Learning-Enhanced High-Throughput Pattern Matching Based on Programmable Data Plane

Guanglin Duan, Yucheng Huang, Zhengxin Zhang, Qing Li, Dan Zhao,

Zili Meng, Dirk Kutscher, Ruoyu Li, Yong Jiang, and Mingwei Xu

清華大學

Tsinghua University

Cornell University

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY (GUANGZHOU)

Pattern matching is essential to many network applications

- Looking for known patterns in packet payload
  - String multi-string matching (Fixed-length string) and Regular expression matching (regex or PCRE)
  - 5K ~ 26K rules in public rule-sets for network applications

- Rule Examples
  - Rule 1
  - Rule 2
  - Rule 3

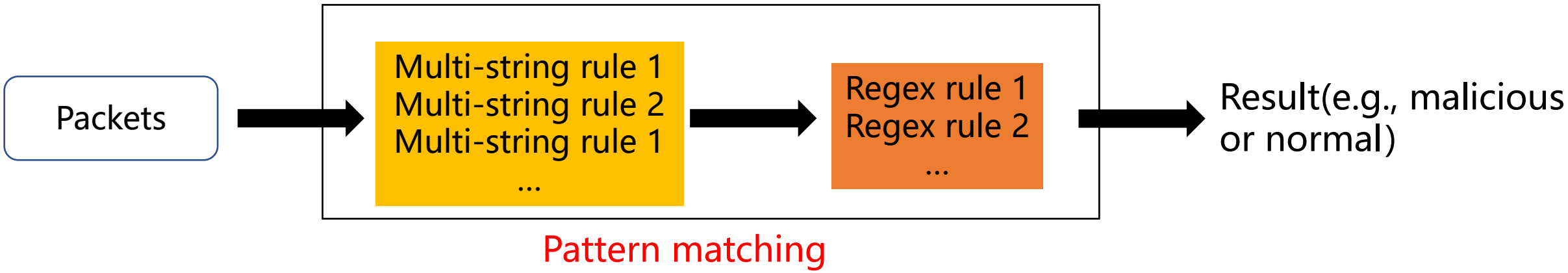
String pattern matching	Regular expression matching
Content:"MyWay";	PCRE:"/User-Agent\x3a[^\n]+MyWay/iH";
Content:"/upnp/";	PCRE:"/\/upnp\/[a-z0-9]{8}-[a-z0-9]{4}-[a-z0-9]{4}-[a-z0-9]{16}\//i";
Content:"SELECT ";	PCRE:" /SELECT.+geometrycollectionfromwkb/si ";

Pattern matching is essential to many network applications

- Network applications using pattern matching

Network Intrusion Detection	Web Application Firewall
Exfiltration Detection	Application Identification System

- Network Intrusion Detection

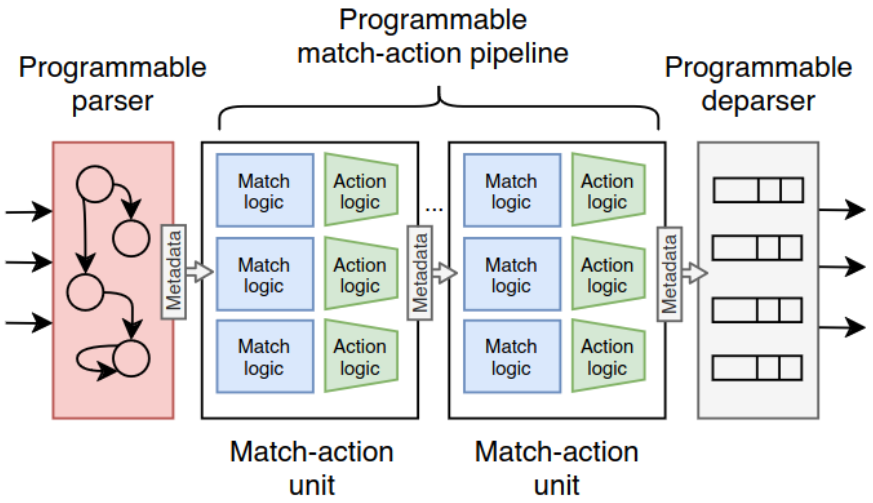


Limitations of Existing Approaches

1. CPU and hardware accelerators (GPU/FPGA/NPU) face challenges in cost-effectively sustaining high throughput as traffic and pattern complexity grow.

