Neural Architecture Search for Adaptive Website Fingerprinting

Gurjot Singh[1,2] · Rob Jansen[3] · Diogo Barradas[1]

[1]University of Waterloo, [2]University of New South Wales
[3]U.S. Naval Research Laboratory

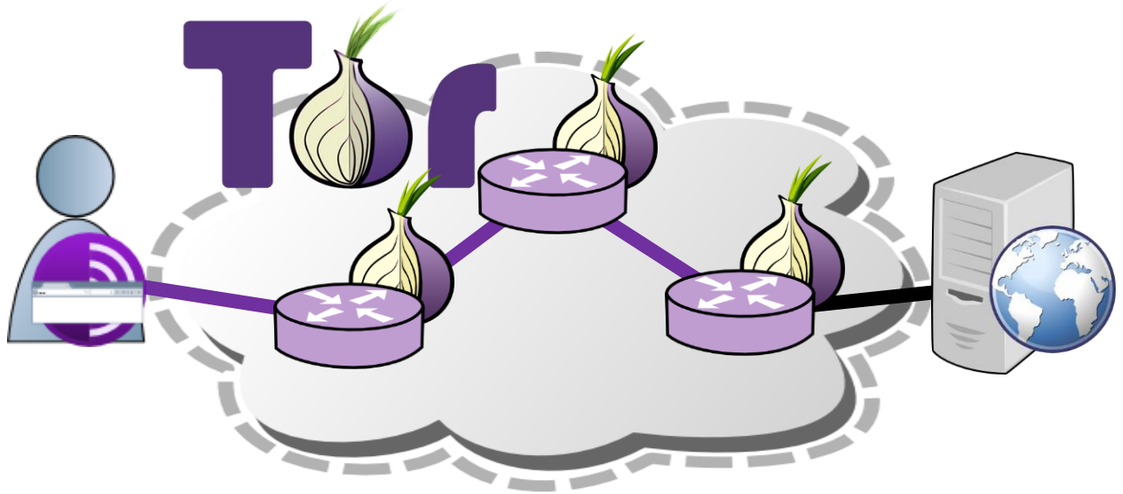
Workshop on Privacy in the Electronic Society (WPES) 2026

Rob Jansen
U.S. NAVAL RESEARCH LABORATORY
robgjansen.com

Website fingerprinting • Tor traffic analysis • Neural architecture search

Tor hides content and destination, but low-latency traffic patterns remain visible.

- Tor routes traffic through multiple relays to separate **identity** from **routing**.
- Encryption does not remove every pattern: **direction**, **timing**, and **volume** are still observable.
- Those patterns create a **privacy risk** that researchers study as **website fingerprinting**.



