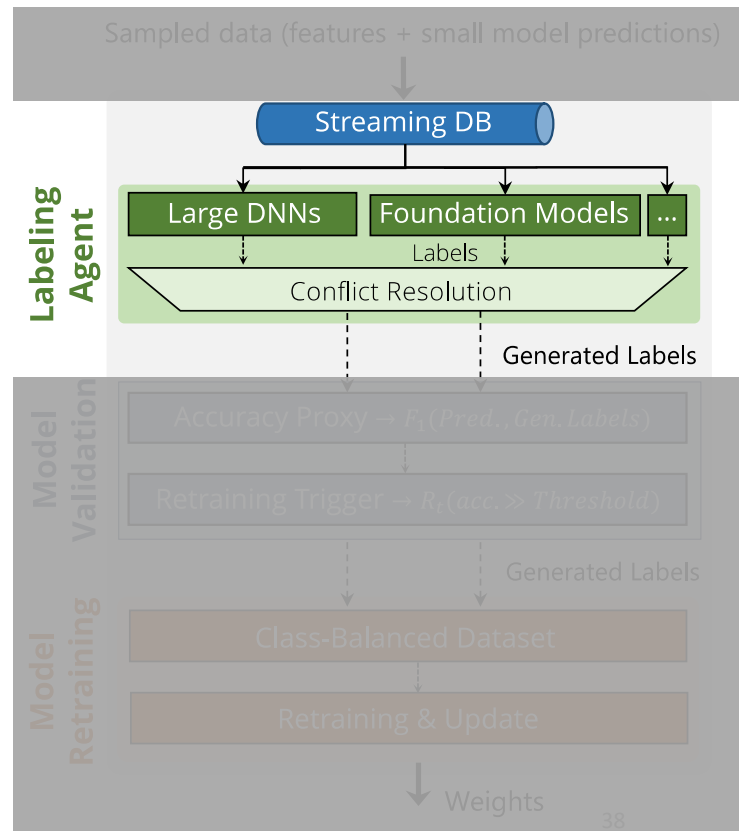
Putting them together (Caravan)

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



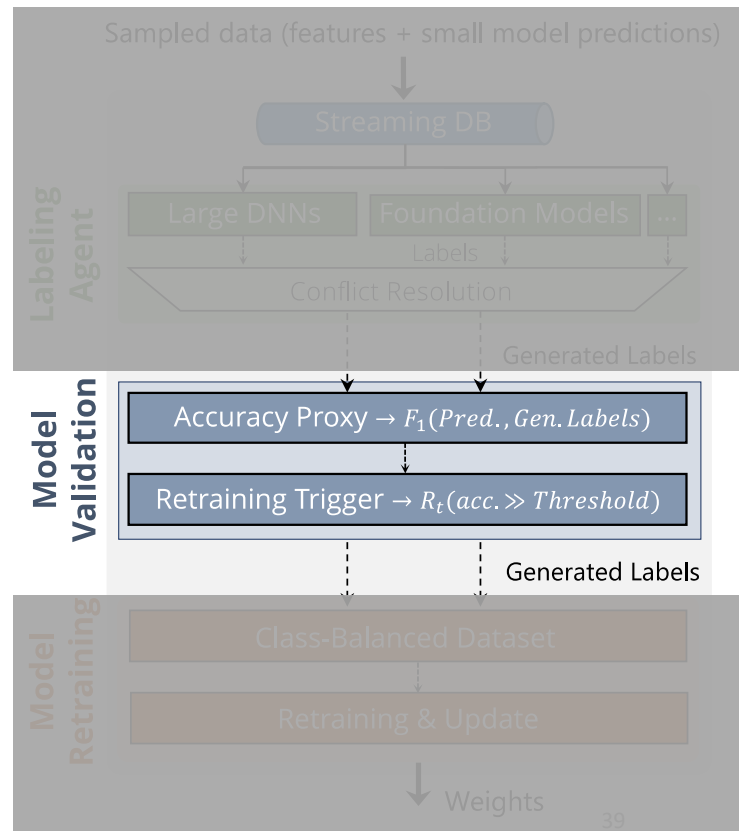
Putting them together (Caravan)