Limitations of Existing Approaches

1. CPU and hardware accelerators (GPU/FPGA/NPU) face challenges in cost-effectively sustaining high throughput as traffic and pattern complexity grow.
2. Programmable switch-based solutions



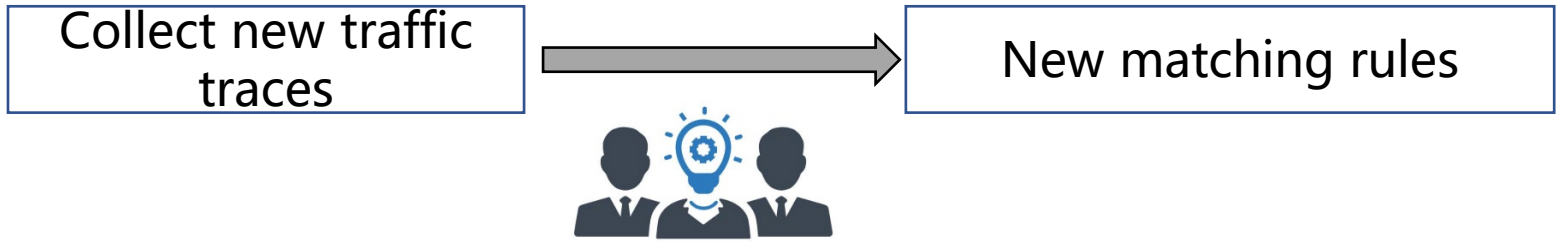
- Handle multi-Tbps traffic at a line rate with lower cost than GPU/FPGA/NPU/CPU
- Customizable Packet Processing

- Computation Constraints
  - simple operations，limited stages
- Storage Constraints
  - once register access，limited storage

Supports only limited multi-string matching and
cannot be extended to regular expression matching.

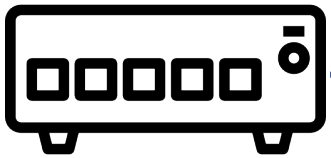
Limitations of Existing Approaches

1. CPU and hardware accelerators (GPU/FPGA/NPU) face challenges in cost-effectively sustaining high throughput as traffic and pattern complexity grow.
2. Programmable switch-based solutions: Supports only limited multi-string matching and cannot be extended to regular expressions.
3. Common limitation: Inefficiency in updating pattern sets to ensure sustained accuracy.
   - Existing rules are often crafted offline by experts or proprietary tools, making updates labor-intensive and costly.



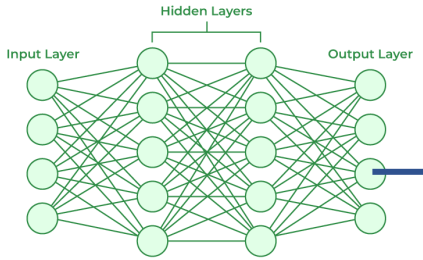
Design Goals

- Overcomes the limitations of existing approaches
  - High Throughput: Achieves high throughput at a low cost.
  - Accuracy: Maintains accurate pattern recognition
  - Manageability: Supports efficient updates for evolving traffic patterns.



Programmable switch

High throughput and low cost



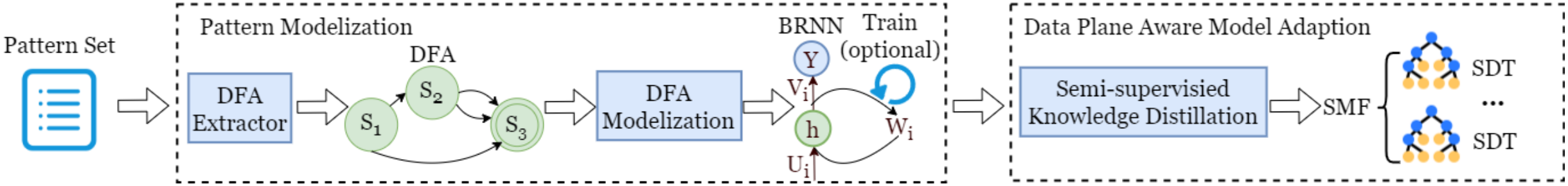
Neural network models

Learning ability

Trochilus: Learning-Enhanced High-Throughput Pattern Matching Based on Programmable Data Plane

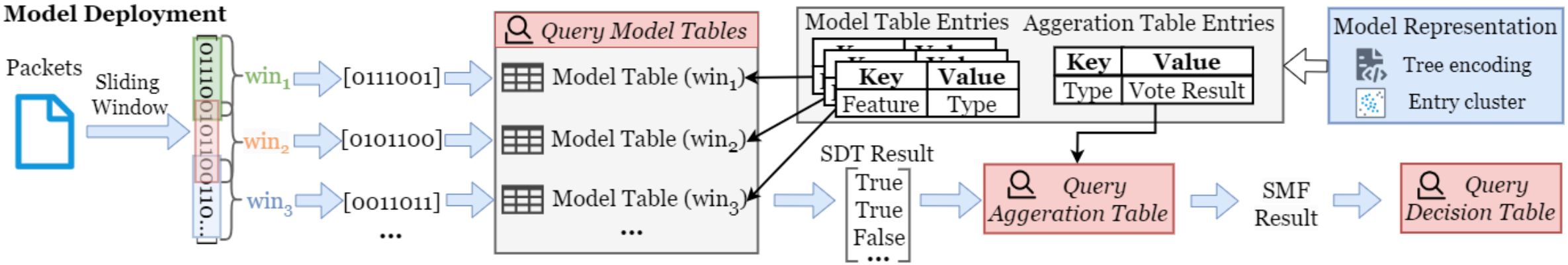
Challenge#1: How to replace traditional pattern matching with model inference?