Onion-encrypted payloads but traffic patterns are still observable

Website Fingerprinting: the Classification Problem

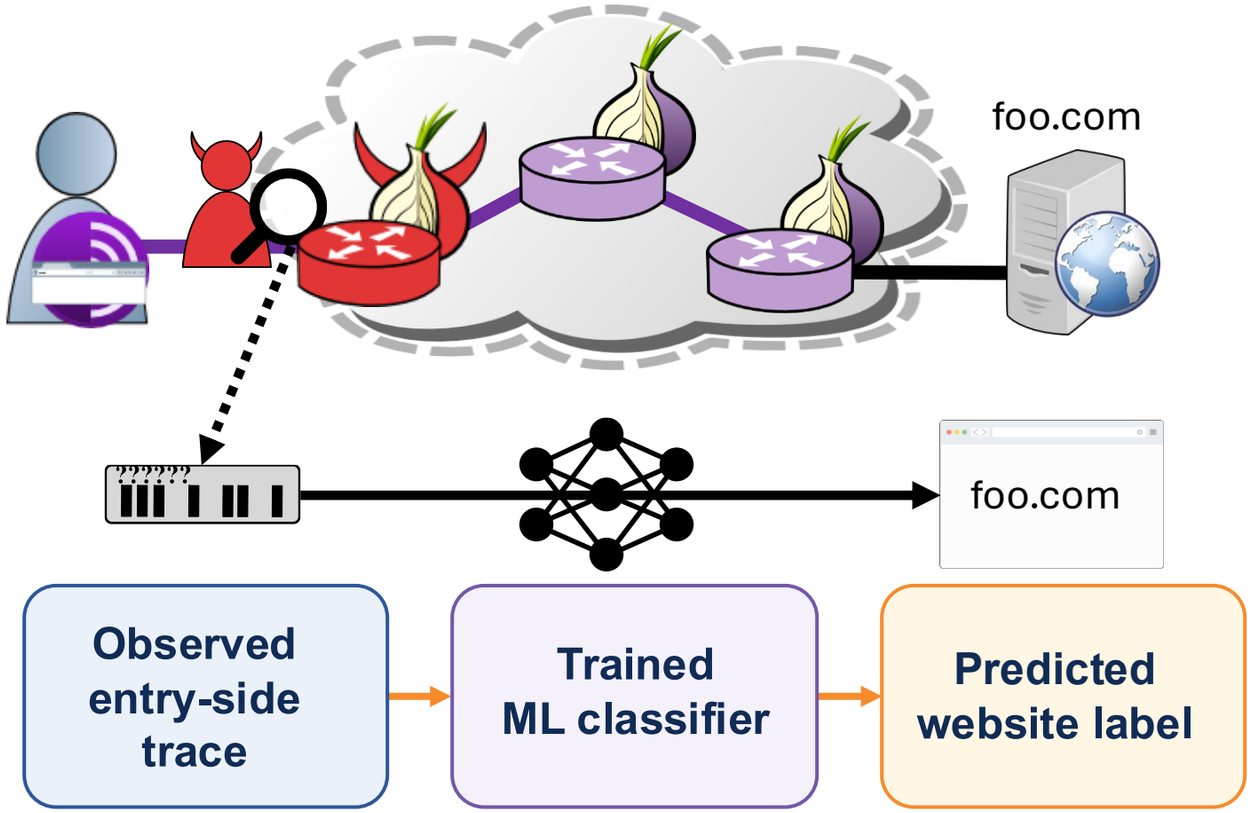
Website fingerprinting (WF) attacks can link a source to its destination, breaking Tor's anonymity

Threat model

The adversary with a suitable observation point:

1. collects labeled reference traces for monitored destinations
2. uses reference traces to train a ML classifier
3. deploys classifier to classify a real user trace

An observer learns a mapping from encrypted traffic traces to likely destinations



New architectures are underexplored in recent work

Researchers have improved data, features, and evaluation settings.
The classifier architecture is still often chosen by hand.

1. Better data

Synthetic collections can underrepresent real traffic variation. Recent work has shifted attention toward genuine Tor traces and realistic evaluation.

[Usenix ‘22, PAM ‘26]

2. Better features

WF work has introduced timing, multi-channel, and engineered representations to expose useful trace structure.

