New Directions In Automated Traffic Analysis

github.com/nprint/

Jordan Holland, Paul Schmitt, Nick Feamster, Prateek Mittal

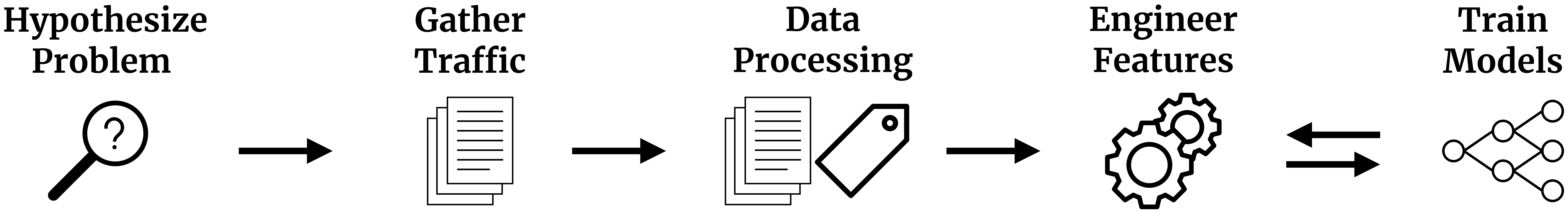
Network Traffic Analysis?

- Can we identify remote devices by probing them?
- How can we identify and stop attacks?
- Can we improve performance by analyzing traffic?
- Can users be tracked via their network traffic?

Network Traffic Analysis?

- Can we identify remote devices by probing them?
- How can we identify and stop attacks?
- Can we improve performance by analyzing traffic?
- Can users be tracked via their network traffic?
- Recently - Can machine learning techniques solve these problems?

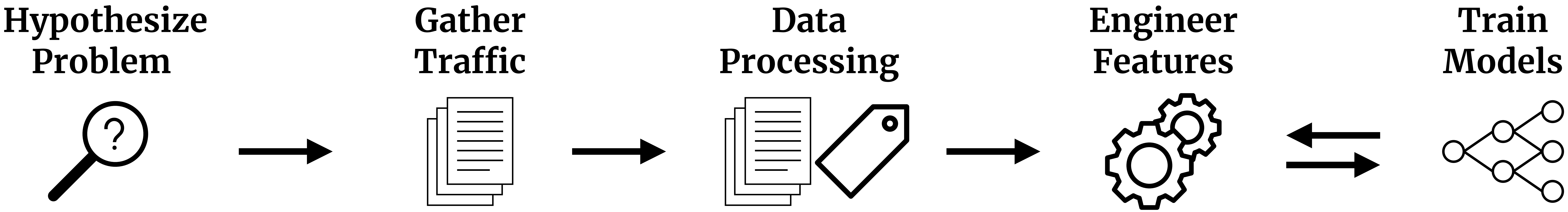
Classic ML Pipeline



Hypothesize Problem → Gather Traffic → Data Processing → Engineer Features ⇄ Train Models

Bespoke Solutions

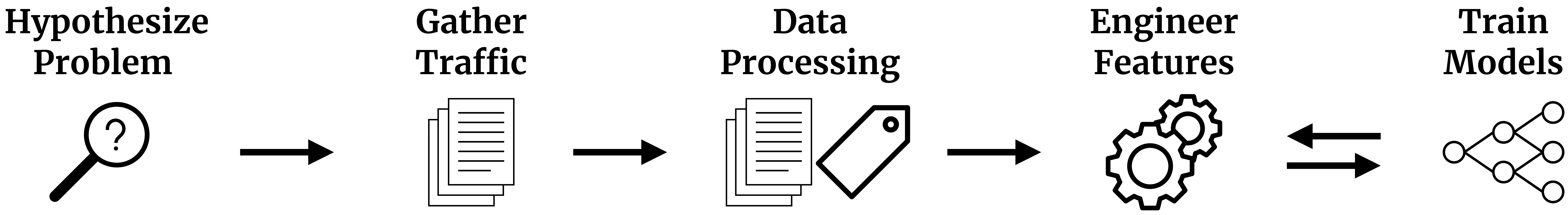
Application Identification



Hypothesize Problem → Gather Traffic → Data Processing → Engineer Features ⇄ Train Models

Bespoke Solutions

Application Identification



Hypothesize Problem

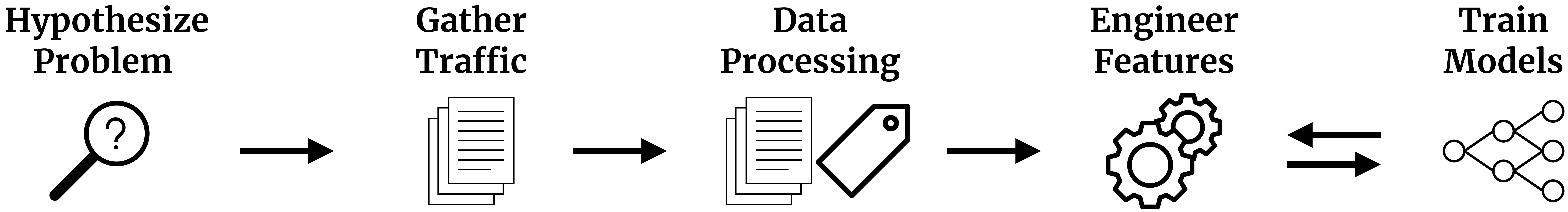
Gather Traffic

Data Processing

Engineer Features

Train Models

Anomaly Detection



Hypothesize Problem

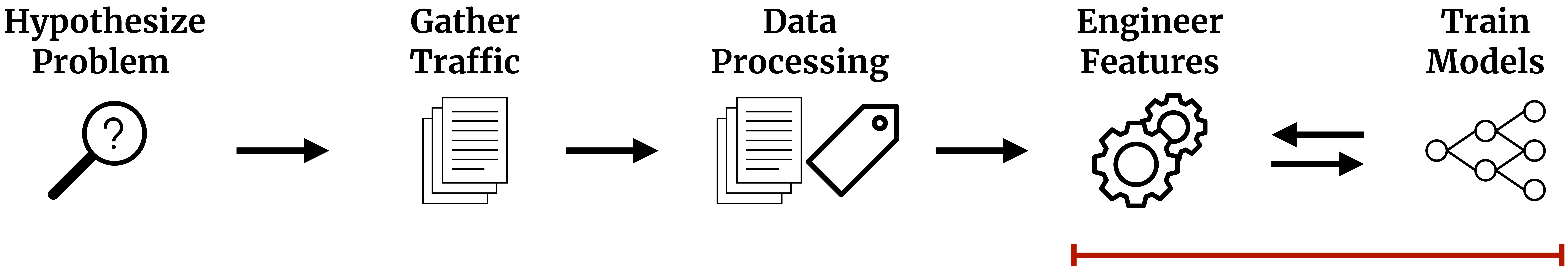
Gather Traffic

Data Processing

Engineer Features

Train Models

Generalizable Solutions?



Hypothesize Problem

Gather Traffic

Data Processing

Engineer Features

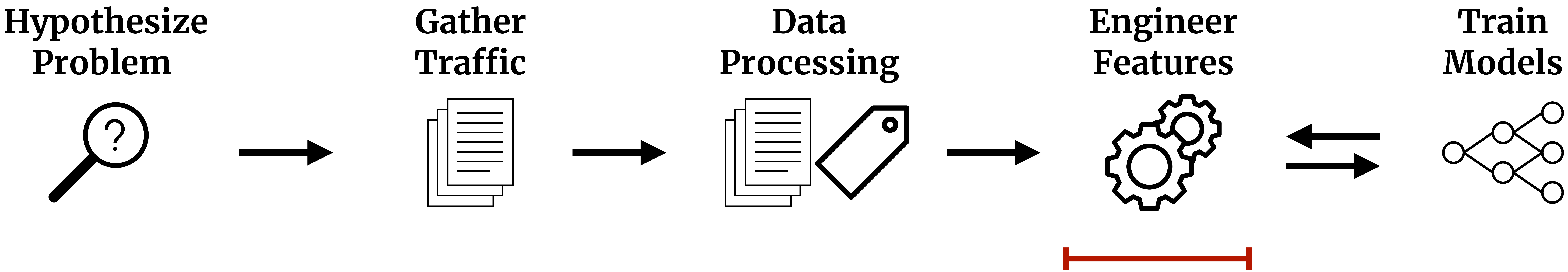
Train Models

Where We Are Headed

- Introduce nPrint, a generalizable packet representation that works across multiple traffic analysis tasks

- Combine nPrint and AutoML to create nPrintML, an open source system that generates full analysis pipelines

Classic ML Pipeline



Hypothesize Problem

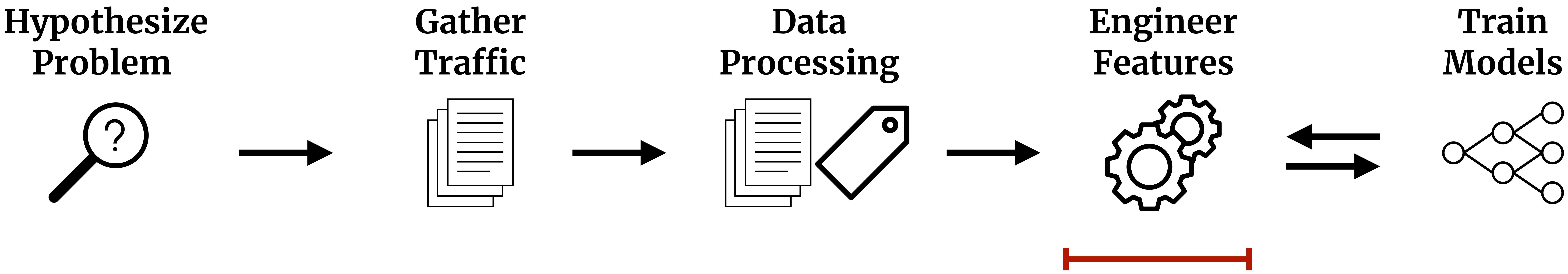
Gather Traffic

Data Processing

Engineer Features

Train Models

Features Are Expensive



Hypothesize Problem

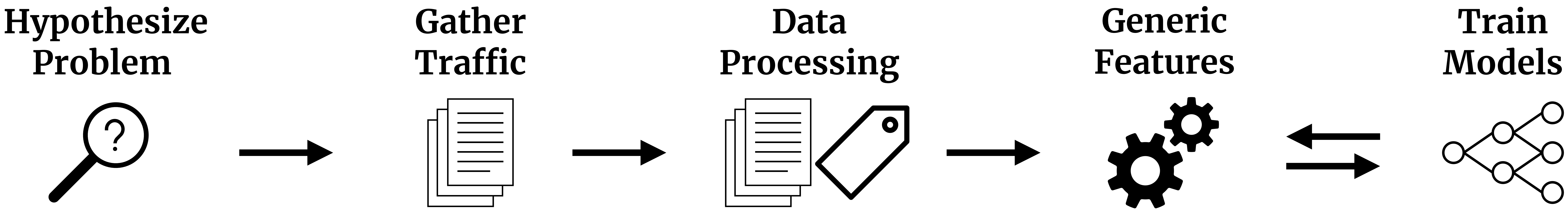
Gather Traffic

Data Processing

Engineer Features

Train Models

Are We Working Too Hard?



Hypothesize Problem

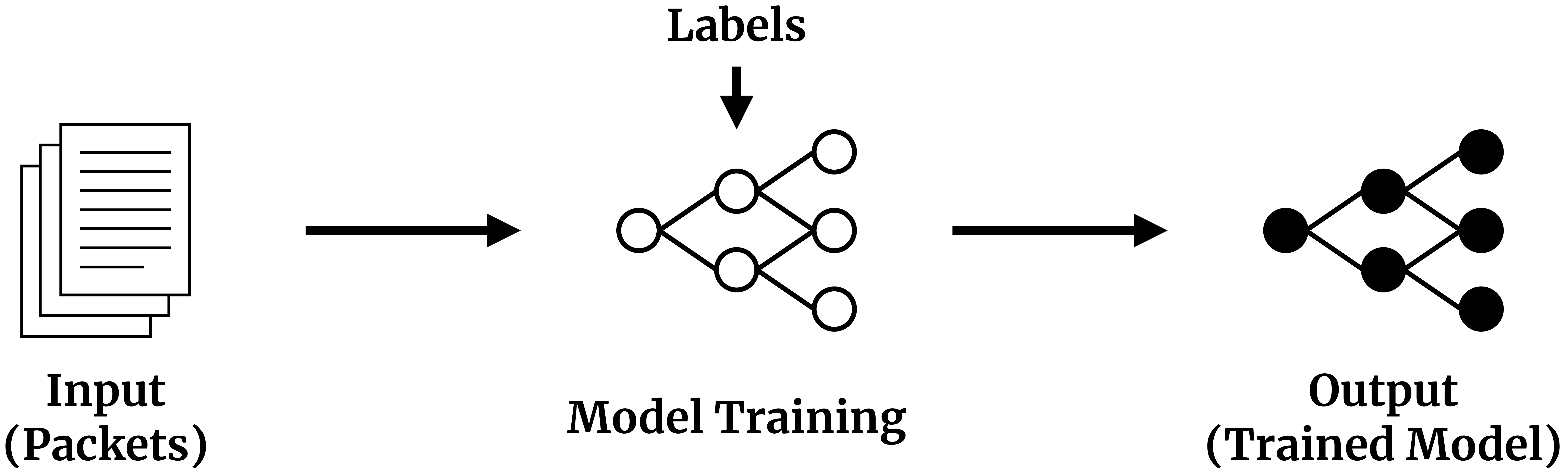
Gather Traffic

Data Processing

Generic Features

Train Models

Goal Pipeline



Labels

Input
(Packets)

