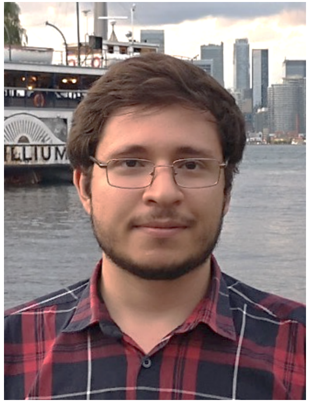
NeuRA: Using Neural Networks to Improve WiFi Rate Adaptation

Shervin Khastoo, Tim Brecht and Ali Abedi



UNIVERSITY OF WATERLOO | DAVID R. CHERITON SCHOOL OF COMPUTER SCIENCE

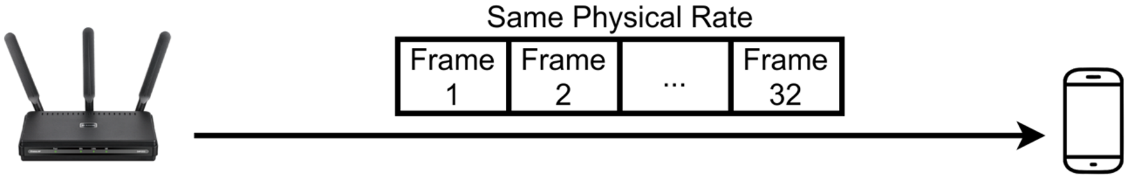
MSWiM, 2020

Background

Two critical decisions before transmitting each frame

1) Which physical (PHY) rate to use

2) How many subframes (MPDUs) to aggregate in a frame (A-MPDU length)



Both can have a big impact on throughput

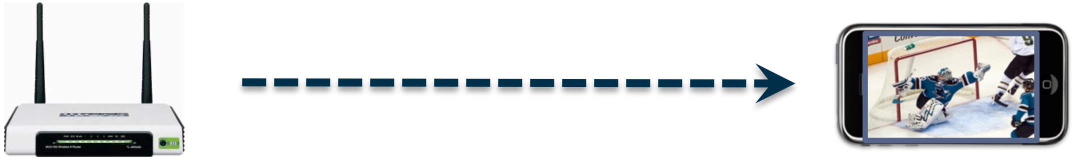
Main Contributions

NeuRA: uses a neural network to improve rate adaptation and throughput

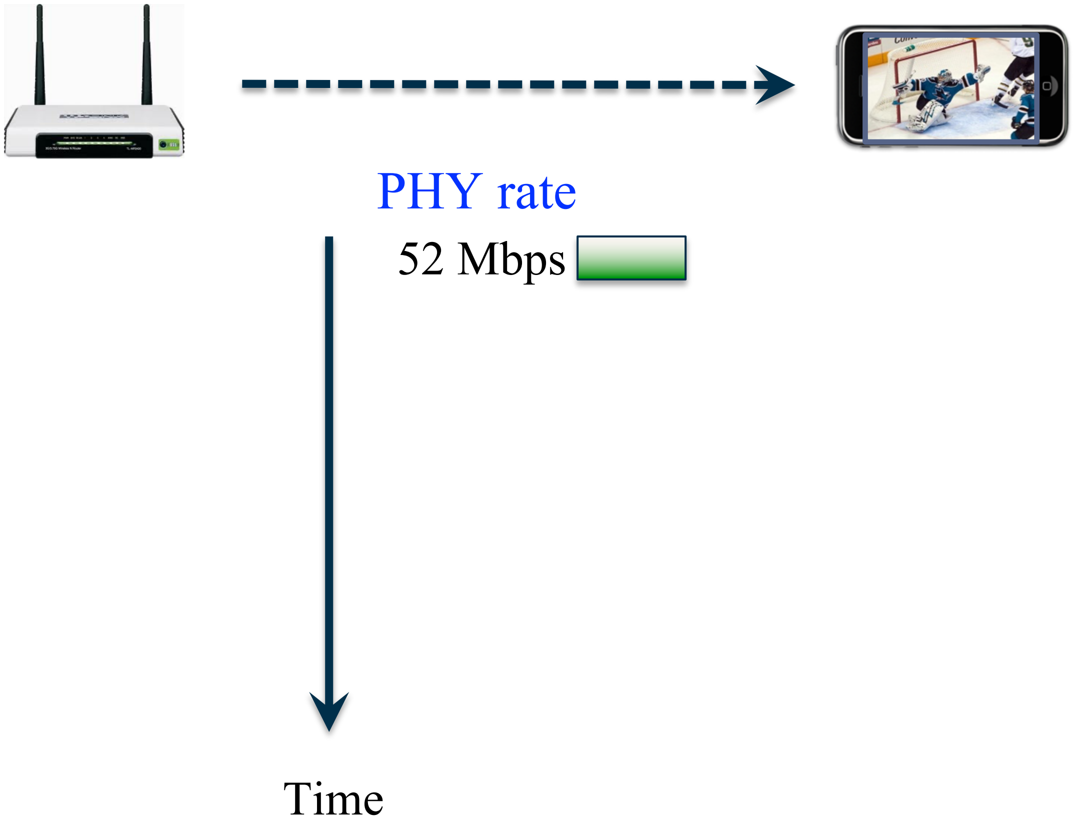
Offline Statistically Optimal: rate adaptation and frame aggregation algorithm