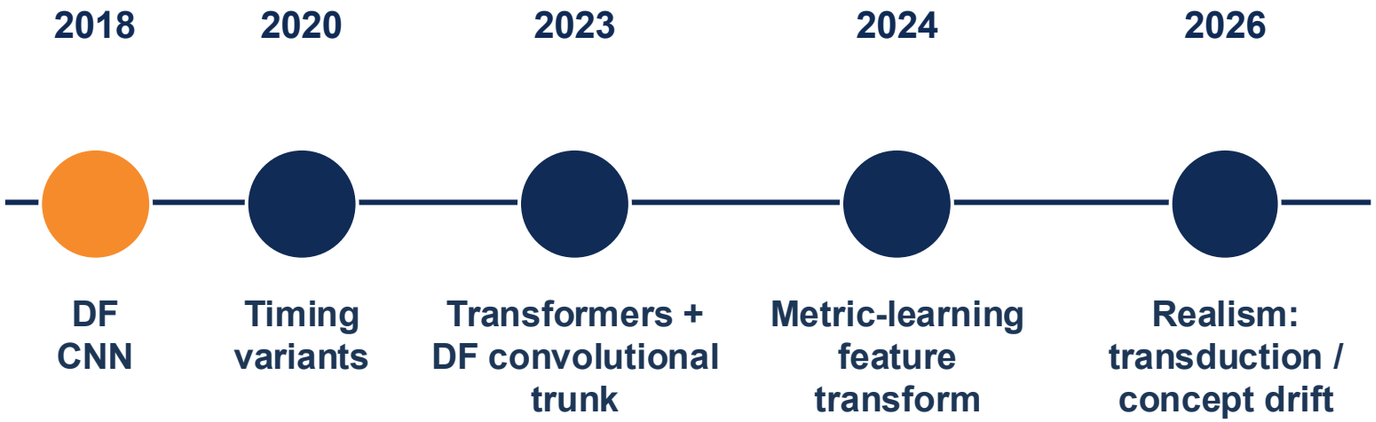
[TIFS ‘24]

3. Fixed architectures

Many attacks retain legacy CNN backbones or hand-crafted hybrids. Architecture selection is not being systematically adapted to the task.

[Table 1 in our paper]

Deep Fingerprinting (DF) is used across most recent WF research.



Limitation

Heuristic selection can miss **unconventional architectures** that can perhaps better exploit a particular trace representation, traffic dataset, or defense.

Year	Venue	Work	Role of DF
2018	CCS	[80]	– Original DF paper
2018	CCS	[49]	◇ Study (corroborates information-leakage analysis)
2019	CCS	[81]	■ DF backbone (embeds traces for N-shot matching)
2019	PETS	[7]	○ Baseline (work introduces a ResNet-based CNN)
2019	PETS	[61]	○ Baseline (work introduces a CNN/DNN architecture)
2020	S&P	[85]	○ Baseline (DF compared to precision-optimized attacks)
2020	PETS	[69]	■ DF backbone (classifies new timing features)
2020	PETS	[67]	▲ DF as base (augmented by a website oracle)
2021	CCS	[86]	◇ Study (defense evaluation)
2021	PETS	[60]	○ Baseline (introduces a semi-supervised GAN)
2021	PETS	[82]	▲ DF as base (one of several, QUIC setting)
2022	USENIX	[15]	■ DF backbone (via TF, adapted to the online setting)
2022	TIFS	[90]	■ DF backbone (ensemble of DF snapshots)
2023	CCS	[2]	■ DF backbone (self-supervised pretraining)
2023	CCS	[45]	■ DF backbone (CNN stem feeding a Transformer)
2023	USENIX	[75]	○ Baseline (introduces the TAM representation)
2023	S&P	[22]	○ Baseline (introduces the Trans-WF transformer)
2023	S&P	[55]	◇ Study (defense SoK)
2023	PETS	[42]	◇ Study (dataset built via simulation)
2024	CCS	[21]	○ Baseline (introduces a contrastive encoder)
2024	CCS	[101]	■ DF backbone (metric-learning feature transform)
2024	USENIX	[56]	▲ DF as base (boosted with subpage sets)
2024	TIFS	[54]	○ Baseline (introduces a multi-channel transformer)
2025	CCS	[76]	○ Baseline (introduces the CIF representation)
2025	S&P	[24]	○ Baseline (introduces a state-space model)
2026	NDSS	[41]	■ DF backbone (classifies transduced traces)
2026	NDSS	[23]	▲ DF as base (fine-tuned against drift)

Role of DF: – original DF, ■ DF is the attack's backbone, ▲ one of several base attacks / enhanced, ○ comparison baseline, ◇ object of study.