Model Training

Output
(Trained Model)

Inspiration

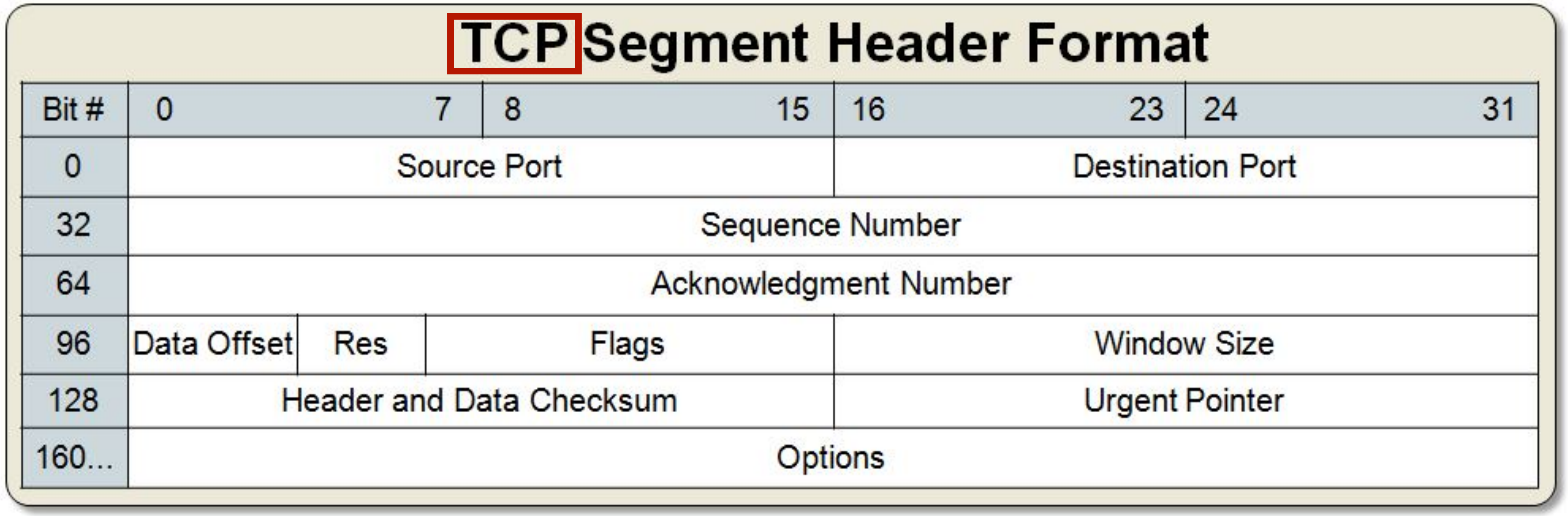
- Image recognition

- Website fingerprinting on Tor traffic [1,2]

Network Traffic Issues

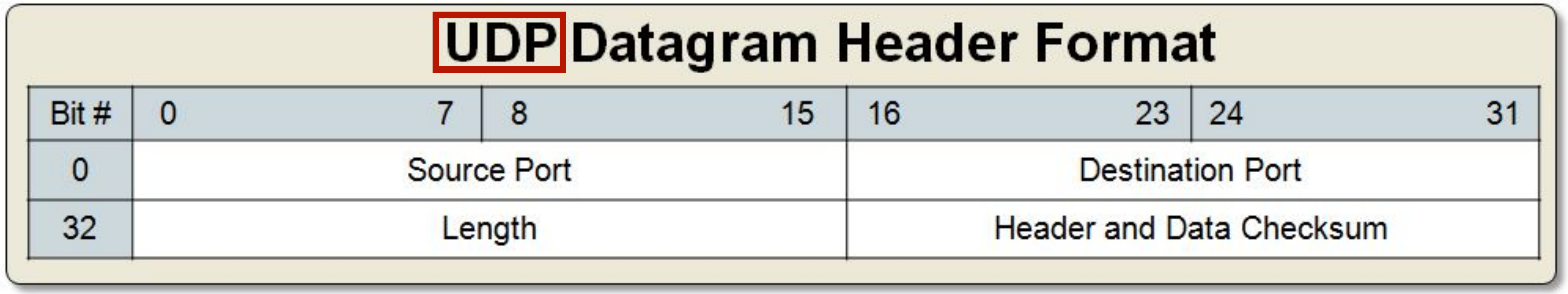
- Image recognition
- Website fingerprinting on Tor traffic [1,2]
- Problem - outside of Tor, network traffic not as simple!

Different Protocols



TCP Segment Header Format

Bit #	0	7	8	15	16	23	24	31
0	Source Port				Destination Port			
32	Sequence Number							
64	Acknowledgment Number							
96	Data Offset	Res	Flags		Window Size			
128	Header and Data Checksum				Urgent Pointer			
160...	Options							

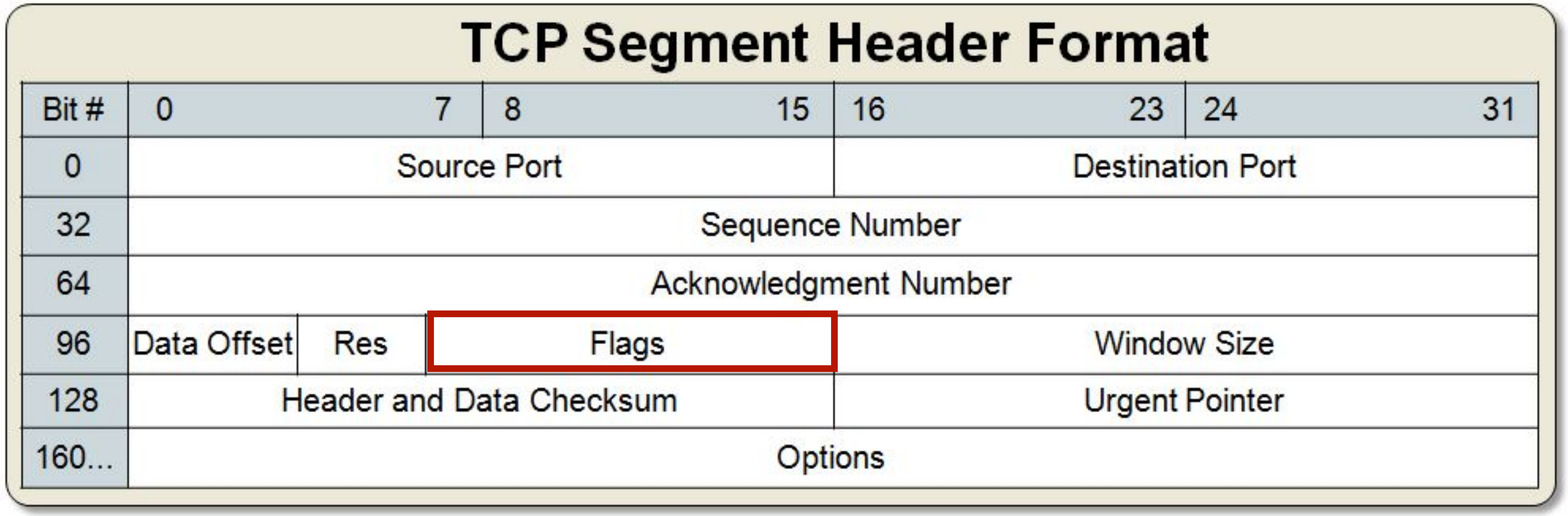


UDP Datagram Header Format

Bit #	0	7	8	15	16	23	24	31
0	Source Port				Destination Port			
32	Length				Header and Data Checksum			

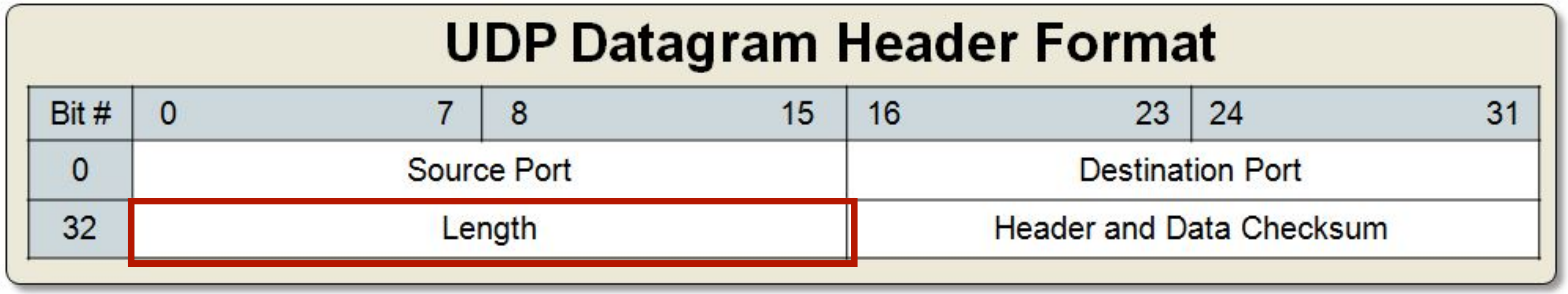
[3]

Different Values



TCP Segment Header Format

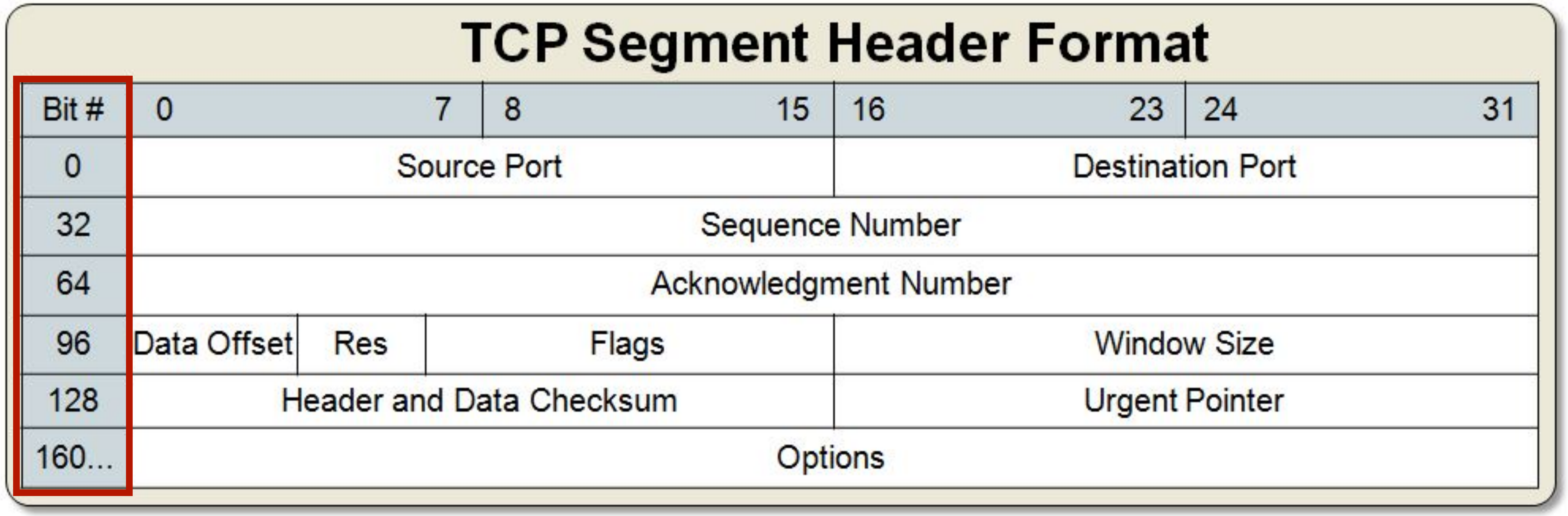
Bit #	0 7	8 15	16 23	24 31
0	Source Port		Destination Port	
32	Sequence Number			
64	Acknowledgment Number			
96	Data Offset	Res	Flags	Window Size
128	Header and Data Checksum		Urgent Pointer	
160...	Options			



UDP Datagram Header Format

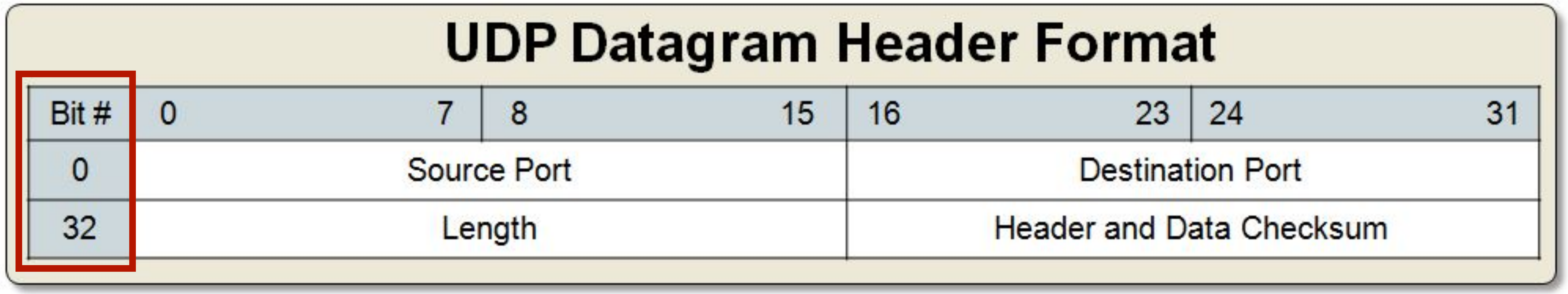
Bit #	0 7	8 15	16 23	24 31
0	Source Port		Destination Port	
32	Length		Header and Data Checksum	