- Upper bound on throughput
- Can finally better determine how well algorithms are performing

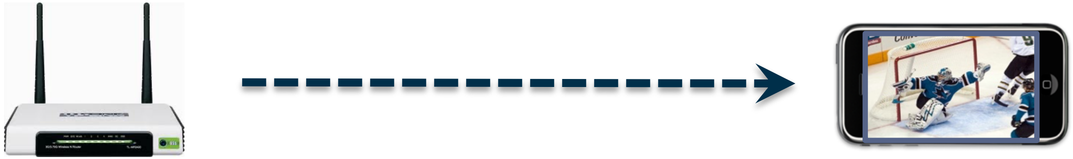
Rate Adaptation



Rate Adaptation



Rate Adaptation



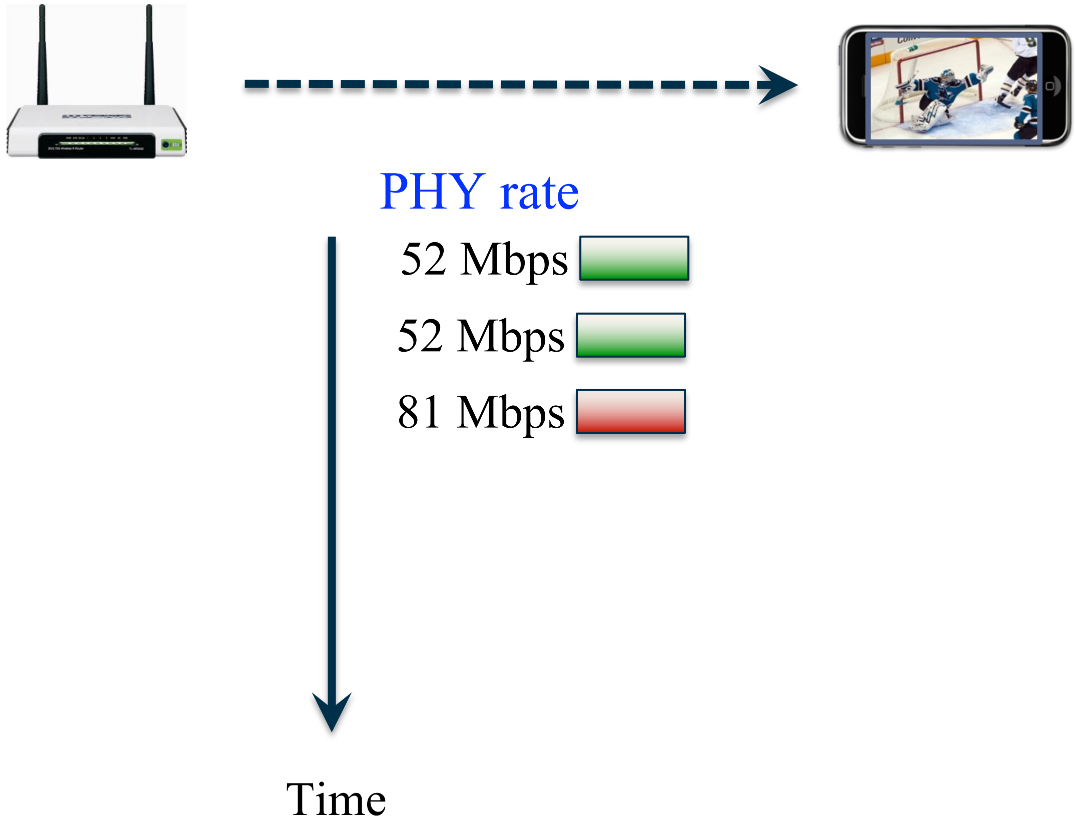
PHY rate

52 Mbps

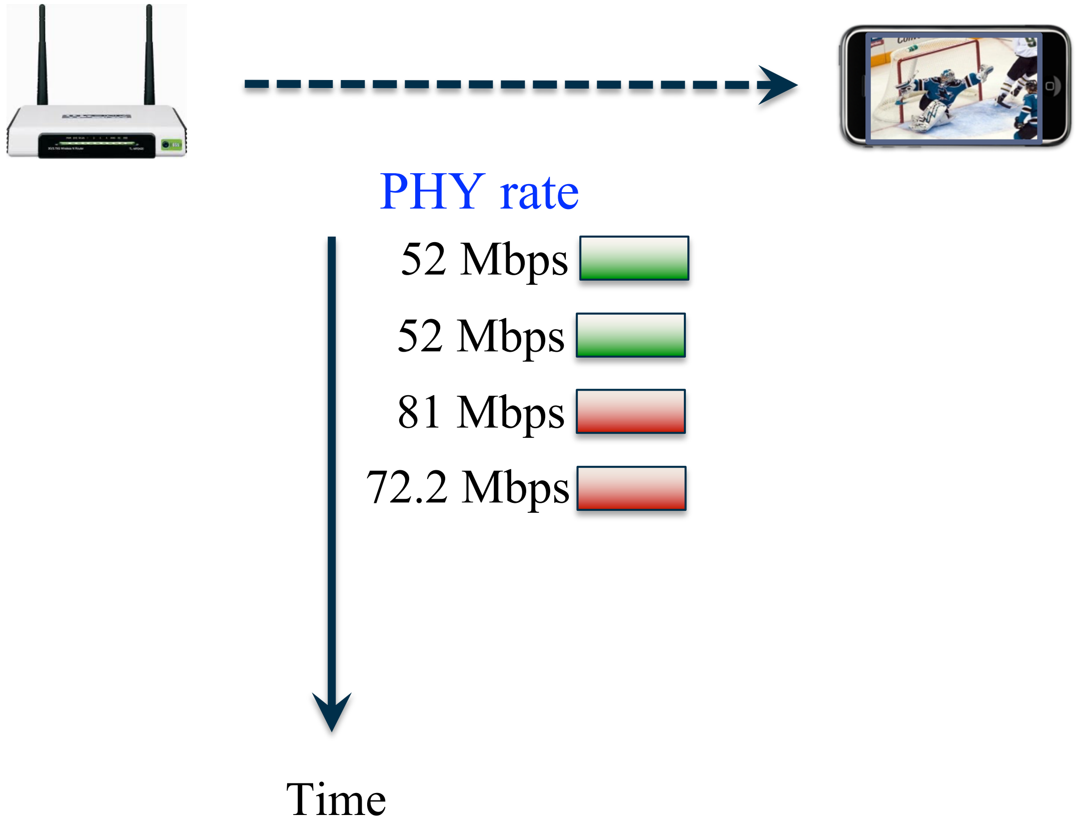
52 Mbps

Time

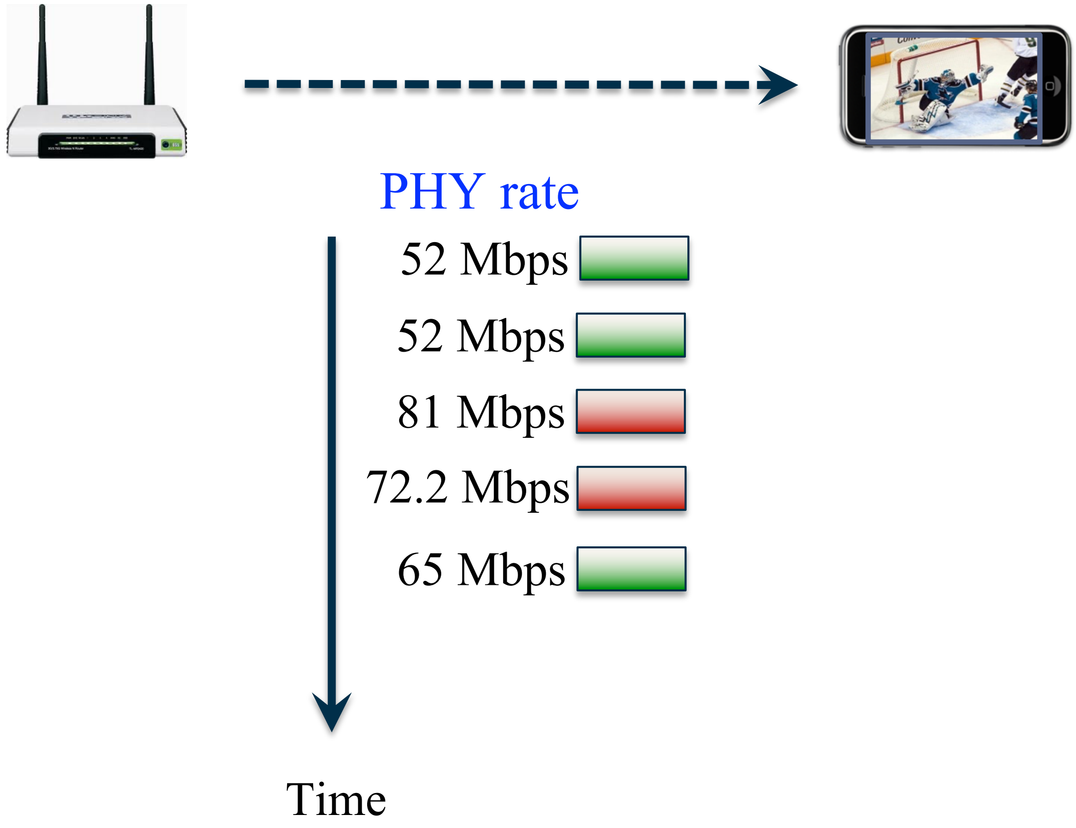
Rate Adaptation



Rate Adaptation



Rate Adaptation



Rate Adaptation

PHY rate

52 Mbps
52 Mbps
81 Mbps
72.2 Mbps
65 Mbps
65 Mbps

Time

Rate Adaptation

PHY rate

52 Mbps

52 Mbps

81 Mbps

72.2 Mbps

65 Mbps

65 Mbps

Time

Challenge:

Channel is constantly changing!

Rate Adaptation

PHY rate

52 Mbps

52 Mbps

81 Mbps

72.2 Mbps

65 Mbps

65 Mbps

Time

Challenge:

Channel is constantly changing!

Rate Adaptation

PHY rate

52 Mbps

52 Mbps

81 Mbps

72.2 Mbps

65 Mbps

65 Mbps

Time

Challenge:

Channel is constantly changing!

Rate Adaptation

PHY rate

52 Mbps
52 Mbps
81 Mbps
72.2 Mbps
65 Mbps
65 Mbps
65 Mbps

Time

Challenge:
Channel is constantly changing!

Rate Adaptation

PHY rate

- 52 Mbps
- 52 Mbps
- 81 Mbps
- 72.2 Mbps
- 65 Mbps
- 65 Mbps
- 65 Mbps

Time

Challenge:
Channel is constantly changing!