Research question:
Can we *automatically* discover a compact DNN tailored to a WF environment?

We introduce Cricket: a WF framework for neural architecture search based on the popular one-shot DARTS method [ICLR 2019]

Cricket replaces a *fixed model* with a *search procedure* that produces a model for a target environment

Input conversion

Represent Tor traces as C channel × L length sequential arrays

Generalizes existing representations, so we can determine which yield the best cricket models

Search Space Adaptation

Adapt DARTS search space dimensions from 2D to 1D sequential data

Avoids improperly learning 2D spatial correlations which Tor traces are not known to exhibit

Neural Architecture Search

Run adapted DARTS to search for and then train a new model architecture

Find effective model by jointly optimizing network weights and architecture parameters

Goal: discover architectures that are effective, compact, and re-searchable when data, features, or defenses change.

Neural Architecture Search (NAS)

Search Space

The collection of operations (e.g., convolution, pooling) and their params (e.g., kernel sizes, strides) to explore

Larger search space needs more runtime, smaller search space converged quicker

Search Strategy

An optimization process for exploring the search space

E.g., random search, genetic algorithms, Bayesian optimization, Monte Carlo tree search

Performance Estimation

Quickly predict performance to avoid full architecture training

E.g., shorter training, reduced data, learning-curve extrapolation via surrogate models, transfer weights between architectures

Differentiable ARchiTecture Search (DARTS)

Search Space

Cell-based: represents an architecture as a stack of n cells, each is a small directed acyclic graph (DAG)

Search Strategy

One-shot: use supernetworks that include all potential architectures as subnetworks (can evaluate multiple architectures from one supernet)

Performance Estimation

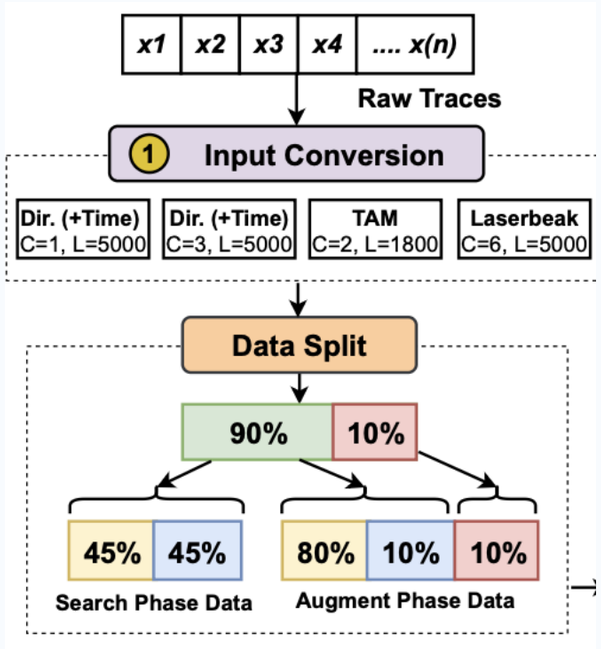
Train the supernetwork and evaluate all subnets (the subnets inherit the weights from the trained supernet)

Cricket adapts DARTS for 1D Traces and 1D Cells

Tor traces are sequences, not images.
Cricket adapts DARTS to operate on one-dimensional temporal sequences.