Different Lengths



TCP Segment Header Format

<table>
<tr><th>Bit #</th><th colspan="3">0 7</th><th>8 15</th><th>16 23</th><th>24 31</th></tr>
<tr><td>0</td><td colspan="4">Source Port</td><td colspan="2">Destination Port</td></tr>
<tr><td>32</td><td colspan="6">Sequence Number</td></tr>
<tr><td>64</td><td colspan="6">Acknowledgment Number</td></tr>
<tr><td>96</td><td>Data Offset</td><td>Res</td><td colspan="2">Flags</td><td colspan="2">Window Size</td></tr>
<tr><td>128</td><td colspan="4">Header and Data Checksum</td><td colspan="2">Urgent Pointer</td></tr>
<tr><td>160...</td><td colspan="6">Options</td></tr>
</table>



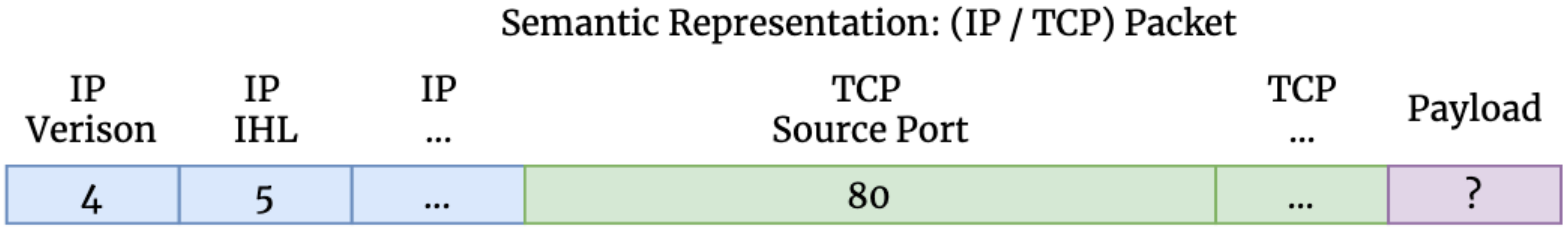
UDP Datagram Header Format

<table>
<tr><th>Bit #</th><th>0 7</th><th>8 15</th><th>16 23</th><th>24 31</th></tr>
<tr><td>0</td><td colspan="2">Source Port</td><td colspan="2">Destination Port</td></tr>
<tr><td>32</td><td colspan="2">Length</td><td colspan="2">Header and Data Checksum</td></tr>
</table>

Generic Packet Representation

The Semantic View

- Encode each header field as a feature



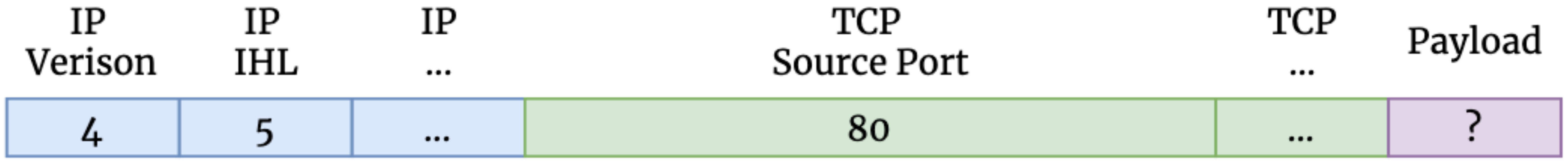
Semantic Representation: (IP / TCP) Packet

IP Verison	IP IHL	IP ...	TCP Source Port	TCP ...	Payload
4	5	...	80	...	?

Semantic Issues

- Semi-structured fields
- Domain expertise
- Normalization
- Payloads

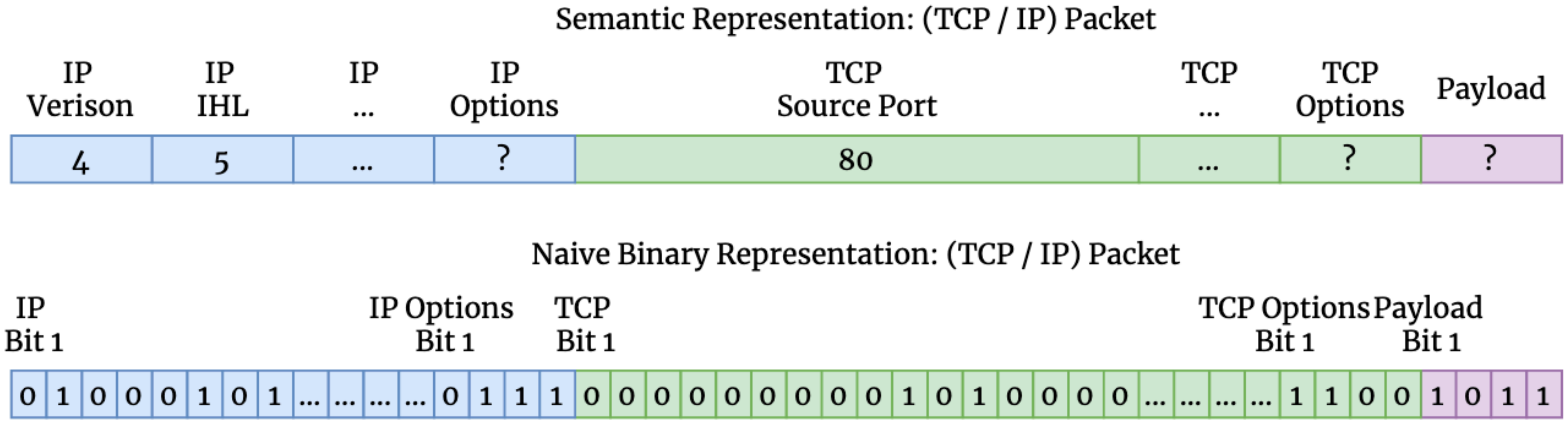
Semantic Representation: (IP / TCP) Packet



IP Verison	IP IHL	IP ...	TCP Source Port	TCP ...	Payload
4	5	...	80	...	?

The Binary View

- Insight: packets are a collection of bits



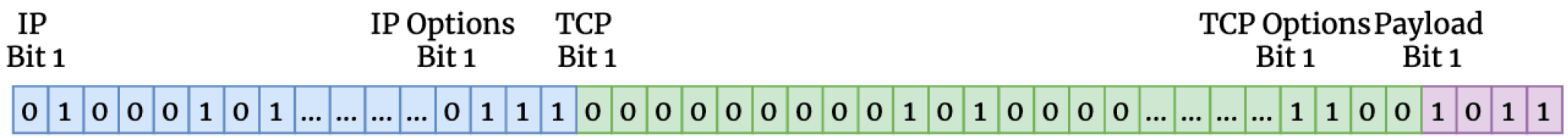
Semantic Representation: (TCP / IP) Packet

IP Verison	IP IHL	IP ...	IP Options	TCP Source Port	TCP ...	TCP Options	Payload
4	5	...	?	80	...	?	?

Naive Binary Representation: (TCP / IP) Packet

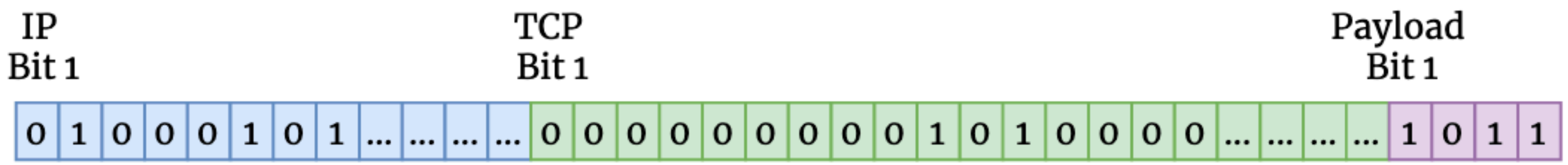
IP Bit 1												IP Options Bit 1				TCP Bit 1																				TCP Options Bit 1				Payload Bit 1			
0	1	0	0	0	1	0	1	...	...	...	...	0	1	1	1	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	...	...	...	...	1	1	0	0	1	0	1	1

Naive Noise



Naive Binary Representation: (TCP / IP) Packet

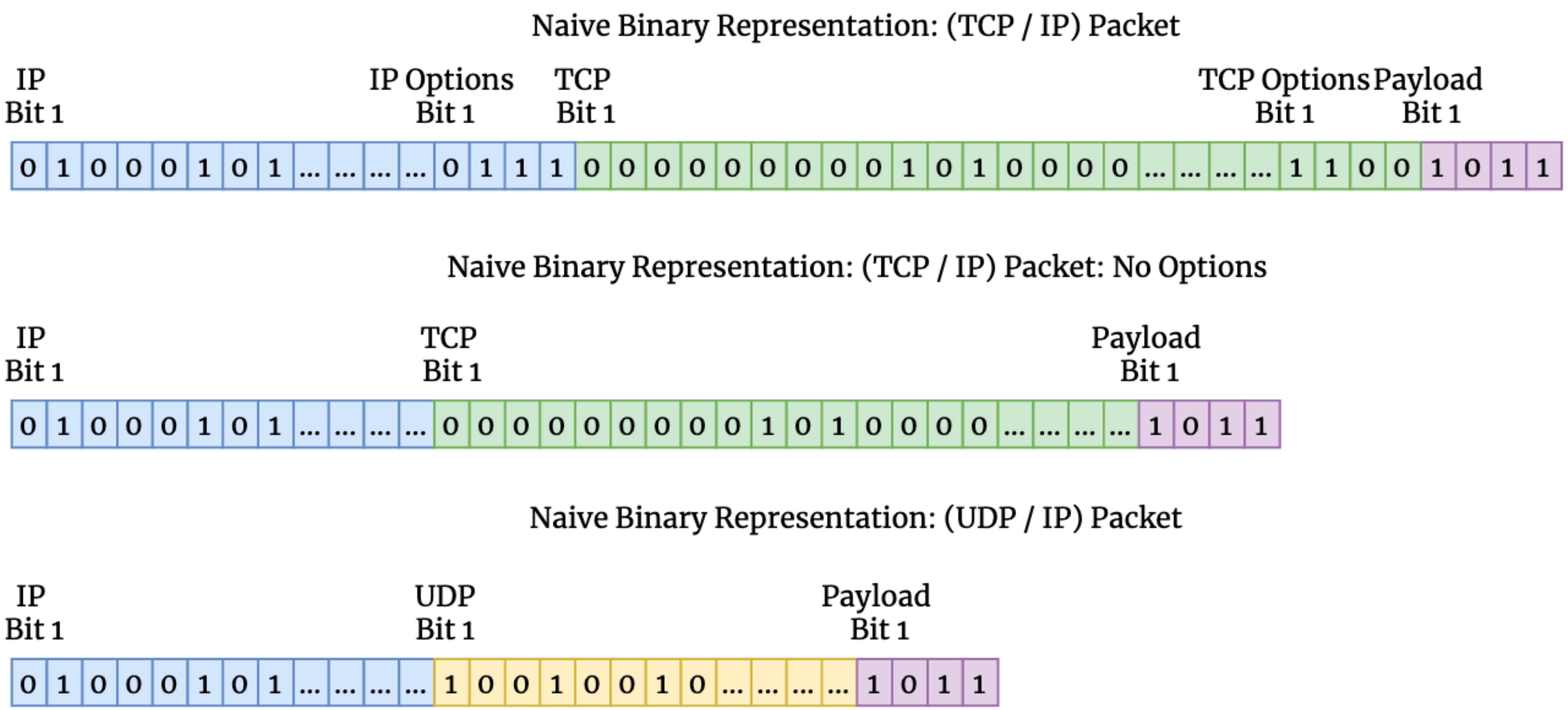
0 1 0 0 0 1 0 1 … … … … 0 1 1 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 … … … … 1 1 0 0 1 0 1 1



Naive Binary Representation: (TCP / IP) Packet: No Options

0 1 0 0 0 1 0 1 … … … … 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 … … … … 1 0 1 1

Naive Noise



Naive Binary Representation: (TCP / IP) Packet

IP Bit 1 | IP Options Bit 1 | TCP Bit 1 | TCP Options Bit 1 | Payload Bit 1

0 1 0 0 0 1 0 1 … … … … 0 1 1 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 … … … … 1 1 0 0 1 0 1 1

Naive Binary Representation: (TCP / IP) Packet: No Options

IP Bit 1 | TCP Bit 1 | Payload Bit 1

0 1 0 0 0 1 0 1 … … … … 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 … … … … 1 0 1 1

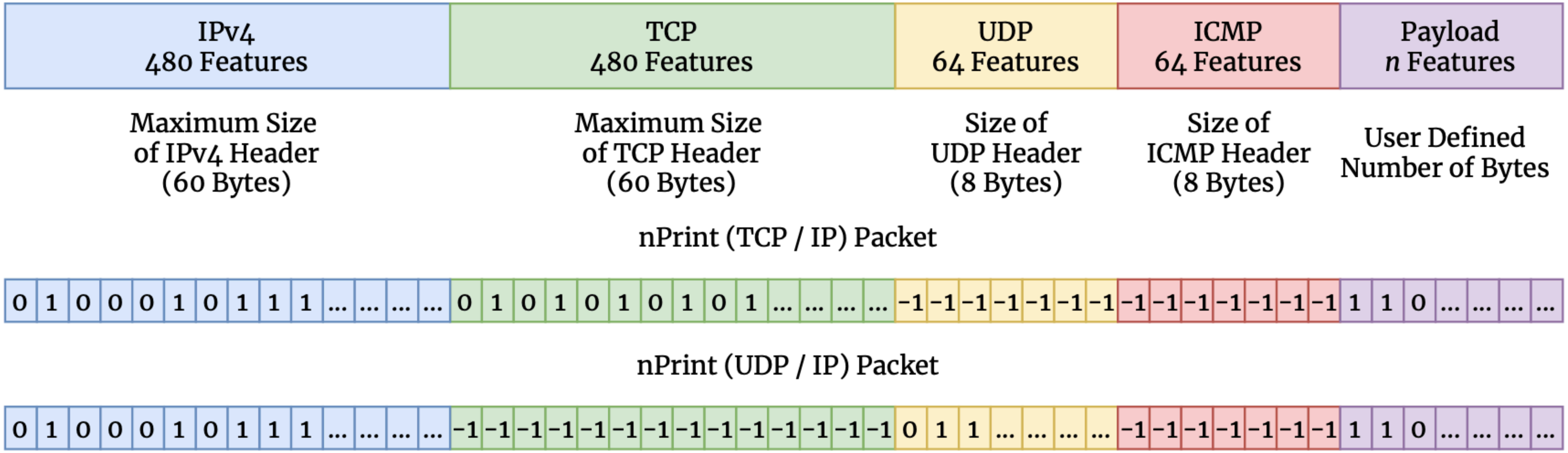
Naive Binary Representation: (UDP / IP) Packet