Practical algorithms sample
(i.e., test/probe potential rates)

Rate Adaptation

PHY rate

52 Mbps	
52 Mbps	
81 Mbps	
72.2 Mbps	
65 Mbps	
65 Mbps	
65 Mbps	

Time

NeuRA:

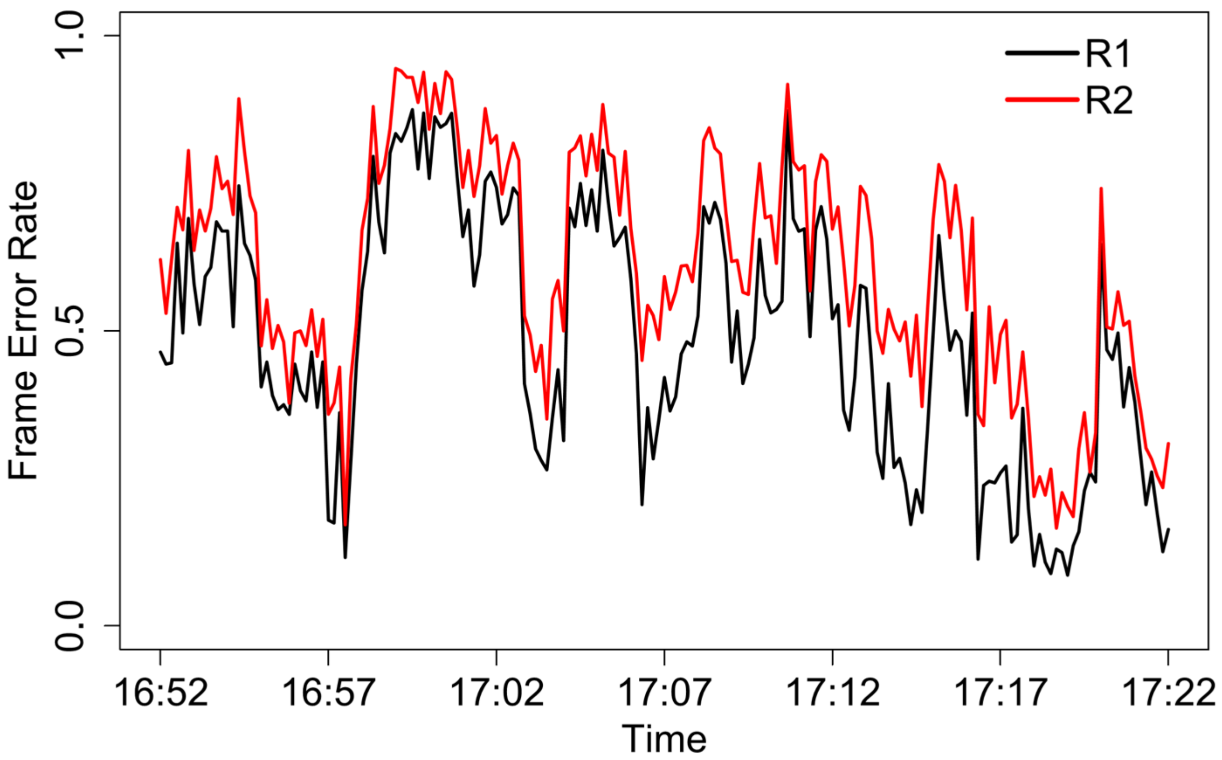
Reduce sampling overhead

- sample smaller subset of rates
- increase throughput

Neural network to

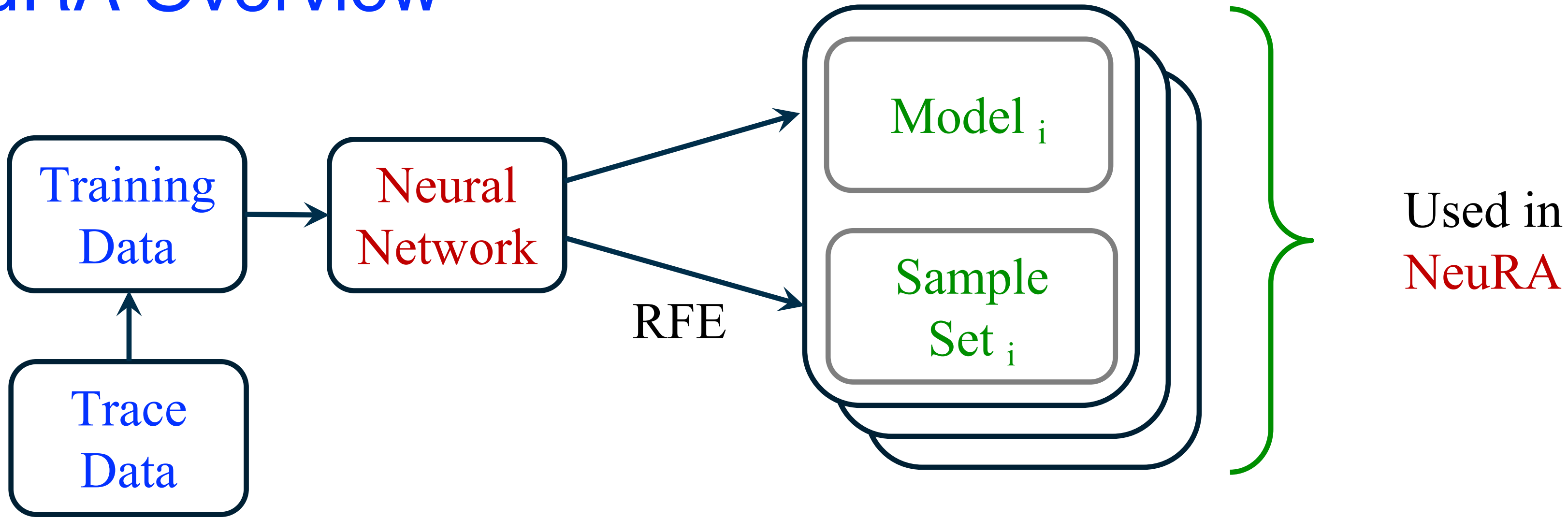
- find good set of rates to sample
- predict tput of other rates