- Challenge:
  - Data Scarcity: Labeled data is hard to obtain in real-world network environments.
  - Knowledge Reuse: How can we leverage existing rule sets effectively?
  - Hardware Constraints: How to build models that are hardware-friendly for deployment?
- Solution
  - Pattern Modelization
  - Data Plane Model Representation

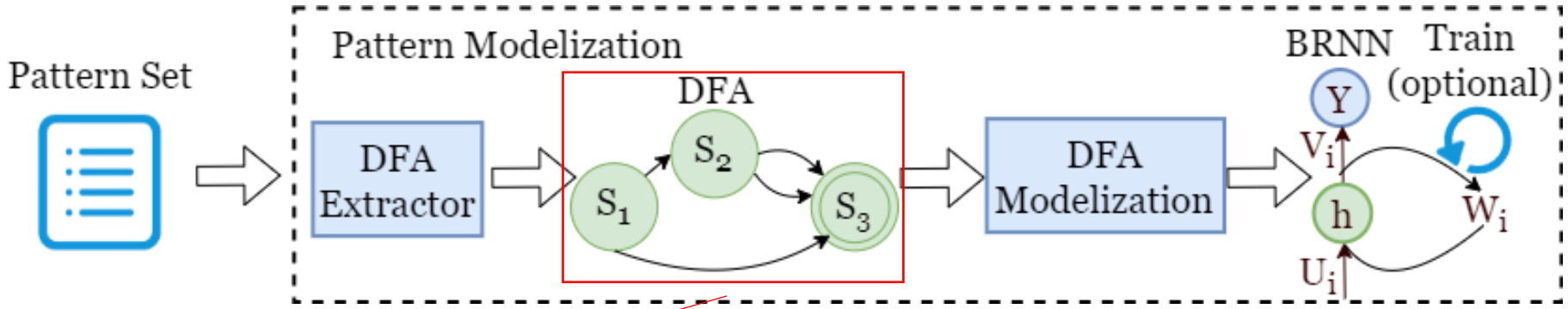


Challenge#2: How to deploy learning models efficiently?

- Hardware limitations:
  - Limited computational capability and storage resource
- Solutions
  - Tree Encoding & Entry Clustering
  - Sliding Window



Pattern Modelization



DFA

Input: $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$

Path: $\mathcal{P} = \{u_1, u_2, \ldots, u_{N+1}\}$

Score of forward algorithm:

$$\mathcal{B}_{\mathrm{fw}}(\mathcal{A}, \mathcal{X}) = \sum_{\mathcal{P} \in \pi(x)} \mathcal{B}(\mathcal{A}, \mathcal{P}) = \alpha_0^T \cdot \left(\prod_{i=1}^{N} \mathcal{W}[x_i] \right) \cdot \alpha_\infty .$$

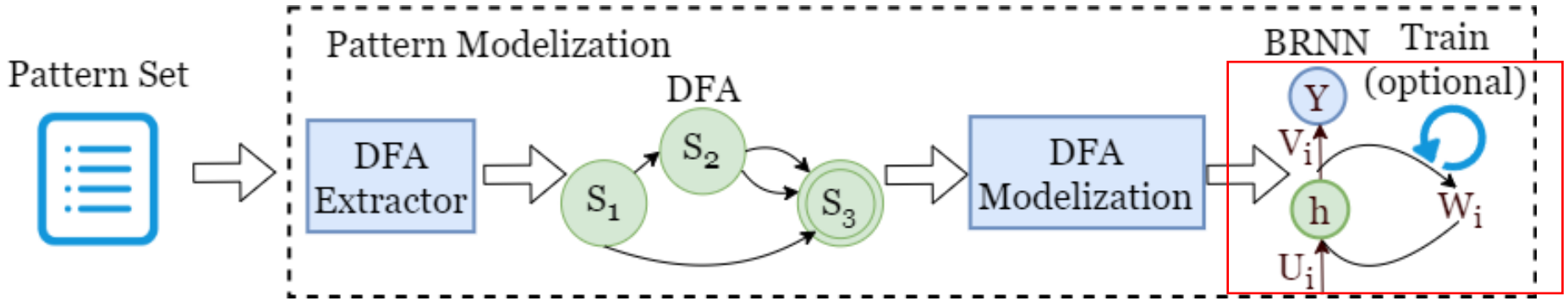
All paths

Parameters of a DFA

Initial and final state weight

Weight of state transfer

Pattern Modelization



DFA

Byte-level Recurrent Neural Networks (BRNN)