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



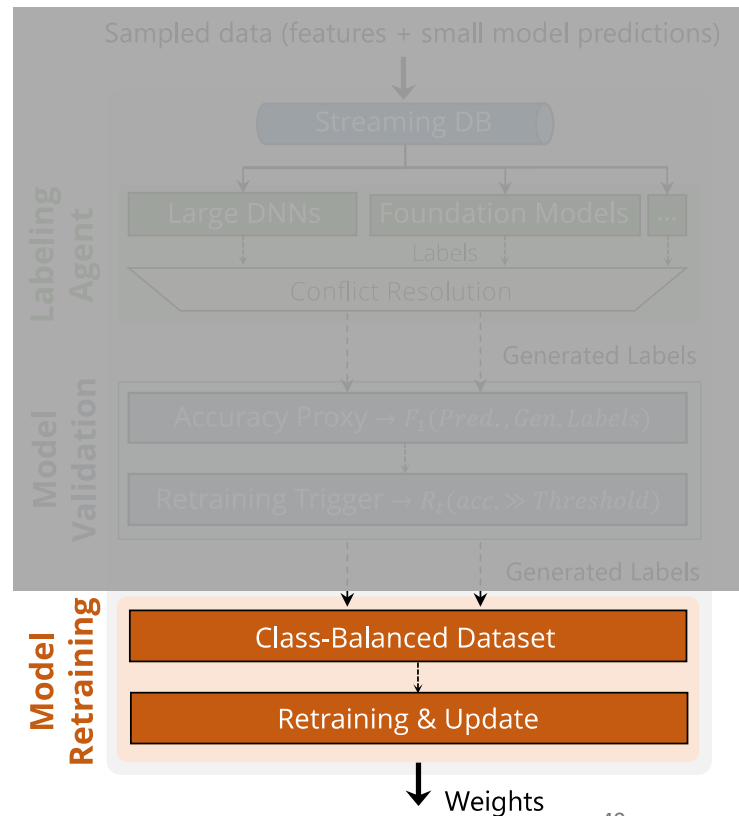
Putting them together (Caravan)

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



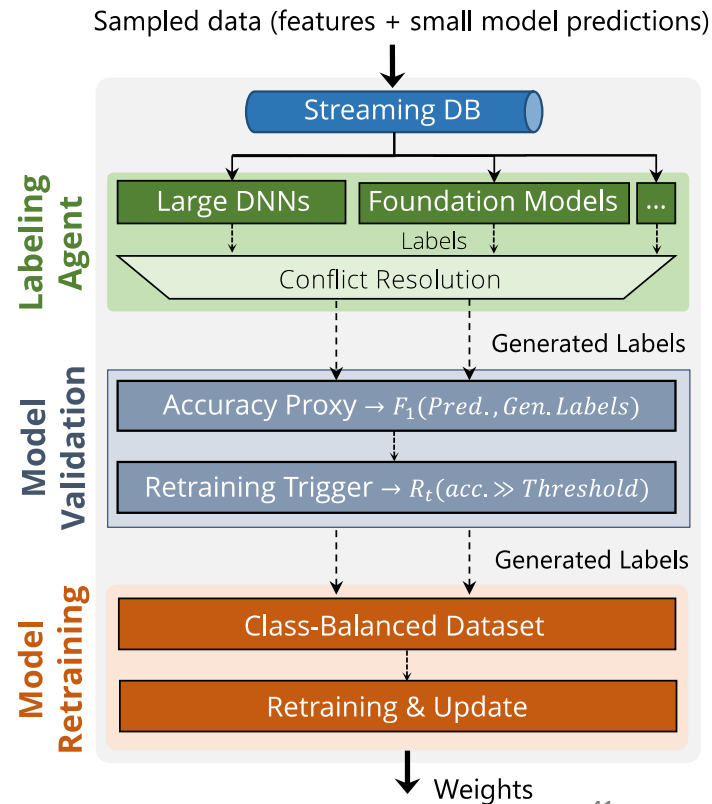
Putting them together (Caravan)