Relationships Exist Between Rates



[Abedi and Brecht, MSWiM, 2016]

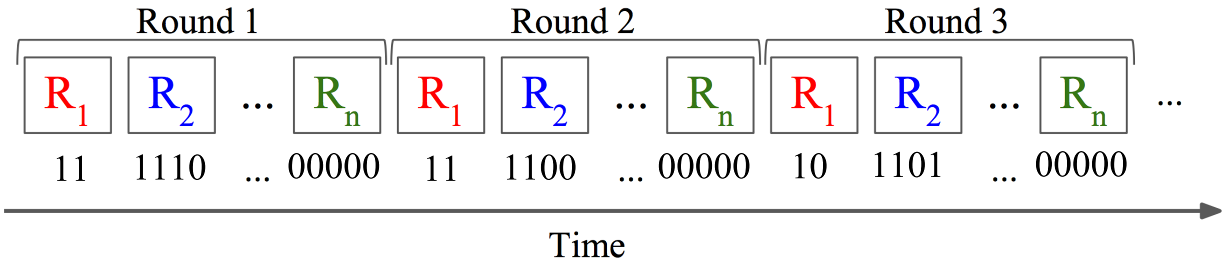
NeuRA Overview



Recursive Feature Eliminate (RFE) optimizes $\frac{\text{Estimation Power}}{\text{Sampling Time}}$

Trace Collection

- Modify WiFi device driver (ath9k)
- Round robin all rates
- Rates see similar channel conditions in round