Input: $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$

Path: $\mathcal{P} = \{u_1, u_2, \ldots, u_{N+1}\}$

Score of forward algorithm:

$$\mathcal{B}_{\text{fw}}(\mathcal{A}, \mathcal{X}) = \sum_{\mathcal{P} \in \pi(x)} \mathcal{B}(\mathcal{A}, \mathcal{P}) = \alpha_0^T \cdot \left(\prod_{i=1}^{N} \mathcal{W}[x_i] \right) \cdot \alpha_\infty .$$

All paths

Parameters of a DFA

Initial and final state weight

Weight of state transfer

Rewrite the forward score into a recurrent form

$$h_0 = \alpha_0^T ,$$

① $$h_t = h_{t-1} \cdot \mathcal{W}[x_t], 1 \le t \le N,$$

$$\mathcal{B}_{\text{fw}}(\mathcal{A}, \mathcal{X}) = h_N \cdot \alpha_\infty .$$

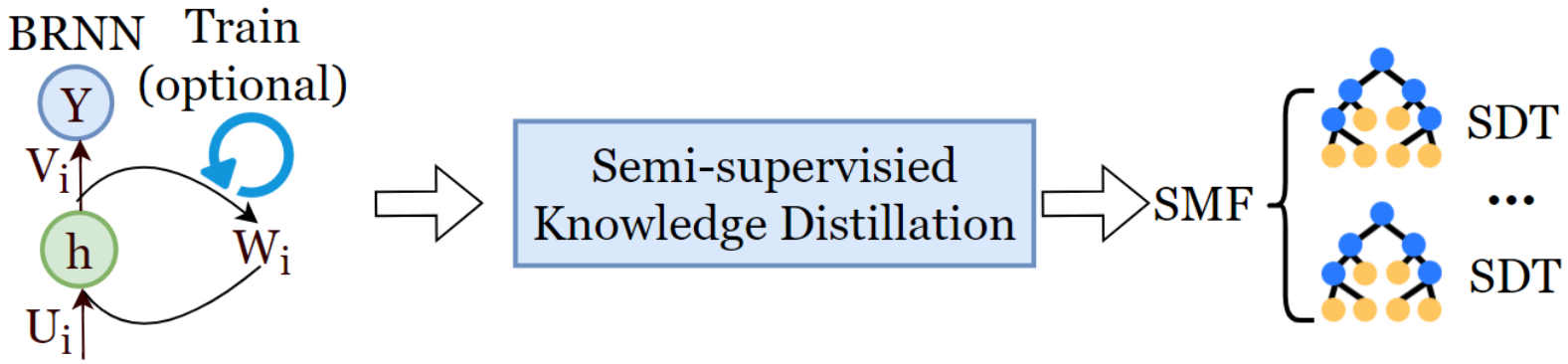
Forward propagation of RNN:

② $$h_t = \theta(Ux^t + Wh^{t-1} + b)$$

$$U = 0, b = 0, W = \mathcal{W}$$

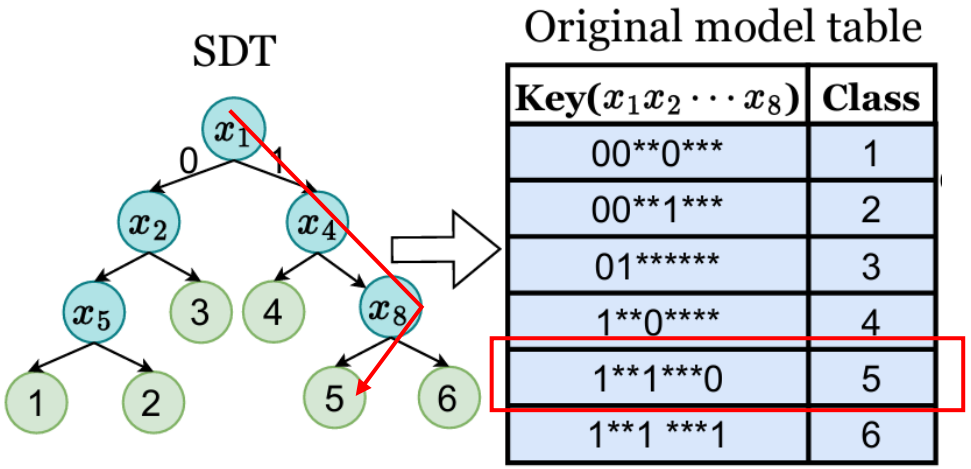
Data Plane Aware Model Adaption

- Deployment Challenge:
  - BRNNs are hard to deploy on the data plane due to floating-point and nonlinear operations.
- Our Solution: Semi-Supervised Knowledge Distillation (SSKD)
  - Transforms BRNN into Soft Multi-view Forest (SMF) — lightweight yet accurate.
  - Uses semi-supervised learning to train a high-accuracy student model in data-scarce settings.
  - SMF Inference: Requires only simple match-action operations, fully supported by switches.