IP Bit 1 | UDP Bit 1 | Payload Bit 1

0 1 0 0 0 1 0 1 … … … … 1 0 0 1 0 0 1 0 … … … … 1 0 1 1

Packet
Representation

nPrint



IPv4 480 Features	TCP 480 Features	UDP 64 Features	ICMP 64 Features	Payload *n* Features
Maximum Size of IPv4 Header (60 Bytes)	Maximum Size of TCP Header (60 Bytes)	Size of UDP Header (8 Bytes)	Size of ICMP Header (8 Bytes)	User Defined Number of Bytes

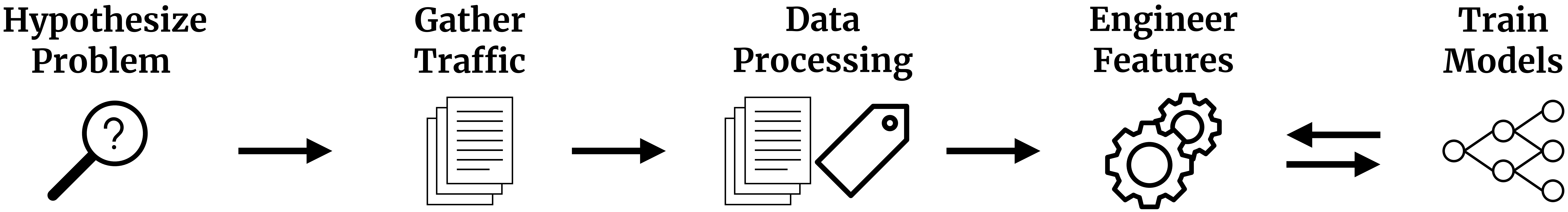
nPrint (TCP / IP) Packet

0 1 0 0 0 1 0 1 1 1 ... ... 0 1 0 1 0 1 0 1 0 1 ... ... -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 0 ... ...

nPrint (UDP / IP) Packet

0 1 0 0 0 1 0 1 1 1 ... ... -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 0 1 1 ... ... -1 -1 -1 -1 -1 -1 -1 1 1 0 ... ...

Classic ML Pipeline



Hypothesize Problem

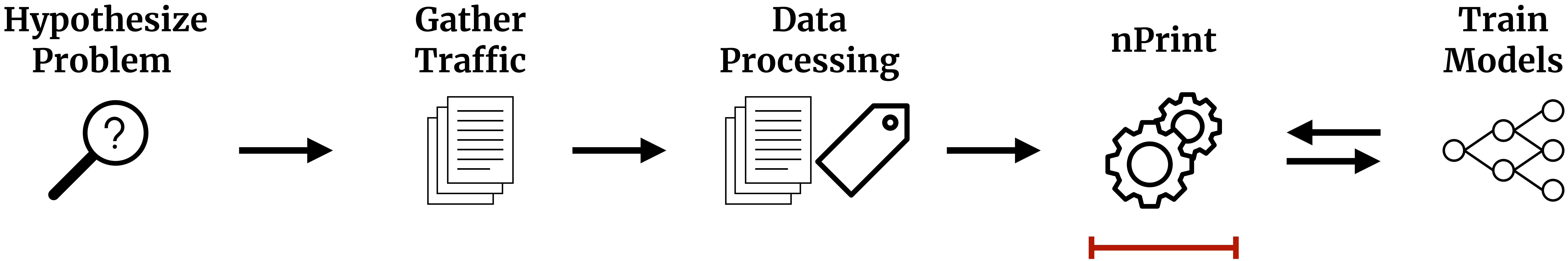
Gather Traffic

Data Processing

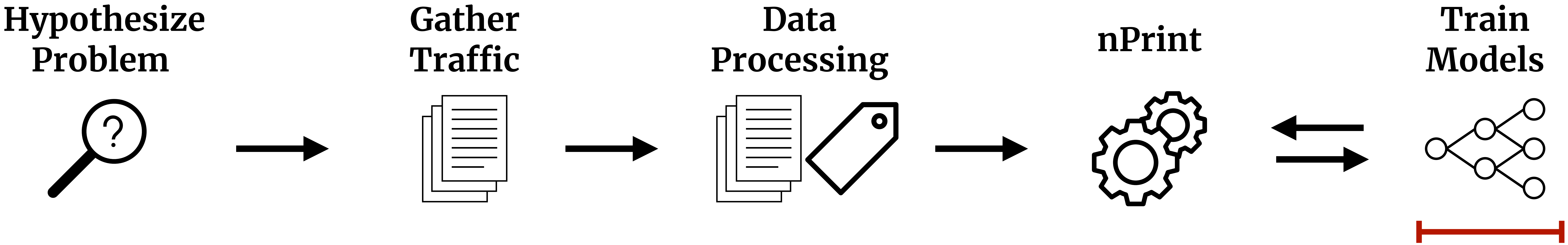
Engineer Features

Train Models

nPrint Replaces Feature Engineering



New Bottleneck



Hypothesize Problem

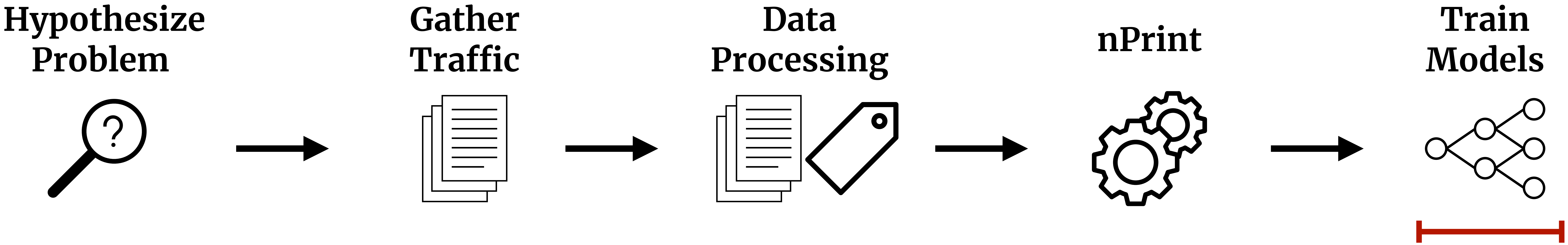
Gather Traffic

Data Processing

nPrint

Train Models

Can We Automate This Step?



Classic Model Training

- Pick favorite model(s)

- Search some hyper-parameters for that model

- Choose best model

Automated Machine Learning

- Model selection

- Feature selection

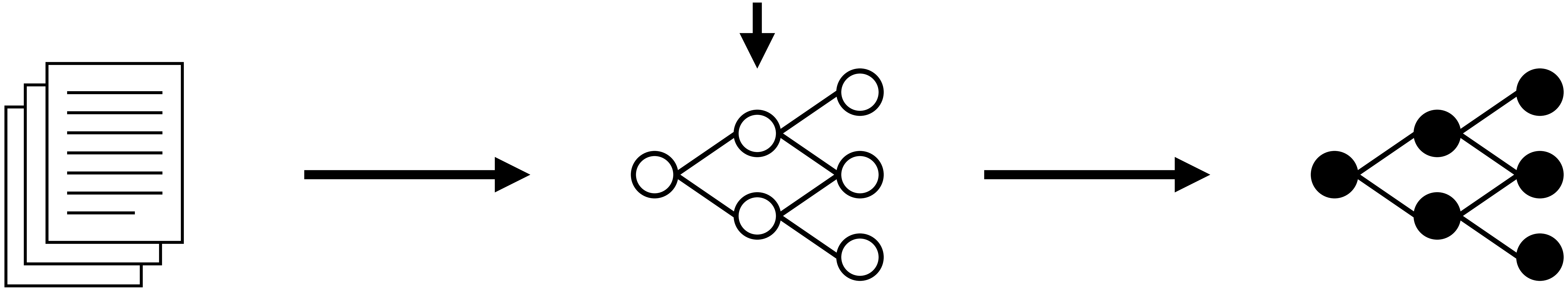
- Hyperparameter search

AutoGluon AutoML

- Model ensembling achieves higher performance than other AutoML techniques[8]

- Train models from 7 base classes
  - Random forests
  - DNN
  - KNN

Original Goal Pipeline



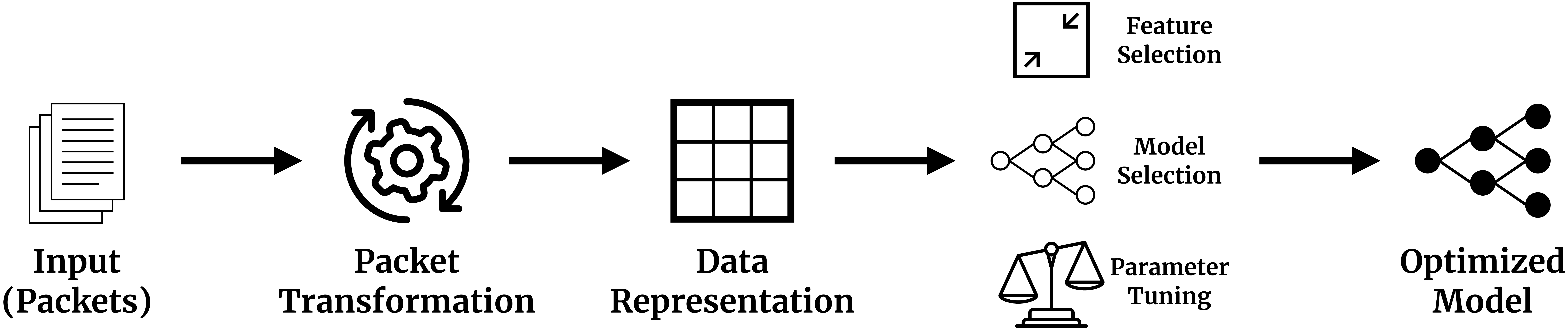
Labels

Input
(Packets)

Model Training

Output
(Trained Model)

Detailed Traffic Analysis



Input
(Packets)

Packet
Transformation

Data
Representation

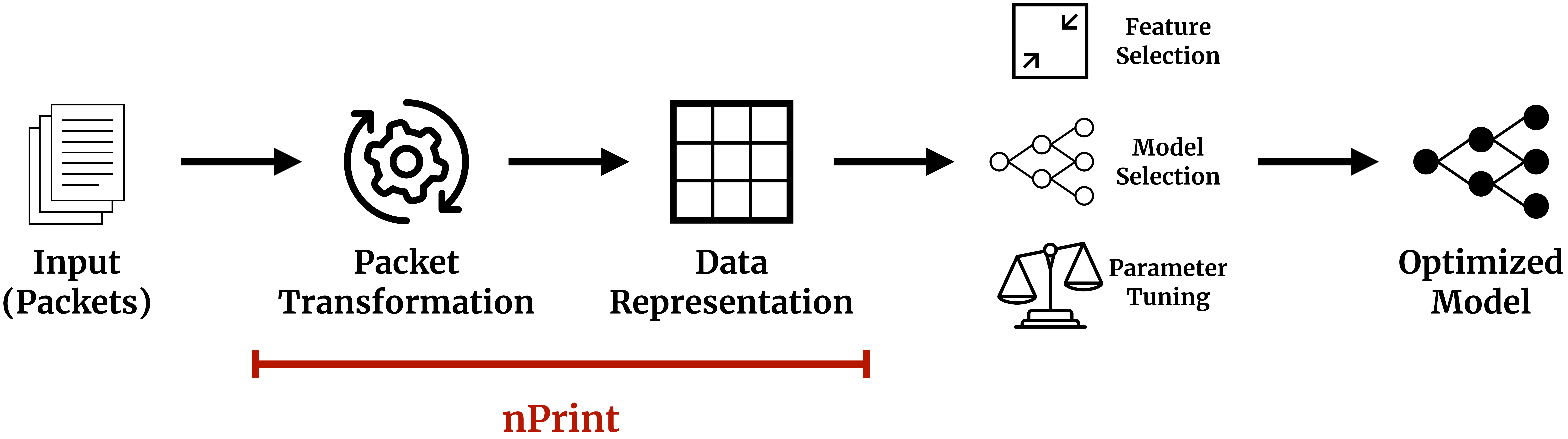
Feature
Selection

Model
Selection

Parameter
Tuning

Optimized
Model

nPrint Transforms And Represents Packets



Input (Packets)