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

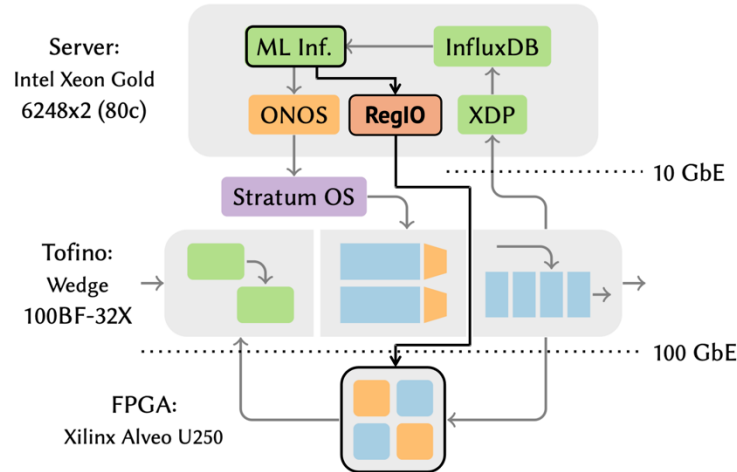


Putting them together (Caravan)

- Caravan: A system for practical online learning of in-network ML models
 - Labeling agent
 - Model validation
 - Model retraining

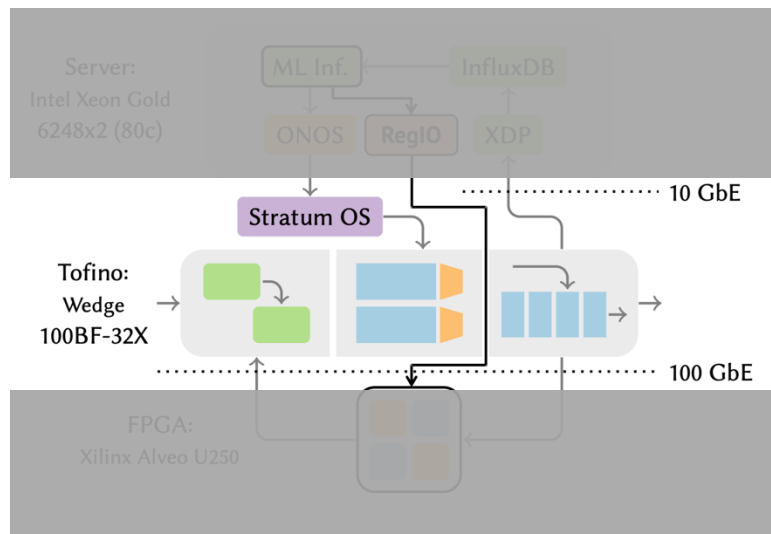


Implementation: Three-piece prototype



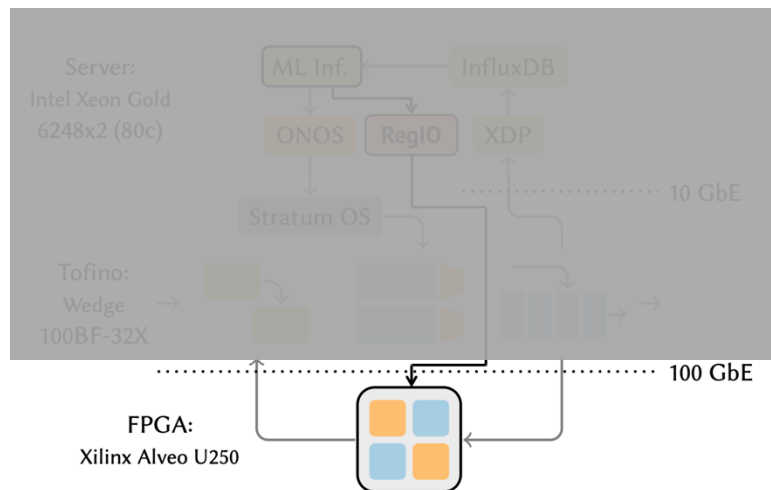
Implementation: Three-piece prototype

- A *Tofino switch* for packet parsing and deparsing
- We send and receive packets with MoonGen (IMC '15).