Data Plane Model Representation

- Tree encoding
  - Convert tree classification to ternary matching



Original model table

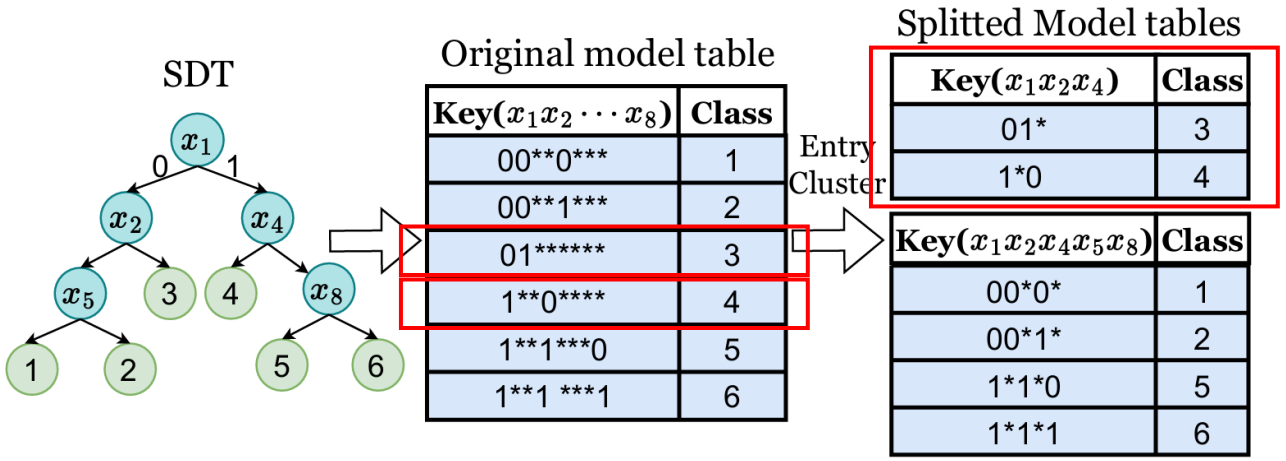
Key($x_1x_2\cdots x_8$)	Class
00**0***	1
00**1***	2
01******	3
1**0****	4
1**1***0	5
1**1 ***1	6

From the root to a leaf, only some bits are specified.

- Ternary Match:
  - Match if (Input & Mask) == Value
- Example For class 5
  - Mask: 0b10010001
  - Value: 0b10010000
  - Input: 0b11111110
  - Result
    - 0b11111110 & 0b10010001 = 0b10010000
    - Match

Data Plane Model Representation

- Entry cluster
  - Using heuristic algorithms to find near-optimal solutions



Example

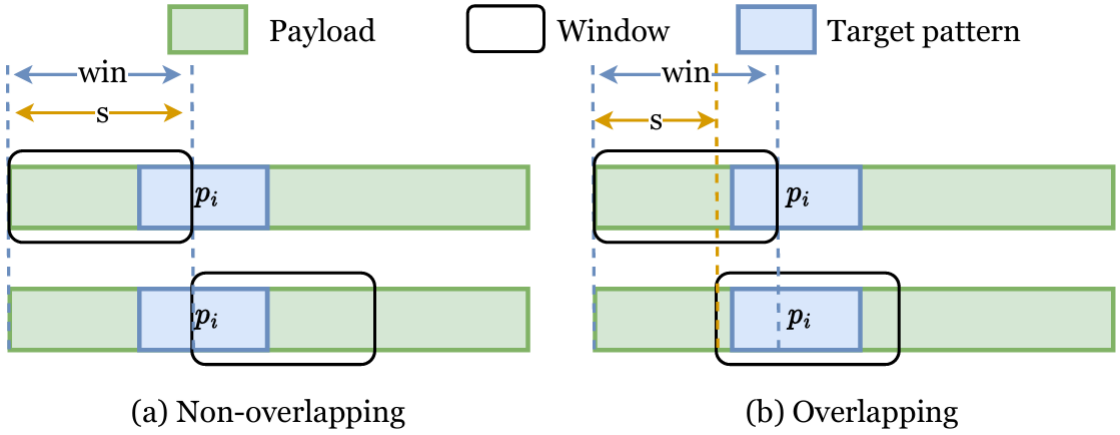
- For class 3 and 4, only $x_1x_2x_{14}$ are relevant, other bits can be disregarded.
- Reduces the matching key from 8 bits to 3 bits.

Table resource minimization problem

- Observation
  - The length of mask and value fields impacts storage resources consumed per table entry.
  - Most table entries contain masks with numerous zeros (wildcards), thus do not require full n-bit key matching
- Optimization Objective:
  - Partition the entire set E into k subsets to minimize storage footprint.

Data Plane Model Representation

- Sliding window
  - Challenges
    - Limited matching length supported by the ternary matching table
      - e.g., 66 bytes for Tofino 1
    - Maximum packet length is typically 1500 bytes
      - Payload may have to be matched across multiple windows



Design:

- Overlapping sliding window
- The matching window length minus the sliding step size must exceed the maximum pattern length
  - win – s > maximum pattern length

Evaluations

- Three questions

  ① Initial & Trainable Accuracy

  Does Trochilus achieve high baseline accuracy, with further improvement via training?