Input Conversion

Represent Tor traces as
C channel × L length sequential arrays

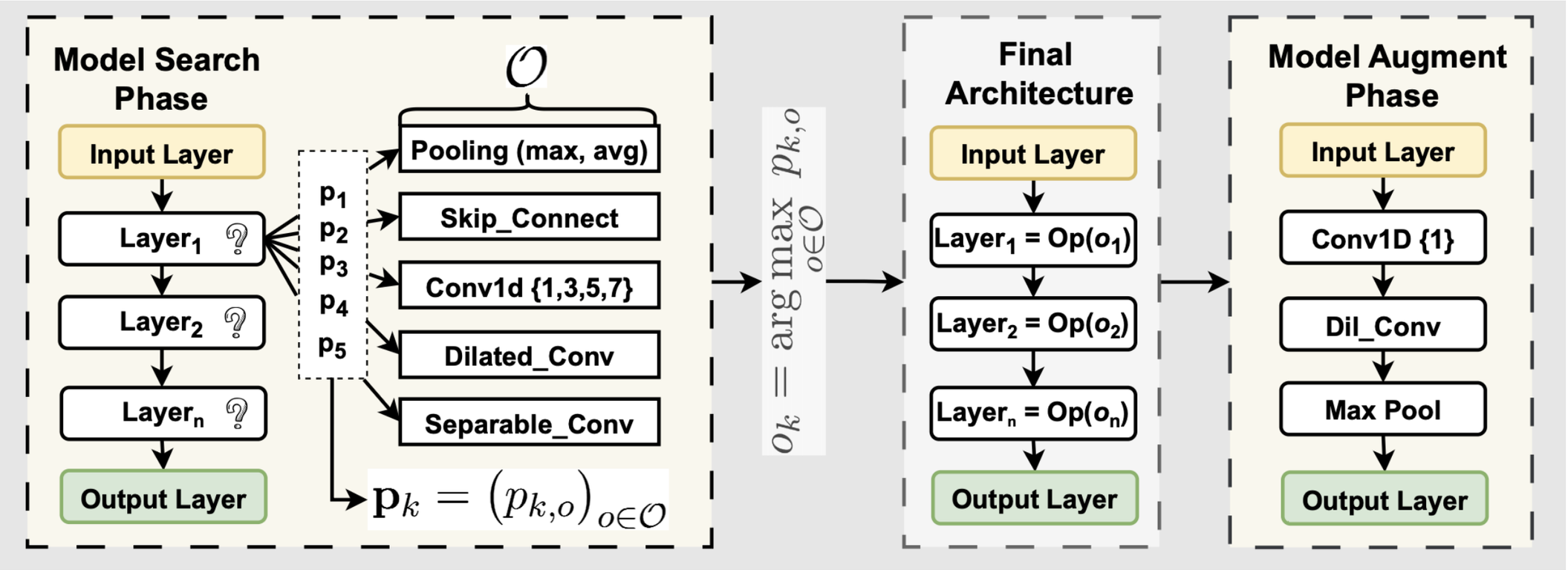


Search-space adaptations

Cricket replaces DARTS 2D operations with 1D operations. This permits models to capture short burst patterns as well as longer request-response dependencies.

- Conv1D k=1,3,5,7,9
- Separable Conv1D
- Dilated Conv1D
- Max / Avg pool
- Skip connection

Cricket's search consists of model search and model augment phases



Cricket performs a bi-level optimization to jointly learn optimal weights and architectural parameters for a neural network model

The final architecture is retrained, validated, and tested on the complete dataset

Evaluation Setup

The evaluation tests efficacy, robustness, transferability, stability, and practicality.

Datasets

AWF_{100}: 100×100 synthetic traces

GTT_{100}: 100×100 genuine traces

GTT_{200}: 200×2000 genuine traces

$OpenGTT_{200}$: GTT_{200} + 400,000 unmonitored sites in open-world model

Baselines Attacks

Deep Fingerprinting (DF) → direction

Tik-Tok → directional time

Robust Fingerprinting (RF) → traffic aggregation matrix (TAM)

Laserbeak → engineered features

Defenses

CS-BuFLO, Tamaraw → fixed-rate padding

WTF-PAD → Adaptive padding

FRONT → Randomized early padding

RegulaTor → Regulation

Evaluation Metrics

Closed world: multi-class classification accuracy
Open world: precision, recall, and F1
Compactness: parameter count