Packet Transformation

Data Representation

Feature Selection

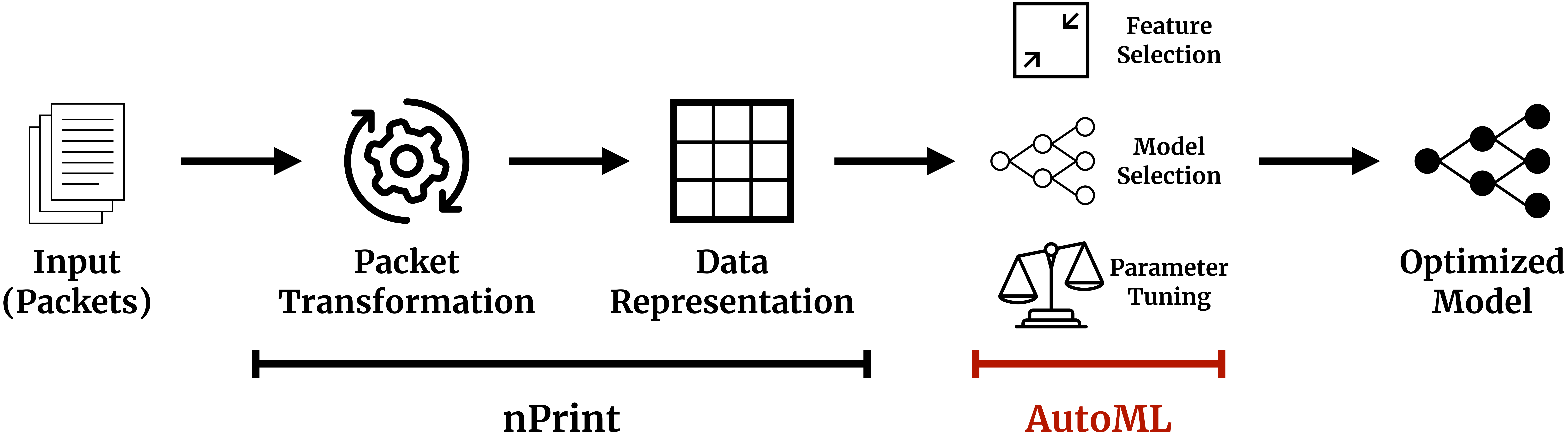
Model Selection

Parameter Tuning

Optimized Model

nPrint

AutoML Finds The Best Model



Input (Packets)

Packet Transformation

Data Representation

Feature Selection

Model Selection

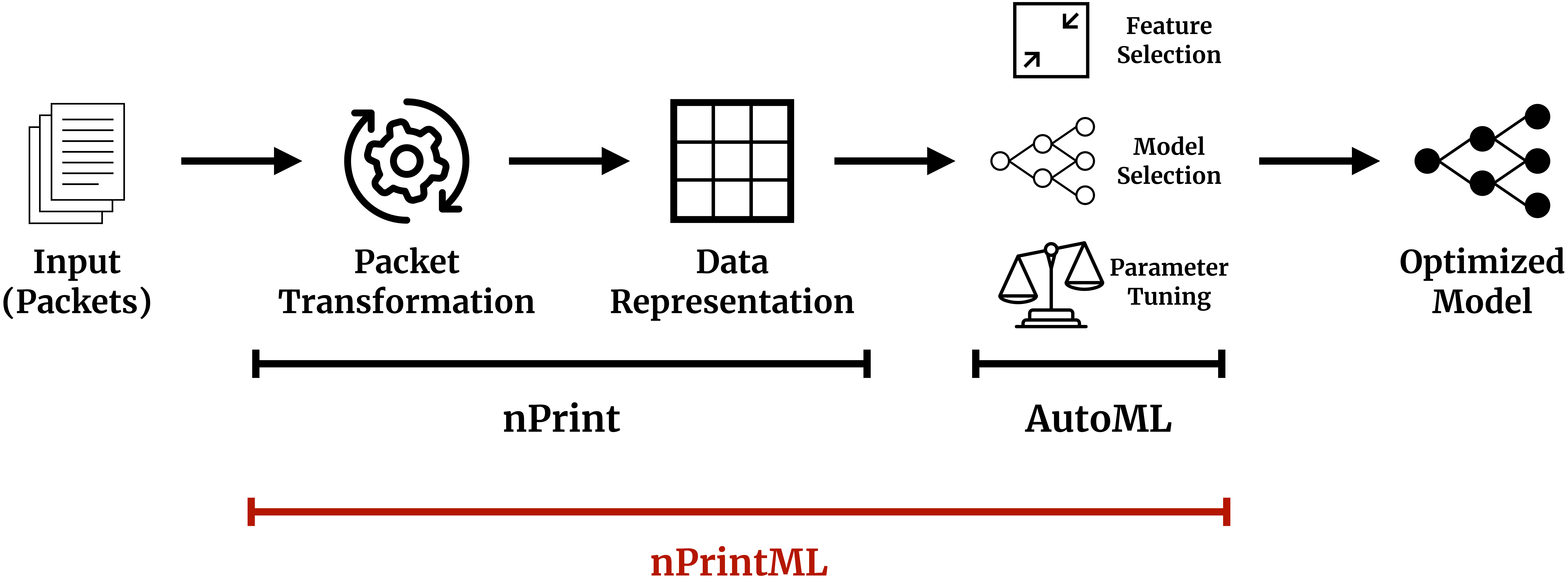
Parameter Tuning

Optimized Model

nPrint

AutoML

nPrintML Combines Both!



Input (Packets)

Packet Transformation

Data Representation

Feature Selection

Model Selection

Parameter Tuning

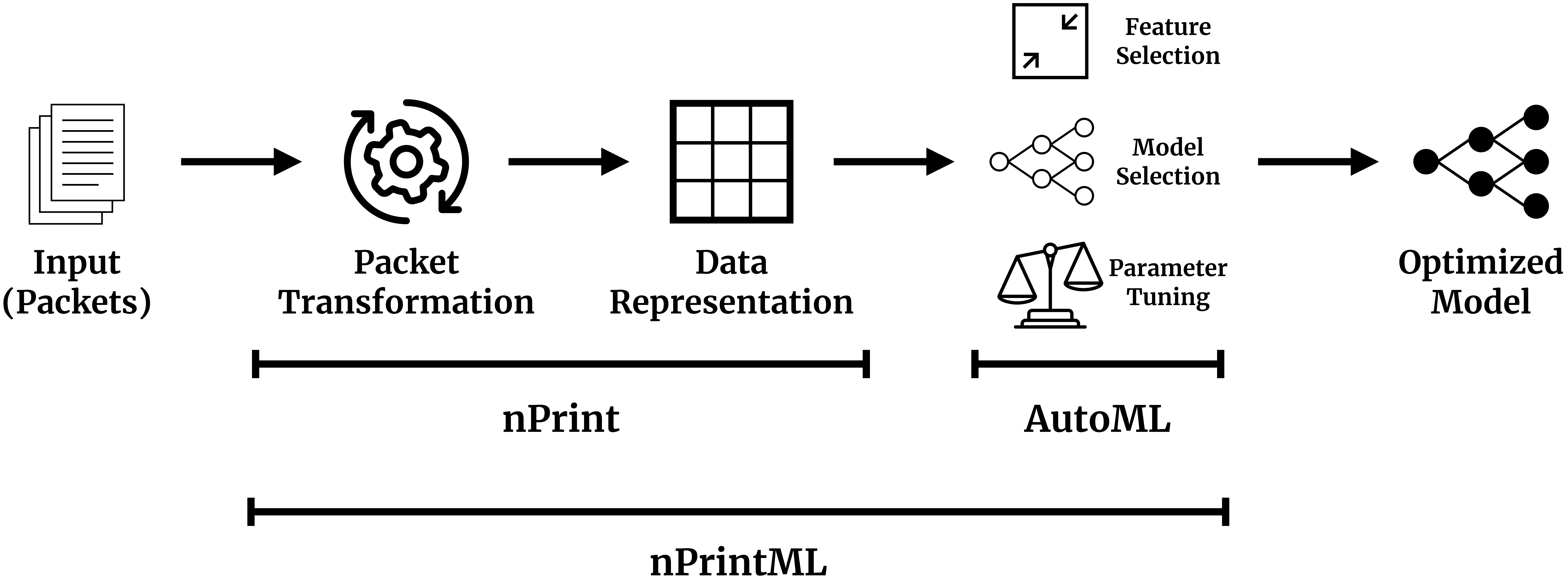
Optimized Model

nPrint

AutoML

nPrintML

Let’s Try it!



Input (Packets)

Packet Transformation

Data Representation

Feature Selection

Model Selection

Parameter Tuning

Optimized Model

nPrint

AutoML

nPrintML

Defining The Problem

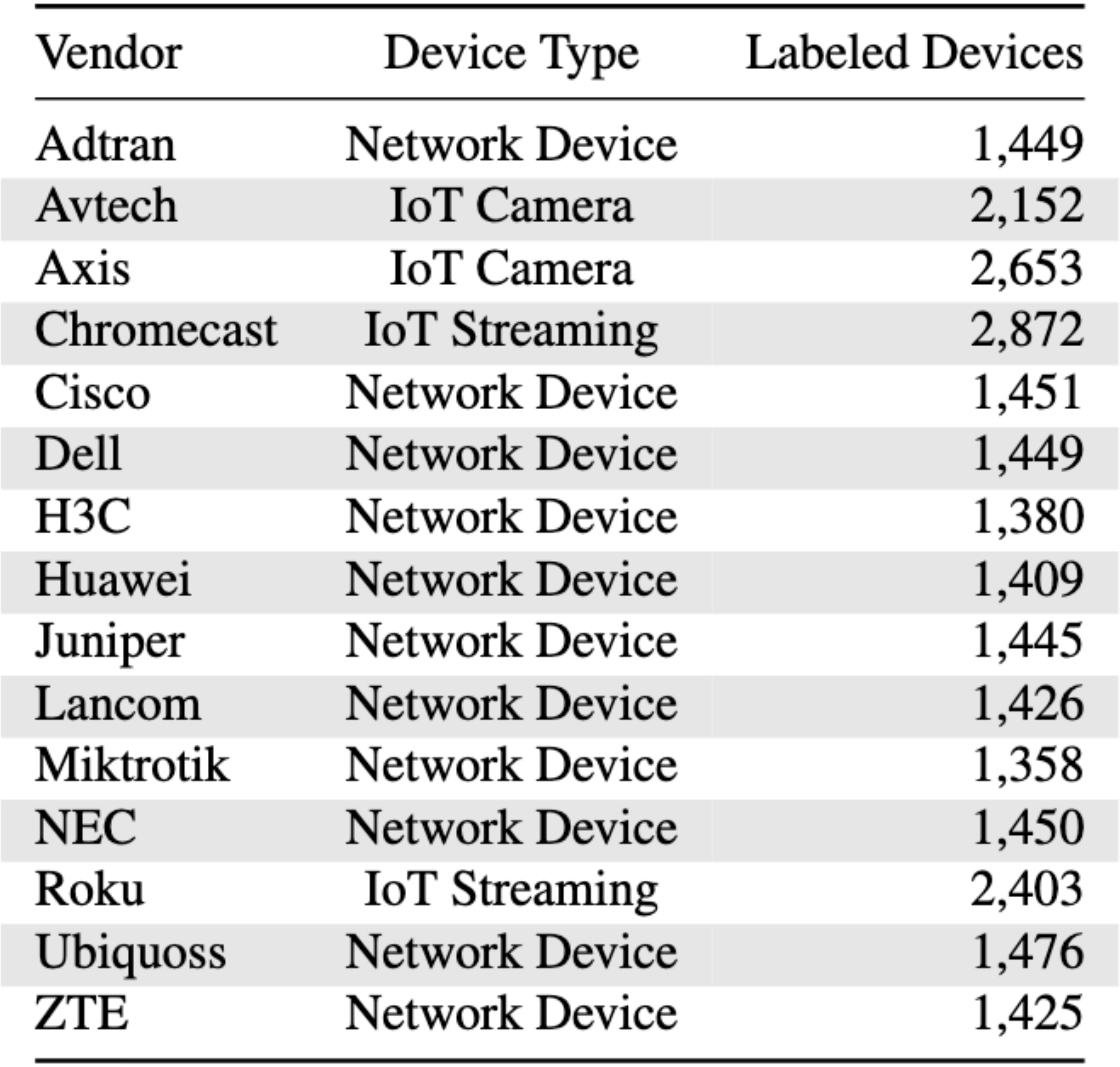
- Remote device fingerprinting

Building A Dataset

- Remote device fingerprinting

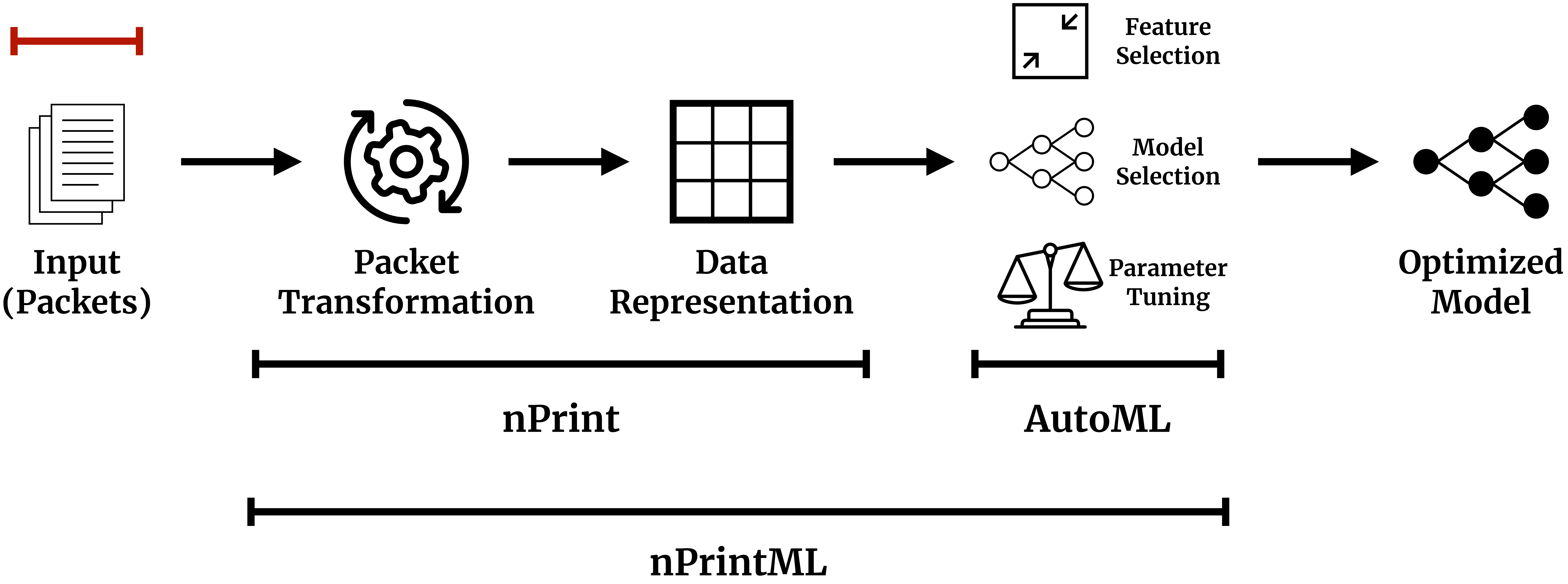
- Labeled Targets
  - Routers[4]
  - IoT Devices (Shodan)