  ② Throughput-Storage Efficiency

  Can Trochilus deliver high throughput with minimal data plane storage?

  ③ Manageability

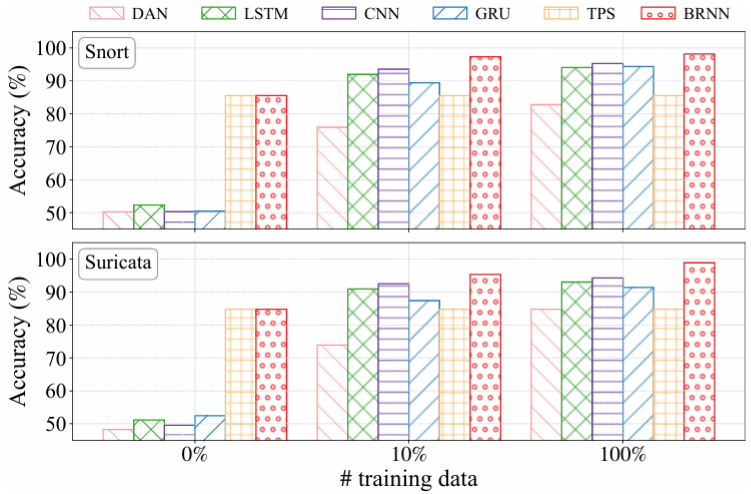
  Does Trochilus efficiently adapt to new traffic patterns?

Experimental Setup

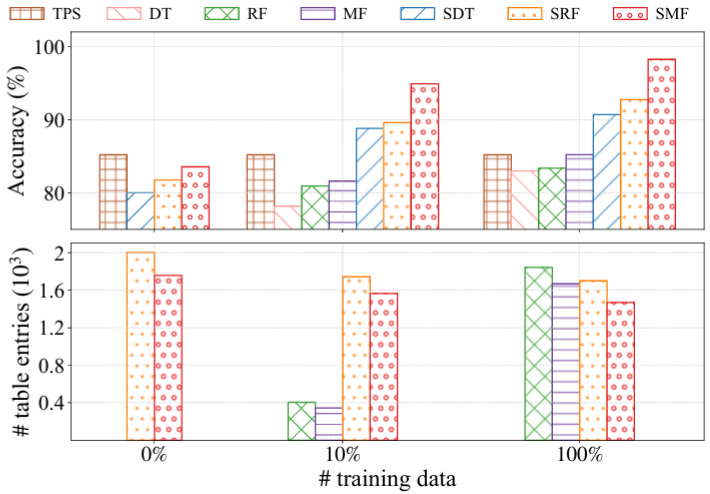
- **Dataset**
  - Pattern set: Snort 2.9.7.0 and Suricata 5.0
  - Packet level traces
    - From one of the largest public cloud providers
    - 10-minute traces sampled across different time periods over a month.
- **Baselines**
  - **TPS:** Traditional Pattern Matching systems (Snort & Suricata)
  - **BOLT:** Supports multi-string matching in the data plane (No existing solution supports full pattern matching in the data plane).
- **Metrics**
  - **Accuracy**
  - **Resource Usage:** Number of table entries and TCAM requirement