Implementation: Three-piece prototype

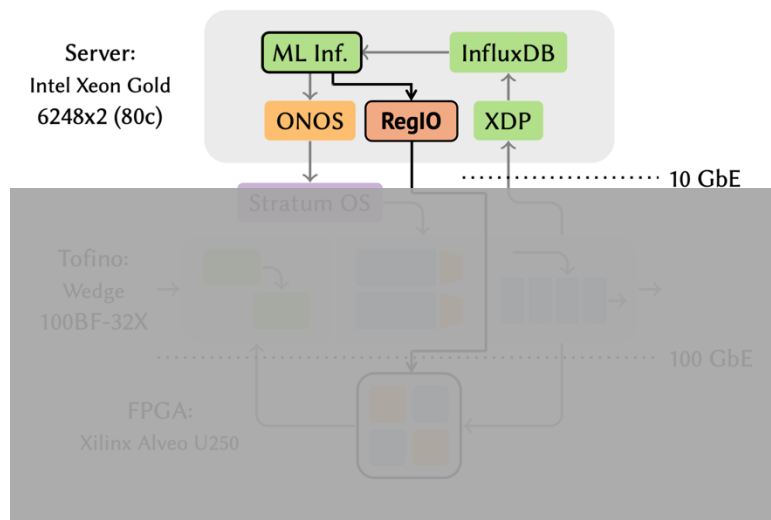
- An **FPGA** for running the in-network ML model
 - We program the architecture of the ML model with the Spatial language (PLDI '18), a Scala-based higher-level hardware description language.



Thanks to Spatial, no Verilog-level programming needed. However, FPGA simulation is still the most time-consuming part of the implementation...

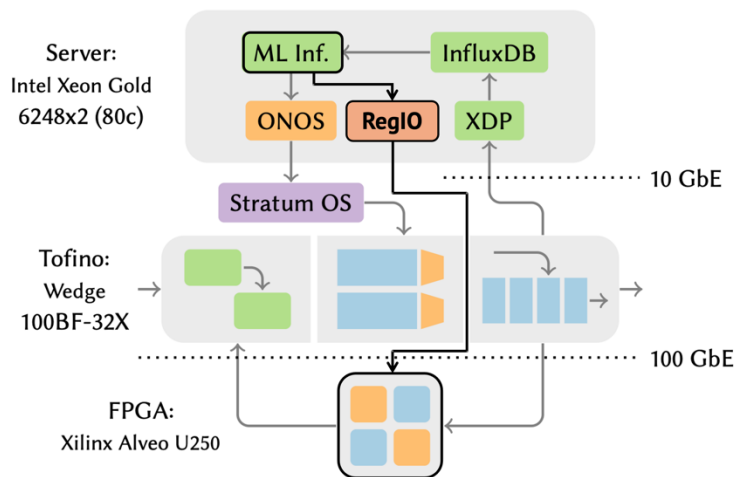
Implementation: Three-piece prototype

- A *compute server* for the Caravan software
 - In the end-to-end experiment, we use CPU for online model training as the sampled training dataset size is relatively small.
 - For simulations and micro-benchmarking (details in the paper), we use GPU.



Implementation: Three-piece prototype

- 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 time, memory usage, FPGA usage