Labeled Dataset

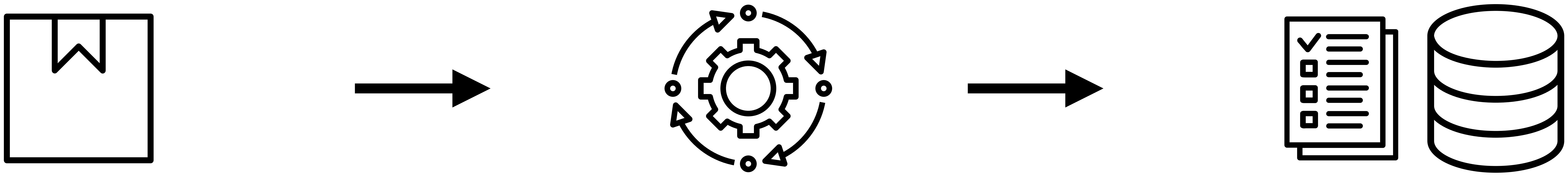


Vendor	Device Type	Labeled Devices
Adtran	Network Device	1,449
Avtech	IoT Camera	2,152
Axis	IoT Camera	2,653
Chromecast	IoT Streaming	2,872
Cisco	Network Device	1,451
Dell	Network Device	1,449
H3C	Network Device	1,380
Huawei	Network Device	1,409
Juniper	Network Device	1,445
Lancom	Network Device	1,426
Miktrotik	Network Device	1,358
NEC	Network Device	1,450
Roku	IoT Streaming	2,403
Ubiquoss	Network Device	1,476
ZTE	Network Device	1,425

Gathering Traffic



Leveraging Nmap

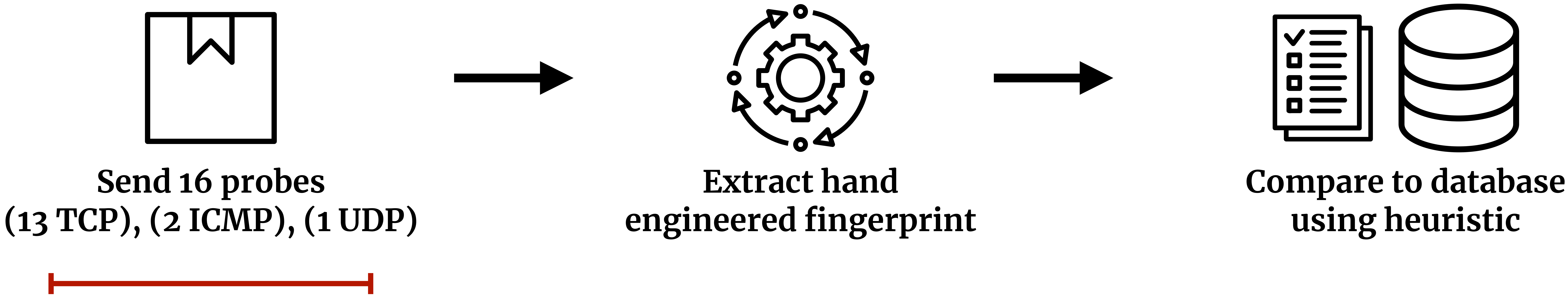


Send 16 probes
(13 TCP), (2 ICMP), (1 UDP)

Extract hand
engineered fingerprint

Compare to database
using heuristic

Leveraging Nmap

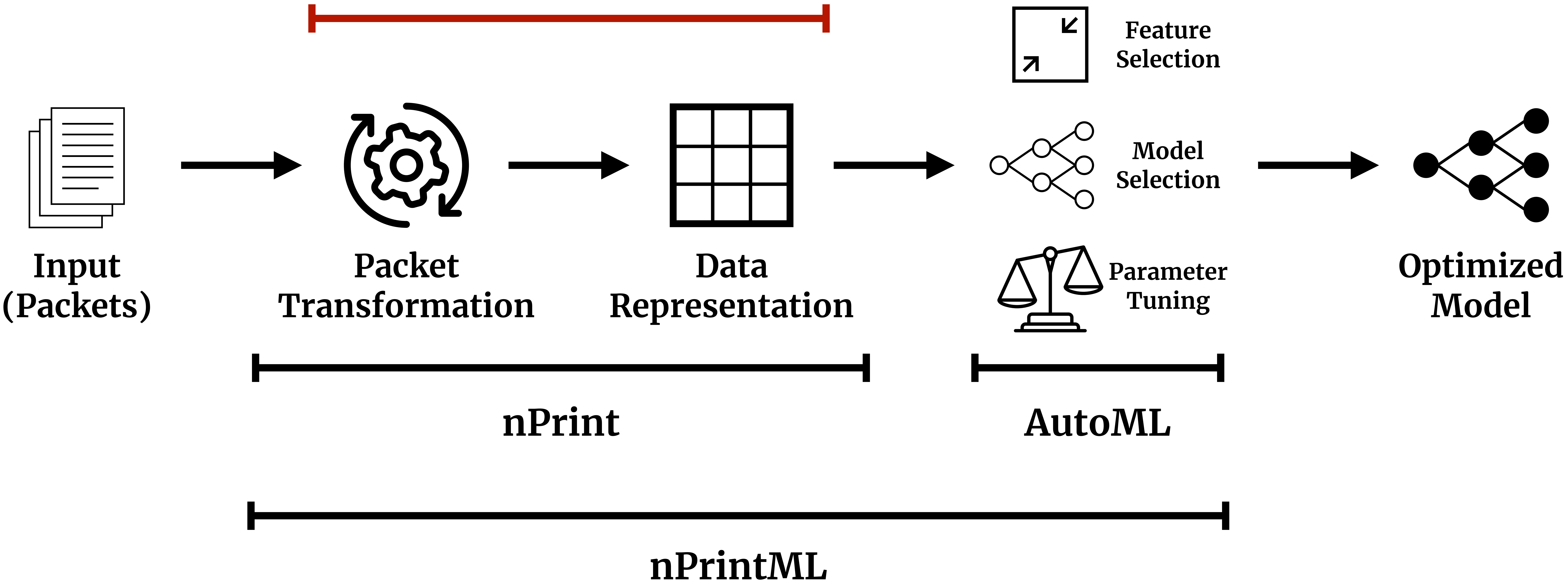


Send 16 probes
(13 TCP), (2 ICMP), (1 UDP)