Evaluation of Accuracy

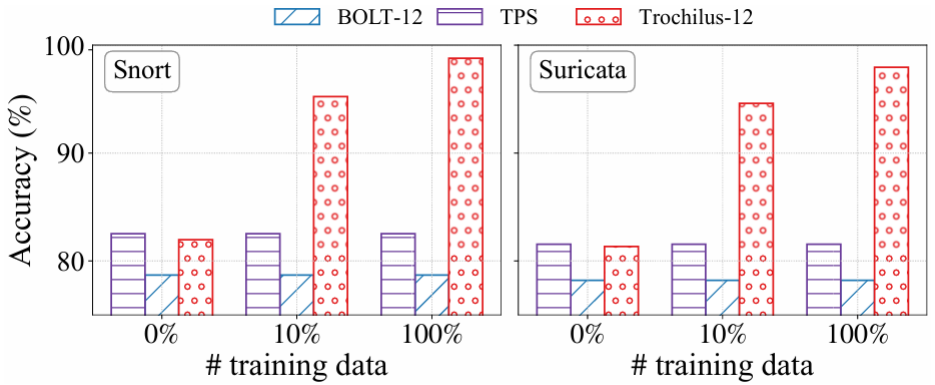
Teacher model accuracy
Pattern matching



Student model accuracy
Pattern matching

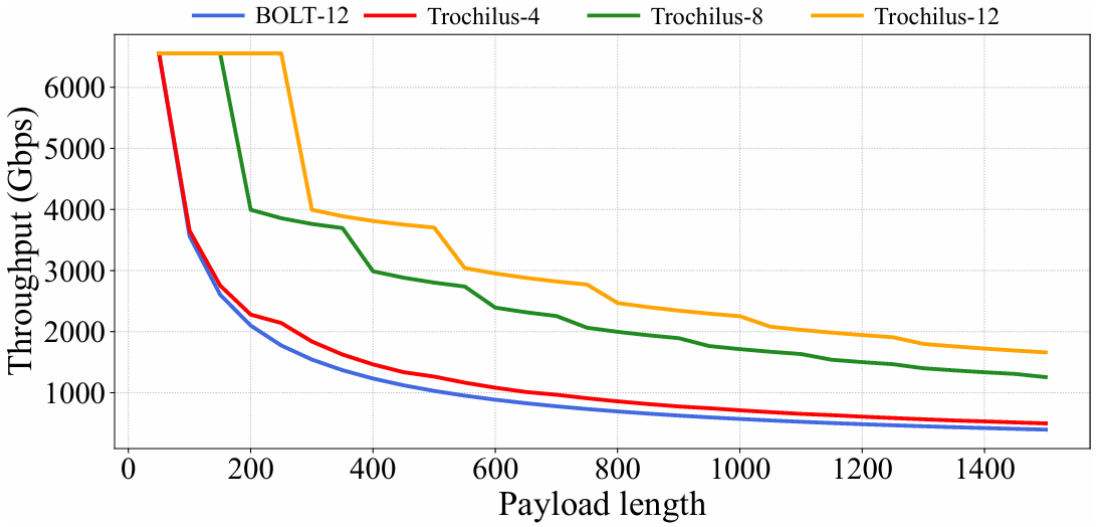


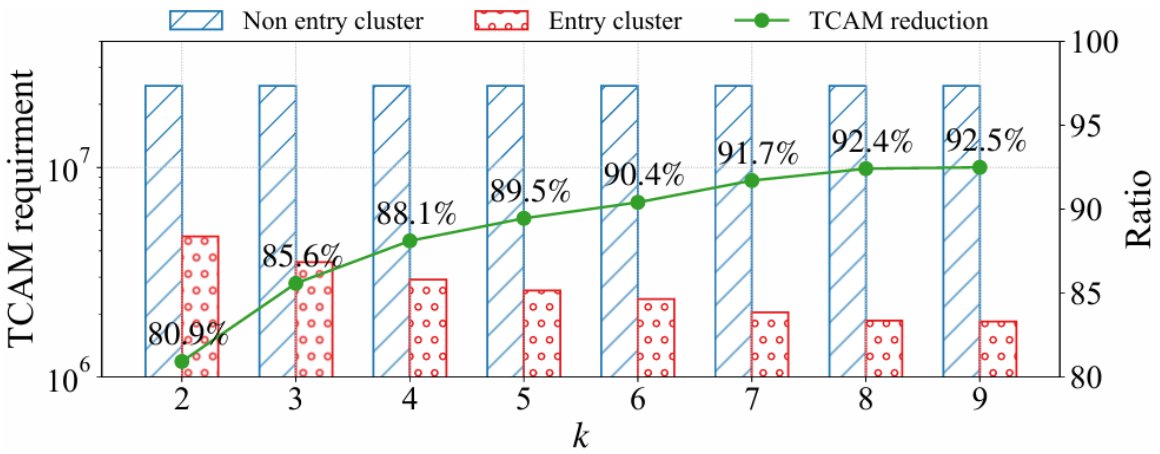
Compare with BOLT
Multi-string matching



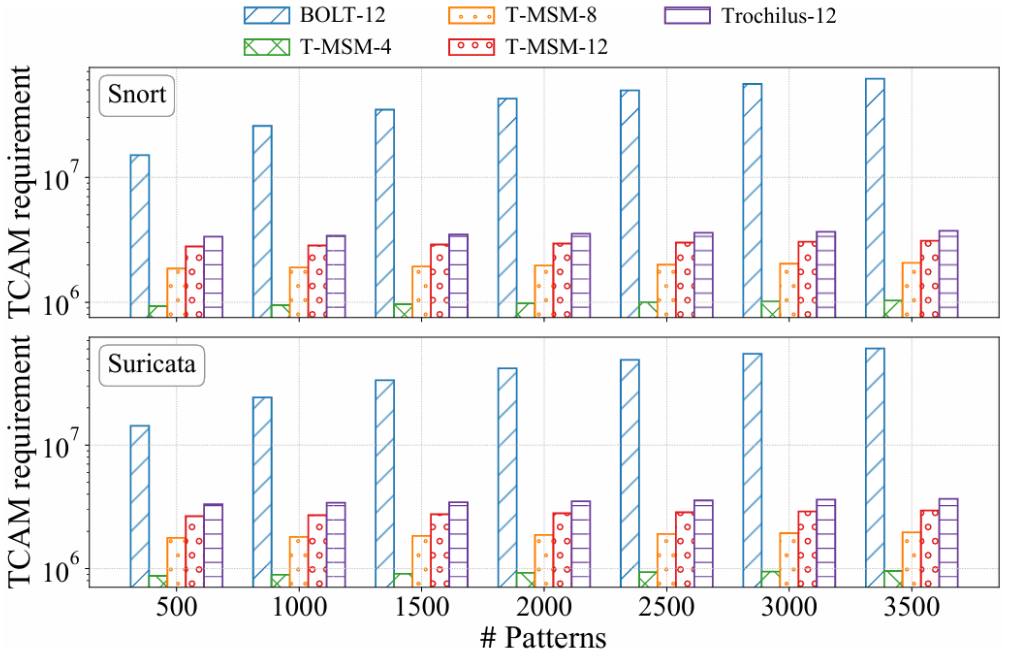
- High Initial Accuracy (No Training): Achieves comparable accuracy to traditional systems without labeled data.
- Enhanced Accuracy with Training: Accuracy improves by ~10% when using training data.
- Table Entries: Trochilus requires significantly fewer entries than traditional systems, reducing TCAM pressure while maintaining high accuracy.