Cricket discovers accurate and compact models across feature representations

Direction Representation	**DF** 61.0% accuracy 3.96M params	**Cricket** **65.0% accuracy** 1.90M params	**+4.0 pp accuracy** 48% as many params
Traffic Aggregation Matrix (TAM) Representation	**RF** 58.59% accuracy 1.01M params	**Cricket** **62.4% accuracy** 1.90M params	**+3.81 pp accuracy** more params
Laserbeak features Representation	**Laserbeak** 61.70% accuracy 4.78M params	**Cricket** **67.32% accuracy** 1.92M params	**+5.62 pp accuracy** 40% as many params

Model Size

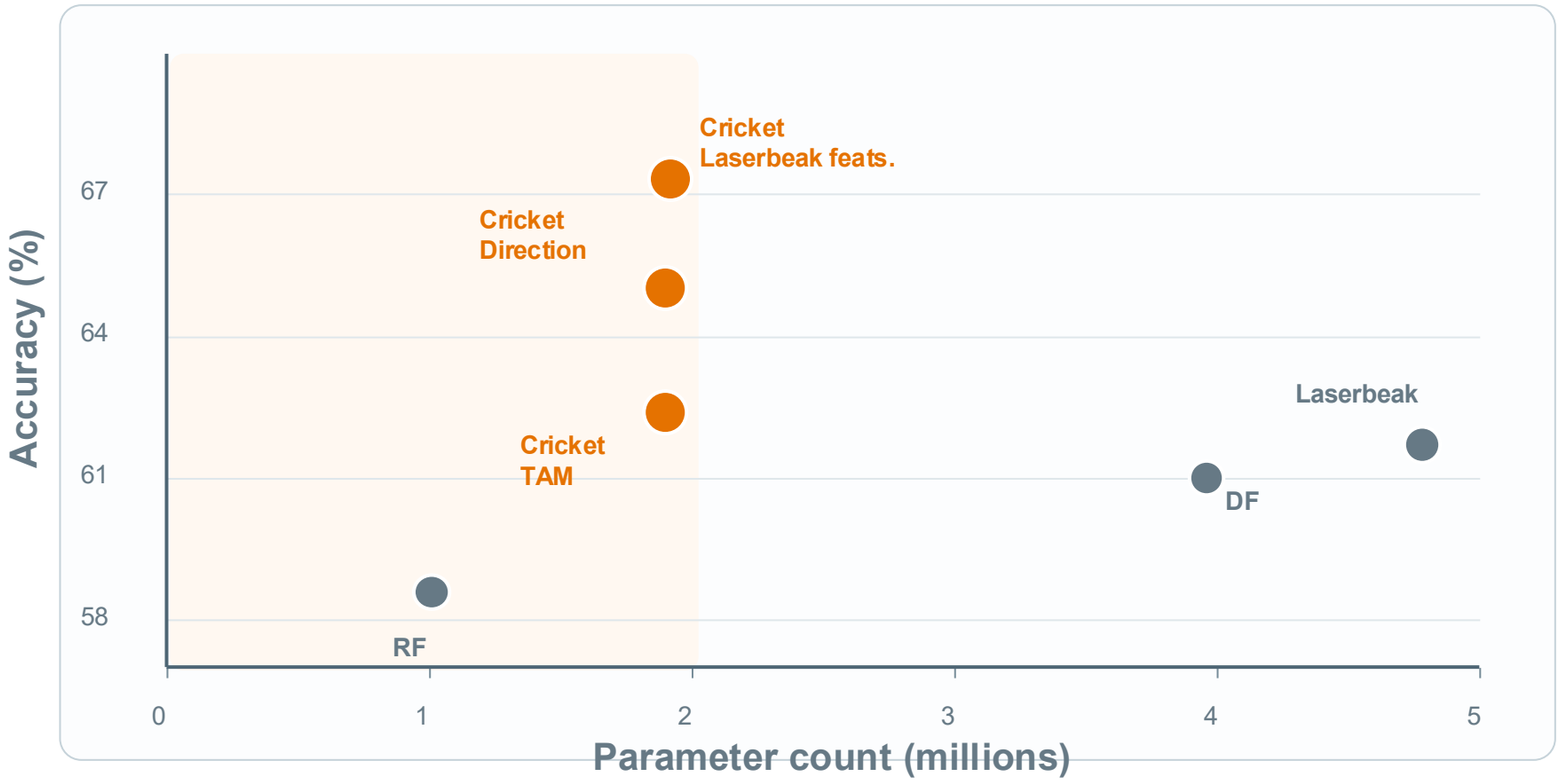
Deep baselines
~4–5M parameters

Typical Cricket model
< 2M parameters

Caveat
RF smaller; Cricket more accurate

similar results hold for closed-world GTT_{200}

Compactness and accuracy form a favorable trade-off



Key observation

All Cricket variants remain below **2M** parameters

Highest accuracy

Cricket with laserbeak feats.