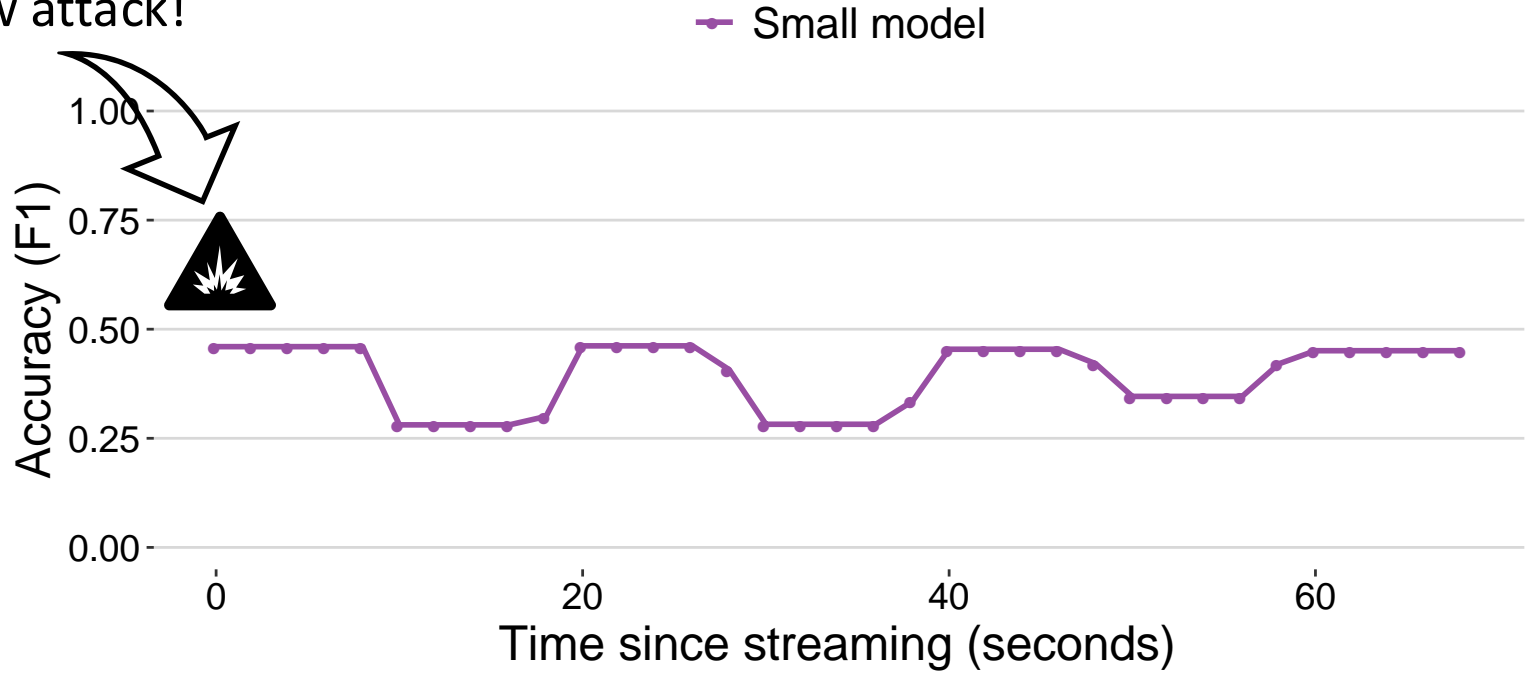
Example: End-to end intrusion detection

- 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