Evaluation of Throughput and Storage

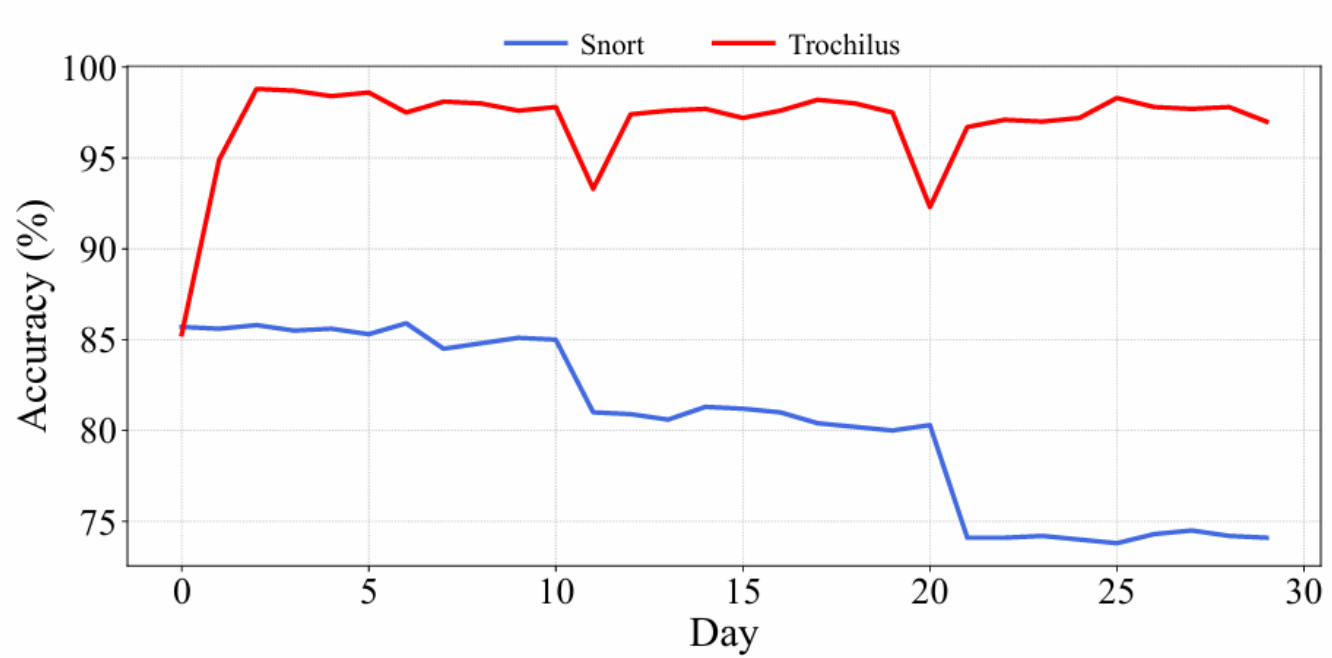




- Trochilus can achieve Tbps-level throughput for pattern matching while consuming minimal data plane storage resources.



Evaluation of Manageability



Comparison of accuracy over a 30-day experiment

- Trochilus keeps its accuracy high by automatically updating rules as traffic patterns change.

Conclusion

- Trochilus: A learning-enhanced, high-throughput pattern matching framework that:

1. Replaces traditional matching with model inference
2. Integrates ML learning ability with data-plane high-throughput and cost-effective advantages
3. Through innovative techniques like BRNN-based pattern modelization, semi-supervised knowledge distillation, and entry cluster optimization, Trochilus efficiently deploys high-performance models in resource-constrained data planes
4. Achieves multi-Tbps throughput while auto-updating for sustained accuracy

- Email: imbaplayer@163.com