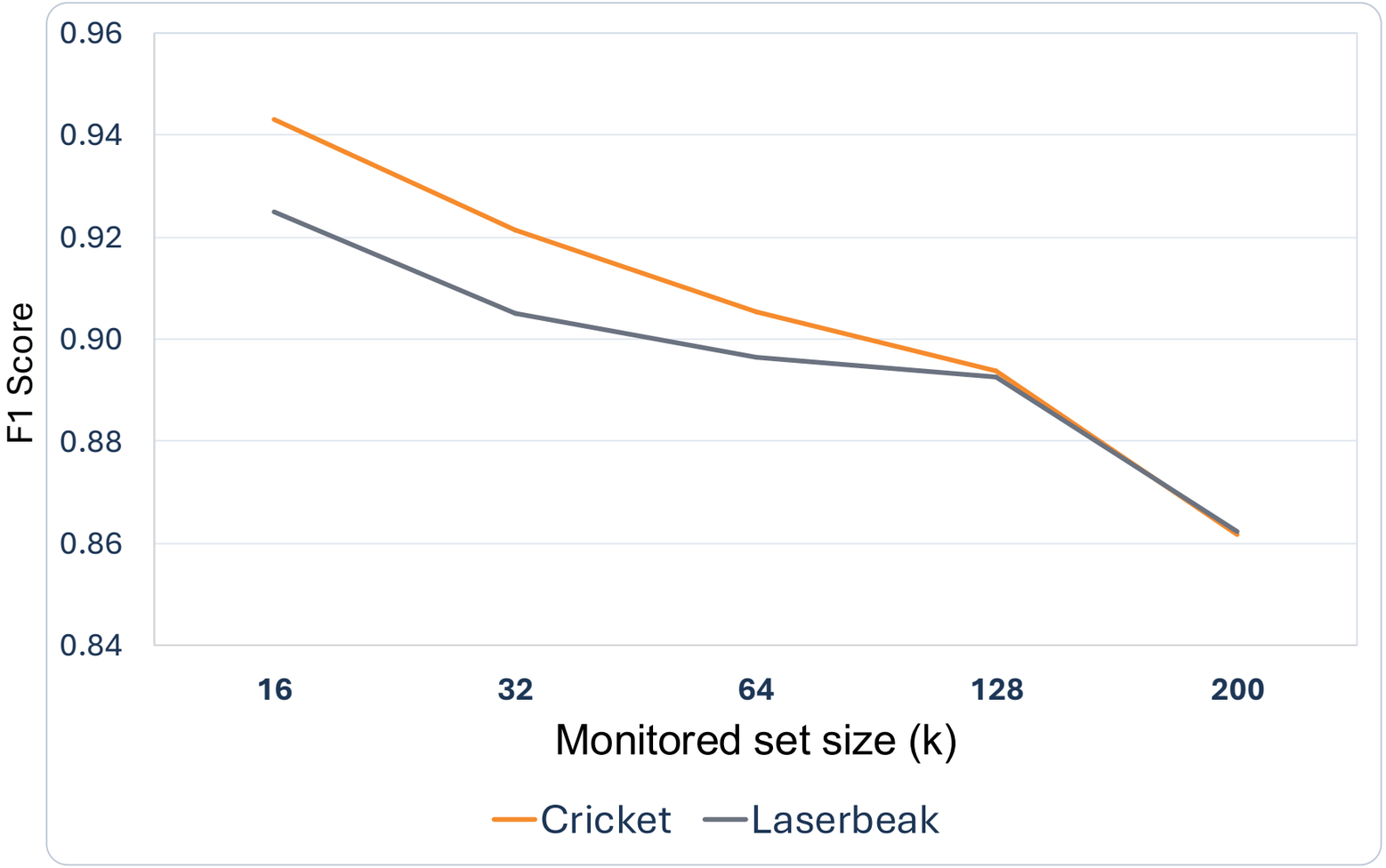
67.32% at 1.92M params

Most accurate baseline

Laserbeak

61.7% at 4.78M params

Cricket meets or exceeds SOTA attack efficacy in more demanding setting



Set size k=64

Cricket: 0.9055 F1
Laserbeak: 0.8964 F1

Largest margin

Set-16: +1.81 pp F1
in favor of Cricket

Takeaway

Cricket maintains competitive in open-world setting while using almost 3M fewer params

Cricket meets or exceeds SOTA attack efficacy against defenses in both closed and open world

Closed World

Defense	Attack	Accuracy (%)		
		AWF_{100}	GTT_{100}	GTT_{200}
CS-BuFLO	Laserbeak	11.40	16.00	17.82
	CRICKET	11.00	14.00	17.18
Tamaraw	Laserbeak	4.30	4.80	3.90
	CRICKET	4.54	4.70	4.06
WTF-PAD	Laserbeak	89.20	50.80	77.16
	CRICKET	89.90	62.50	79.01
FRONT_T1	Laserbeak	87.40	53.10	77.05
	CRICKET	87.30	64.20	79.44
FRONT_T2	Laserbeak	80.90	50.30	76.51
	CRICKET	87.80	63.40	78.94
RegulaTor	Laserbeak	23.80	25.80	39.86
	CRICKET	27.00	27.20	44.17

Open World

Defense	Attack	Efficacy on $OpenGTT_{200}$ (*set-64*)		
		Precision	Recall	F1 Score