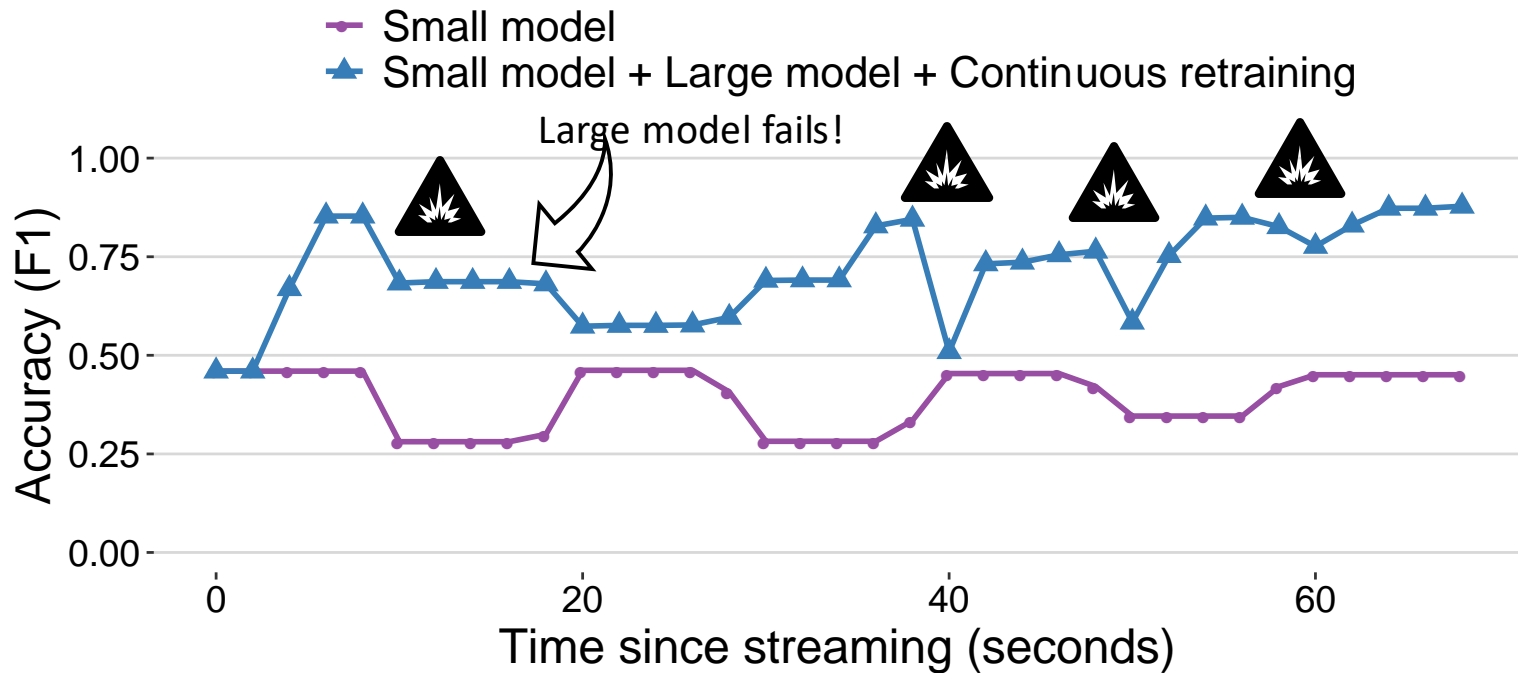
- Packet rate: 0.5 million packets/sec
- Run inference + compute accuracy on *every* packet
- Sample rate for the control plane: 0.1%