Extract hand
engineered fingerprint

Compare to database
using heuristic

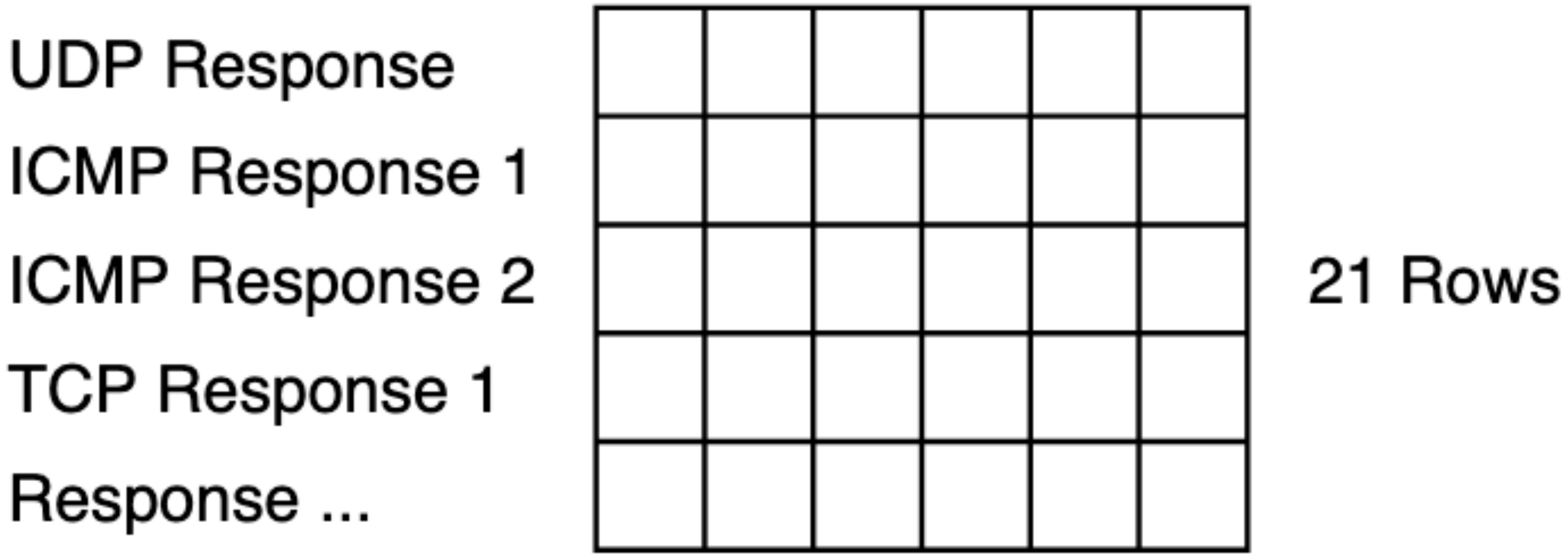
Transforming Packets



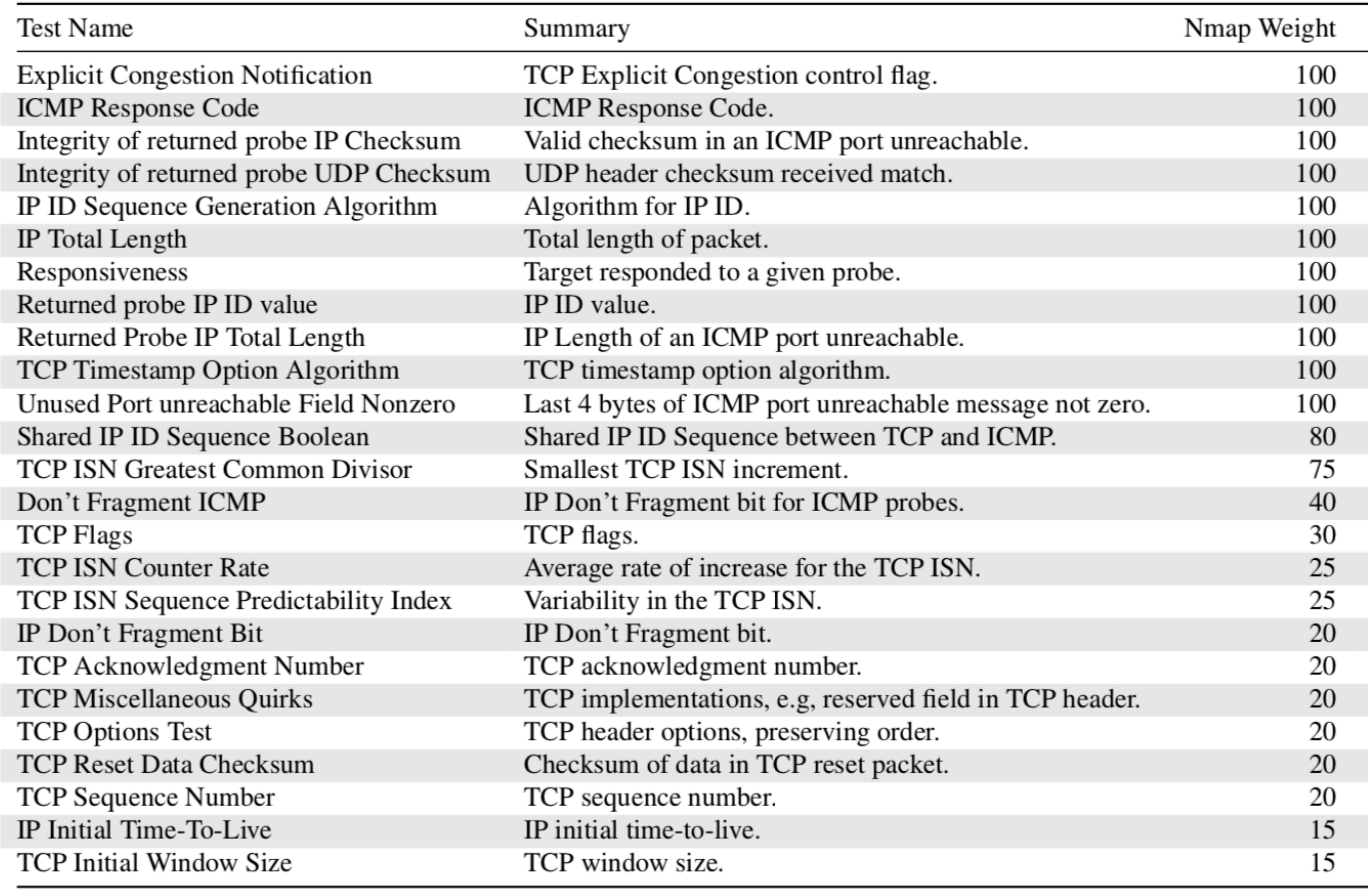
nPrint Packet Transformation

- 21 uniquely named responses

- Sort responses by name and concatenate individual nPrints

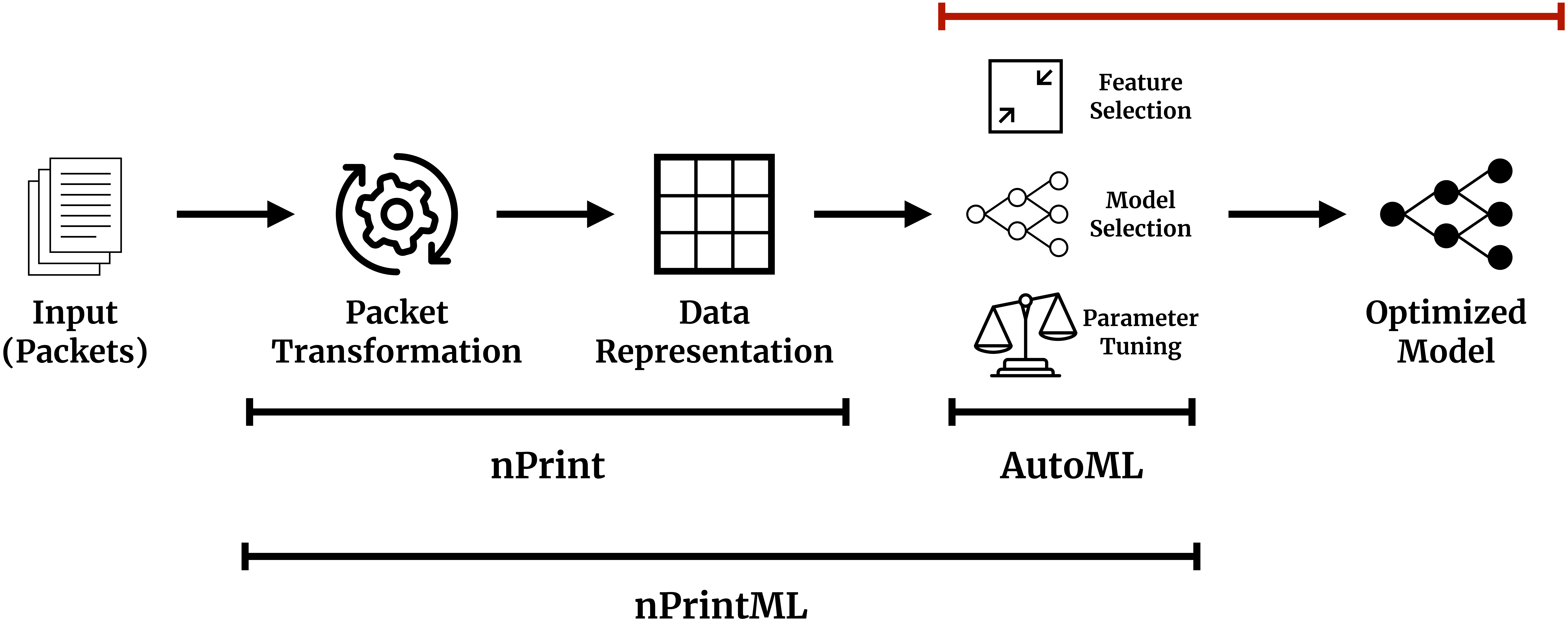


Nmap Packet Transformation

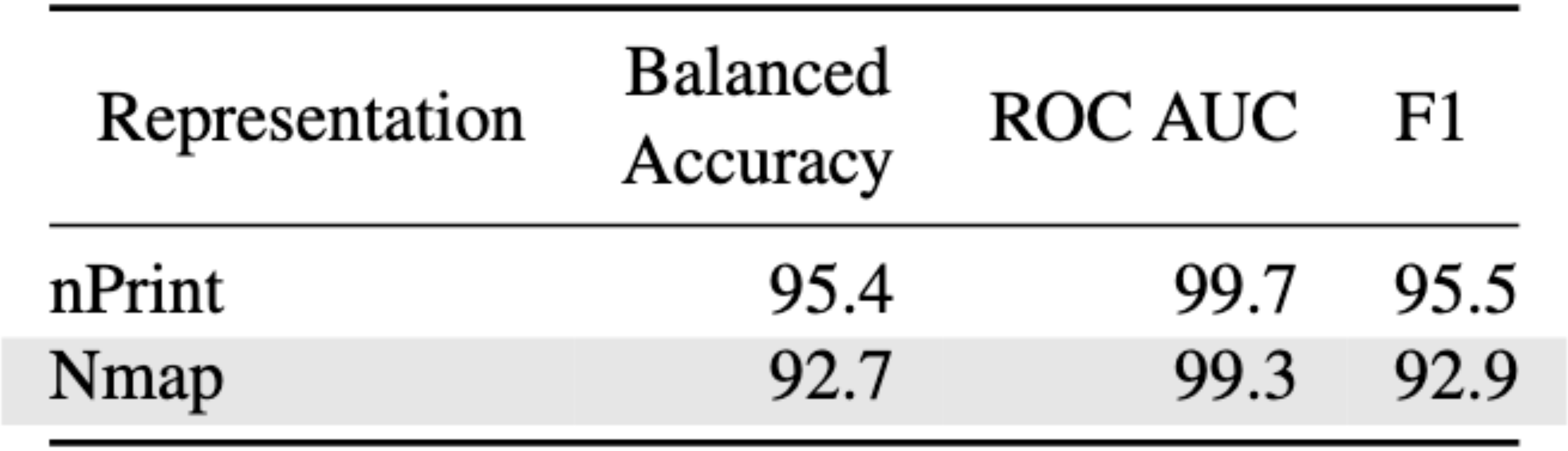


Test Name	Summary	Nmap Weight
Explicit Congestion Notification	TCP Explicit Congestion control flag.	100
ICMP Response Code	ICMP Response Code.	100
Integrity of returned probe IP Checksum	Valid checksum in an ICMP port unreachable.	100
Integrity of returned probe UDP Checksum	UDP header checksum received match.	100
IP ID Sequence Generation Algorithm	Algorithm for IP ID.	100
IP Total Length	Total length of packet.	100
Responsiveness	Target responded to a given probe.	100
Returned probe IP ID value	IP ID value.	100
Returned Probe IP Total Length	IP Length of an ICMP port unreachable.	100
TCP Timestamp Option Algorithm	TCP timestamp option algorithm.	100
Unused Port unreachable Field Nonzero	Last 4 bytes of ICMP port unreachable message not zero.	100
Shared IP ID Sequence Boolean	Shared IP ID Sequence between TCP and ICMP.	80
TCP ISN Greatest Common Divisor	Smallest TCP ISN increment.	75
Don’t Fragment ICMP	IP Don’t Fragment bit for ICMP probes.	40
TCP Flags	TCP flags.	30
TCP ISN Counter Rate	Average rate of increase for the TCP ISN.	25
TCP ISN Sequence Predictability Index	Variability in the TCP ISN.	25
IP Don’t Fragment Bit	IP Don’t Fragment bit.	20
TCP Acknowledgment Number	TCP acknowledgment number.	20
TCP Miscellaneous Quirks	TCP implementations, e.g, reserved field in TCP header.	20
TCP Options Test	TCP header options, preserving order.	20
TCP Reset Data Checksum	Checksum of data in TCP reset packet.	20
TCP Sequence Number	TCP sequence number.	20
IP Initial Time-To-Live	IP initial time-to-live.	15
TCP Initial Window Size	TCP window size.	15

Training Models



nPrint Outperforms Nmap



Representation	Balanced Accuracy	ROC AUC	F1
nPrint	95.4	99.7	95.5
Nmap	92.7	99.3	92.9

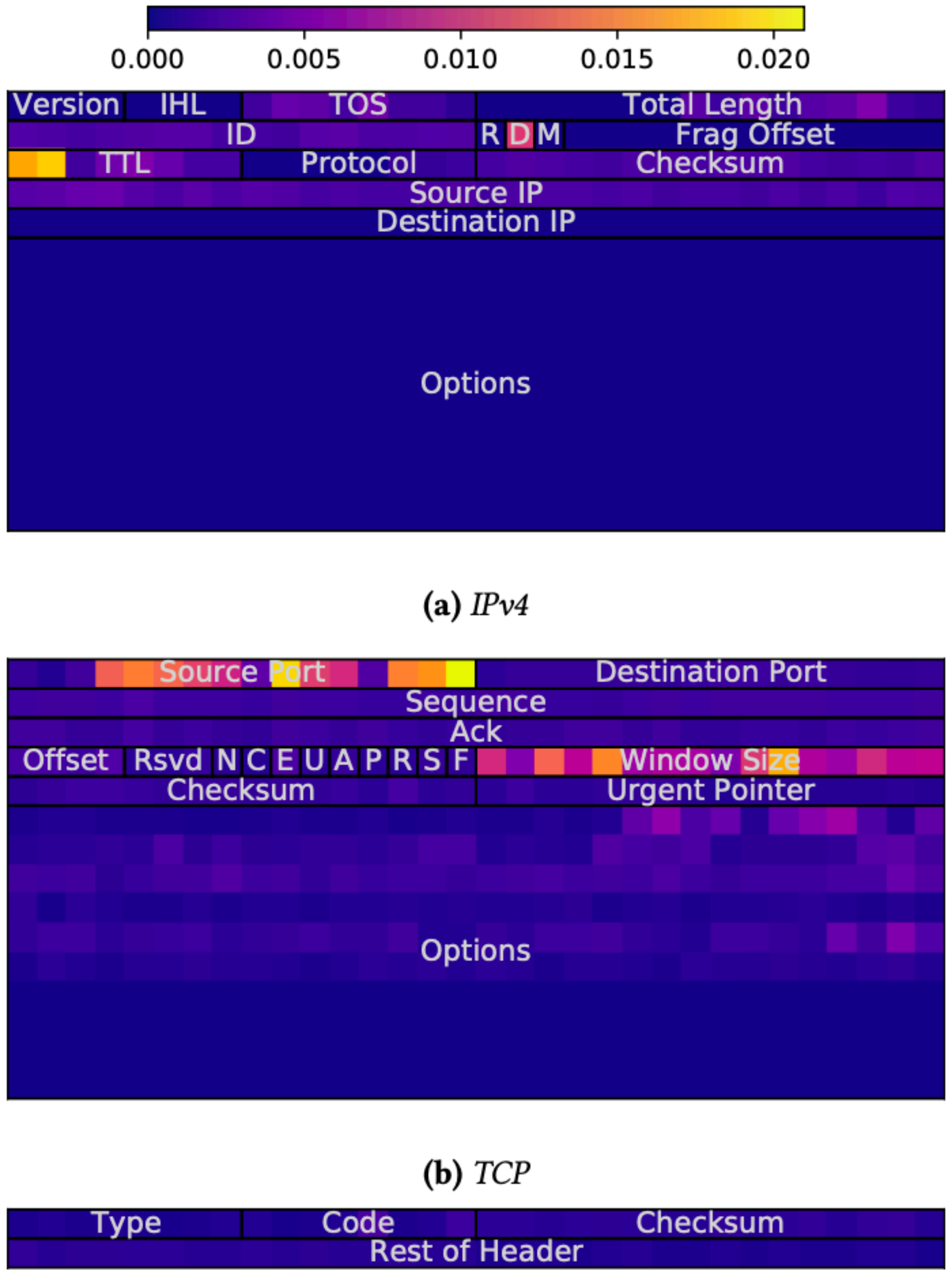
nPrint Enables
Interpretable Machine Learning

- Map features to packet header semantics!