CS-BuFLO	Laserbeak	0.1432	0.1570	0.1404
	CRICKET	0.1370	0.1529	0.1394
Tamaraw	Laserbeak	0.0349	0.0585	0.0436
	CRICKET	0.0351	0.0585	0.0435
WTF-PAD	Laserbeak	0.8777	0.8776	0.8752
	CRICKET	0.8852	0.8893	0.8849
FRONT_T1	Laserbeak	0.8777	0.8786	0.8762
	CRICKET	0.8884	0.8911	0.8867
FRONT_T2	Laserbeak	0.8740	0.8740	0.8711
	CRICKET	0.8865	0.8891	0.8850
RegulaTor	Laserbeak	0.6491	0.6739	0.6340
	CRICKET	0.6826	0.6816	0.6239

Fixed-rate defenses
CS-BuFLO, Tamaraw greatly suppress both attacks.

Largest gain
Vs FRONT T2
+13 pp (+26%)

Conclusions

Cricket makes the neural architecture a variable that can adapt to the WF environment.

Adaptive

DARTS is adapted to 1D Tor cell traces and searches a task-specific architecture.

Effective

On genuine closed-world data, Cricket improves over state-of-the-art baselines by 2–7 points.

Compact

Cricket models are typically below 2M parameters, about one-third to one-half of several deep baselines.

Takeaway: we found that evaluating and updating model architectures alongside datasets, trace representations, and defenses yielded better results.