We start with the small in-network model

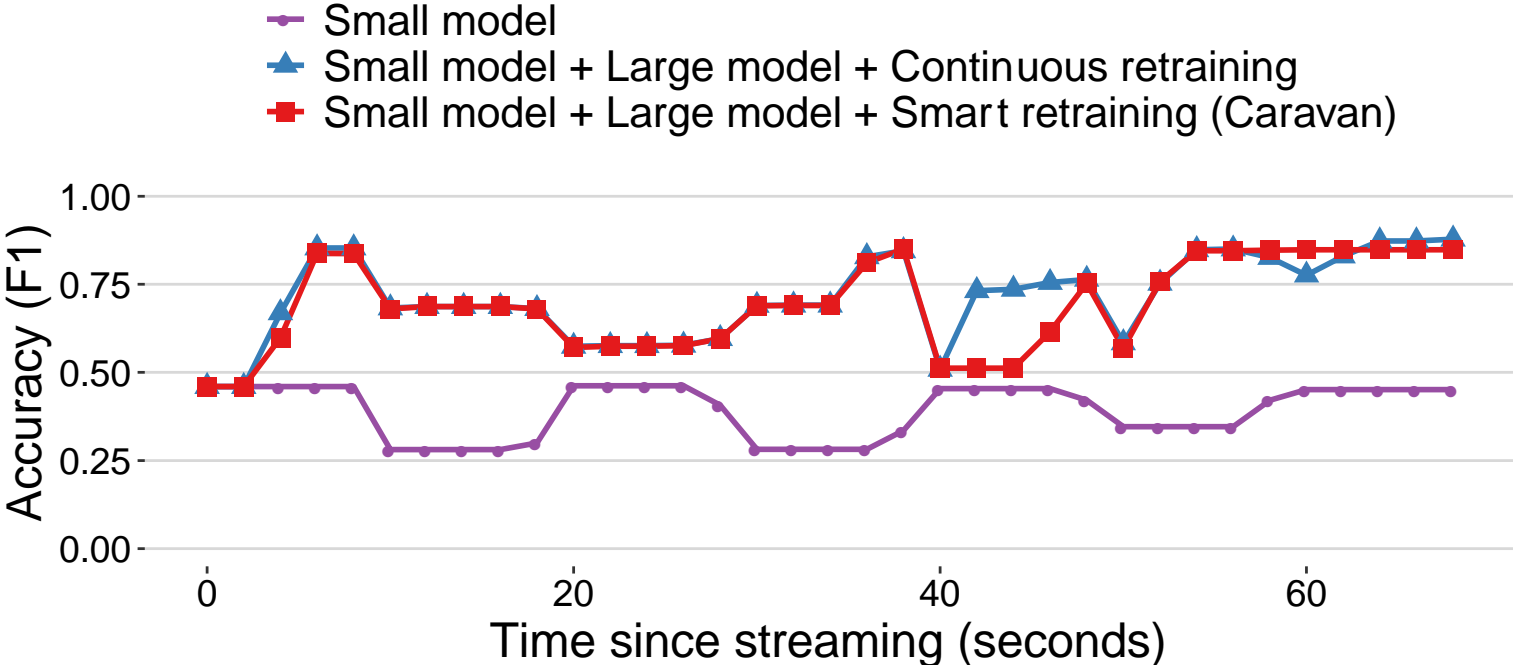
New attack!



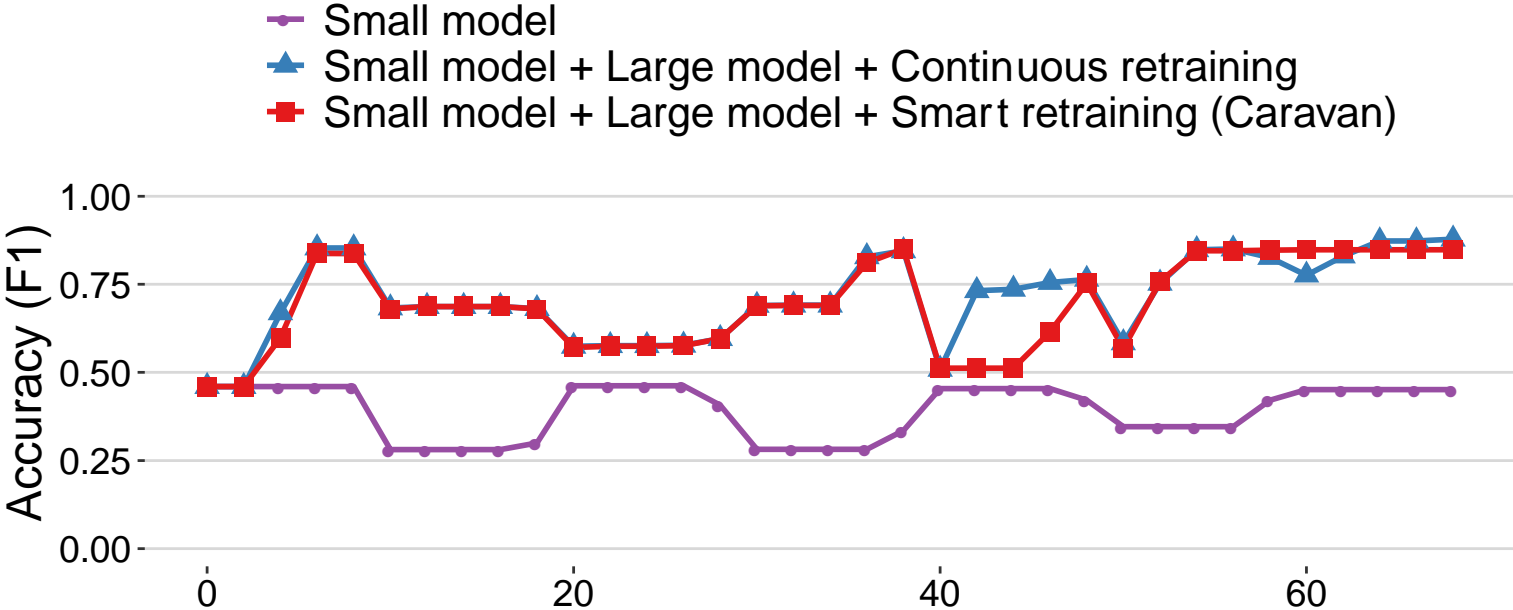
What if the large model guides the small model (via online learning)?