- <u>Automatically learn</u>
  - IP TTL

  - TCP options, window size

  - Source port identifies IoT vs Routers

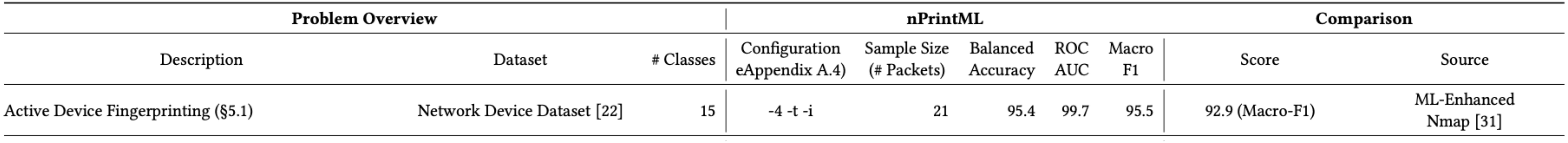


(a) *IPv4*

(b) *TCP*

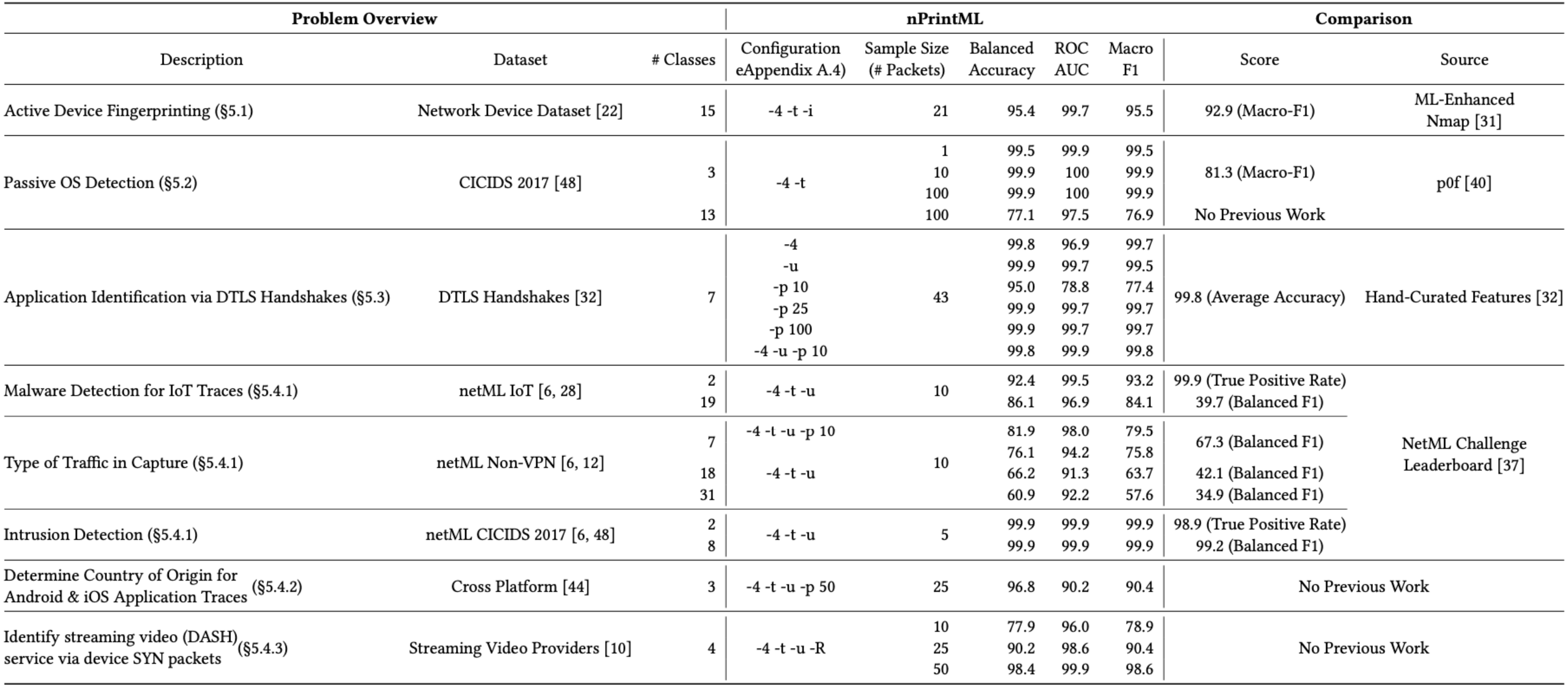
(c) *ICMP*

nPrintML’s Breadth



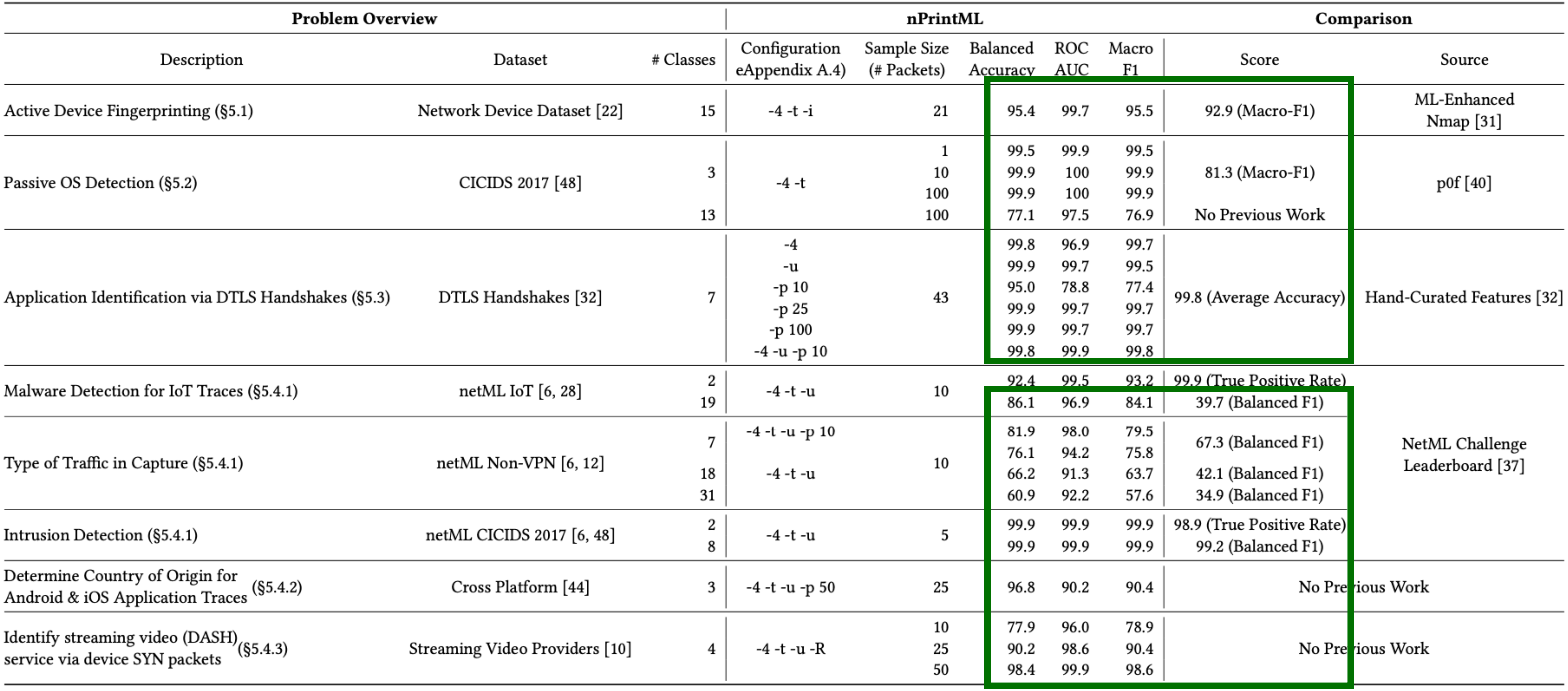
Problem Overview			nPrintML					Comparison	
Description	Dataset	# Classes	Configuration eAppendix A.4)	Sample Size (# Packets)	Balanced Accuracy	ROC AUC	Macro F1	Score	Source
Active Device Fingerprinting (§5.1)	Network Device Dataset [22]	15	-4 -t -i	21	95.4	99.7	95.5	92.9 (Macro-F1)	ML-Enhanced Nmap [31]

8 Discrete Case Studies



Problem Overview			nPrintML					Comparison	
Description	Dataset	# Classes	Configuration eAppendix A.4)	Sample Size (# Packets)	Balanced Accuracy	ROC AUC	Macro F1	Score	Source
Active Device Fingerprinting (§5.1)	Network Device Dataset [22]	15	-4 -t -i	21	95.4	99.7	95.5	92.9 (Macro-F1)	ML-Enhanced Nmap [31]
Passive OS Detection (§5.2)	CICIDS 2017 [48]	3	-4 -t	1	99.5	99.9	99.5	81.3 (Macro-F1)	p0f [40]
				10	99.9	100	99.9		
				100	99.9	100	99.9		
		13		100	77.1	97.5	76.9	No Previous Work	
Application Identification via DTLS Handshakes (§5.3)	DTLS Handshakes [32]	7	-4	43	99.8	96.9	99.7	99.8 (Average Accuracy)	Hand-Curated Features [32]
			-u		99.9	99.7	99.5		
			-p 10		95.0	78.8	77.4		
			-p 25		99.9	99.7	99.7		
			-p 100		99.9	99.7	99.7		
			-4 -u -p 10		99.8	99.9	99.8		
Malware Detection for IoT Traces (§5.4.1)	netML IoT [6, 28]	2	-4 -t -u	10	92.4	99.5	93.2	99.9 (True Positive Rate)	NetML Challenge Leaderboard [37]
		19			86.1	96.9	84.1	39.7 (Balanced F1)	
Type of Traffic in Capture (§5.4.1)	netML Non-VPN [6, 12]	7	-4 -t -u -p 10	10	81.9	98.0	79.5	67.3 (Balanced F1)	
					76.1	94.2	75.8		
		18	-4 -t -u		66.2	91.3	63.7	42.1 (Balanced F1)	
		31			60.9	92.2	57.6	34.9 (Balanced F1)	
Intrusion Detection (§5.4.1)	netML CICIDS 2017 [6, 48]	2	-4 -t -u	5	99.9	99.9	99.9	98.9 (True Positive Rate)	
		8			99.9	99.9	99.9	99.2 (Balanced F1)	
Determine Country of Origin for Android & iOS Application Traces (§5.4.2)	Cross Platform [44]	3	-4 -t -u -p 50	25	96.8	90.2	90.4	No Previous Work	
Identify streaming video (DASH) service via device SYN packets (§5.4.3)	Streaming Video Providers [10]	4	-4 -t -u -R	10	77.9	96.0	78.9	No Previous Work	
				25	90.2	98.6	90.4		
				50	98.4	99.9	98.6		

Outperforming hand-engineered solutions



Problem Overview			nPrintML					Comparison	
Description	Dataset	# Classes	Configuration eAppendix A.4)	Sample Size (# Packets)	Balanced Accuracy	ROC AUC	Macro F1	Score	Source
Active Device Fingerprinting (§5.1)	Network Device Dataset [22]	15	-4 -t -i	21	95.4	99.7	95.5	92.9 (Macro-F1)	ML-Enhanced Nmap [31]
Passive OS Detection (§5.2)	CICIDS 2017 [48]	3	-4 -t	1	99.5	99.9	99.5	81.3 (Macro-F1)	p0f [40]
				10	99.9	100	99.9		
				100	99.9	100	99.9		
		13		100	77.1	97.5	76.9	No Previous Work	
Application Identification via DTLS Handshakes (§5.3)	DTLS Handshakes [32]	7	-4	43	99.8	96.9	99.7	99.8 (Average Accuracy)	Hand-Curated Features [32]
			-u		99.9	99.7	99.5		
			-p 10		95.0	78.8	77.4		
			-p 25		99.9	99.7	99.7		
			-p 100		99.9	99.7	99.7		
			-4 -u -p 10		99.8	99.9	99.8		
Malware Detection for IoT Traces (§5.4.1)	netML IoT [6, 28]	2	-4 -t -u	10	92.4	99.5	93.2	99.9 (True Positive Rate)	NetML Challenge Leaderboard [37]
		19			86.1	96.9	84.1	39.7 (Balanced F1)	
Type of Traffic in Capture (§5.4.1)	netML Non-VPN [6, 12]	7	-4 -t -u -p 10	10	81.9	98.0	79.5	67.3 (Balanced F1)	
					76.1	94.2	75.8		
		18	-4 -t -u		66.2	91.3	63.7	42.1 (Balanced F1)	
		31			60.9	92.2	57.6	34.9 (Balanced F1)	
Intrusion Detection (§5.4.1)	netML CICIDS 2017 [6, 48]	2	-4 -t -u	5	99.9	99.9	99.9	98.9 (True Positive Rate)	
		8			99.9	99.9	99.9	99.2 (Balanced F1)	
Determine Country of Origin for Android & iOS Application Traces (§5.4.2)	Cross Platform [44]	3	-4 -t -u -p 50	25	96.8	90.2	90.4	No Previous Work	
Identify streaming video (DASH) service via device SYN packets (§5.4.3)	Streaming Video Providers [10]	4	-4 -t -u -R	10	77.9	96.0	78.9	No Previous Work	
				25	90.2	98.6	90.4		
				50	98.4	99.9	98.6		

nPrint Is Open Source

- 8 protocols implemented
- Relative & absolute timestamps
- Input formats - live capture, PCAP, scan data, nPrints

nPrintML Is Open Source

- Application Identification

  -
    ```
    nprintml —pcap-dir pcaps/ -L labels.csv -a pcap -4 -u -p 10
    ```

- Passive OS detection

  -
    ```
    nprintml -P traffic.pcap -L labels.csv -a index -4 -t
    ```

Thank You!

github.com/nprint/

References

1. Rimmer, Vera, et al. "Automated website fingerprinting through deep learning." 25th *Annual Network & Distributed System Security Symposium*. NDSS, 2018
2. Oh, Se Eun, Saikrishna Sunkam, and Nicholas Hopper. "p1-FP: Extraction, Classification, and Prediction of Website Fingerprints with Deep Learning." *Proceedings on Privacy Enhancing Technologies* 2019.3 (2019): 191-209.
3. https://skminhaj.wordpress.com/2016/02/15/tcp-segment-vs-udp-datagram-header-format/
4. https://arxiv.org/pdf/2006.13086.pdf
5. https://www.shodan.io/
6. Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani, “Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization”, 4th International Conference on Information Systems Security and Privacy (ICISSP), Portugal, January 2018
7. Sambasivan, Nithya, et al. "“Everyone wants to do the model work, not the data work”: Data Cascades in High-Stakes AI.” *proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 2021.
8. Erickson, Nick, et al. "Autogluon-tabular: Robust and accurate automl for structured data." *arXiv preprint arXiv:2003.06505* (2020).