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

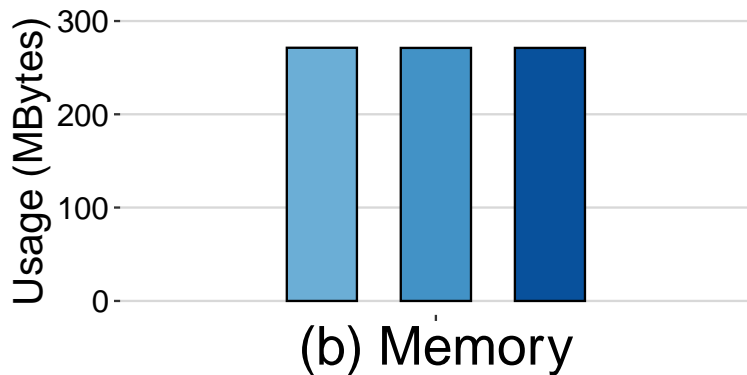
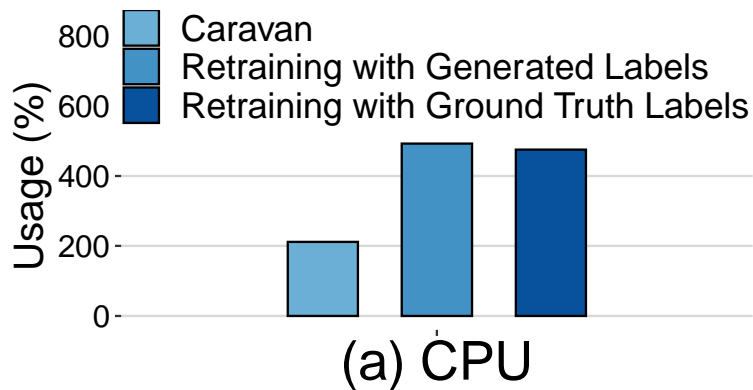


What if we introduce selective retraining via accuracy proxy (Caravan)?



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%

Roadmap

- ML + Online traffic analysis in networking
- Two approaches: Small models (fast) v.s. Large models (accurate)
- How to achieve the benefits of both? Online Learning
 - Three challenges and two insights to enable practical online learning
- Putting insights together: Caravan
- Limitations and future work

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 OSDI paper



paper



artifact

- 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-*

(a) Offline Learning

(b) Online Learning

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 approaches (e.g., heuristics and access control lists) with

Conclusion



paper



artifact

- Two approaches for ML-based online traffic analysis: Small models (fast) v.s. Large models (accurate)
- How to achieve the benefits of both? [Online Learning](#)
 - Three challenges of large models online: High cost, low speed, poor control
 - Two insights: Large models as labeling sources, and sparse retraining
 - Putting insights together: Caravan
- Beyond networking
 - Self-improving AI systems
 - ML systems in dynamic environments
 - Integration of LLMs and autonomous systems

Conclusion



paper



artifact

- We are working on an open-source library for making Caravan (and generative data labeling) easy to use.
- I am open to questions, chats and collaborations.
 - Contact: qizhengz@stanford.edu



library