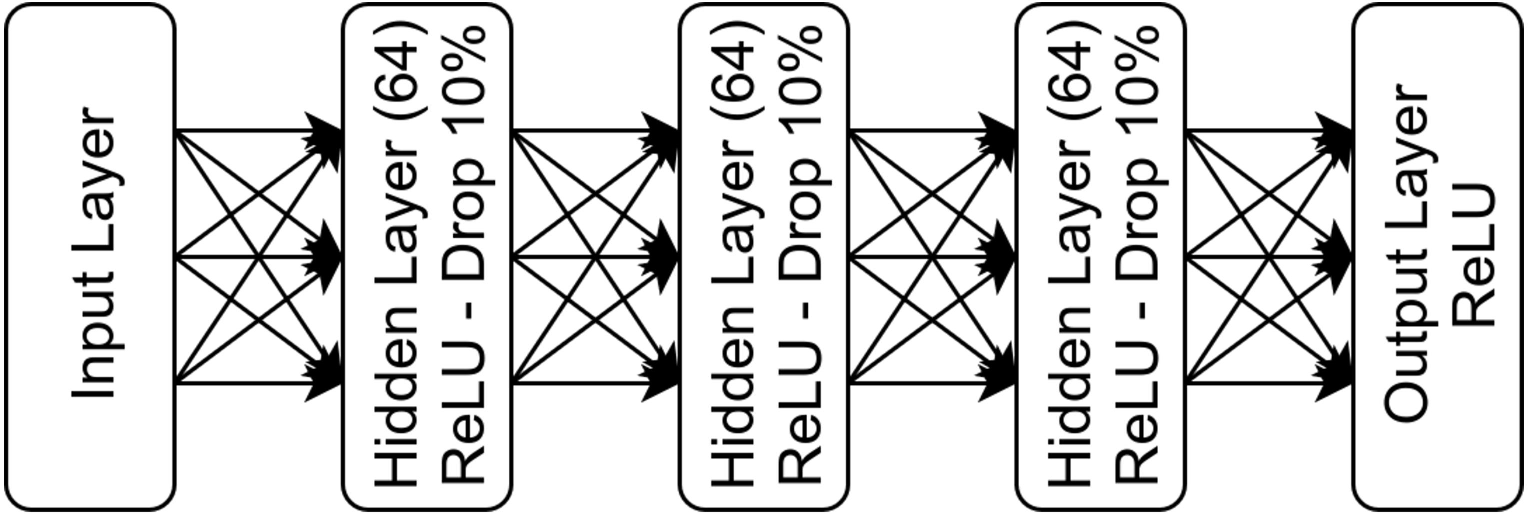


Training Data

- For 1-second time intervals, throughput of each rate is calculated
- Normalize to maximum: ([0, 1] range) to prepare for neural network training

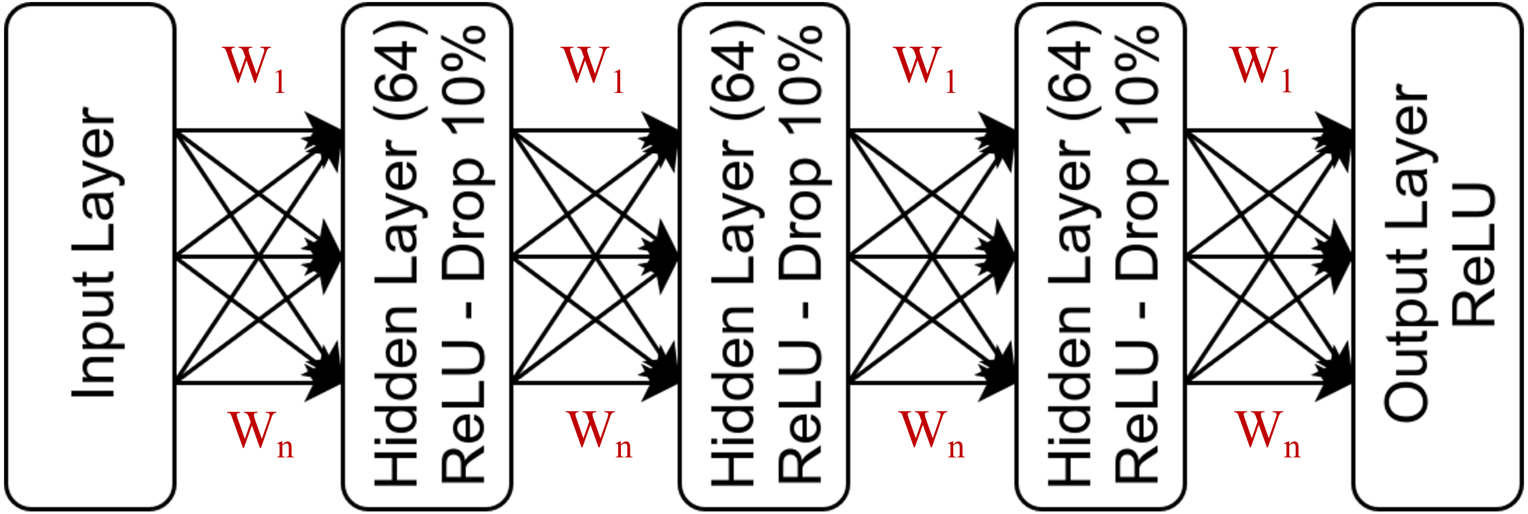
Time (s)	$TPut_1$	$TPut_2$	...	$TPut_{64}$
0	0.015	0.039	...	0.0
1	0.016	0.035	...	0.0
...	...	...	...	...
2399	0.009	0.027	...	0.0

Neural Network



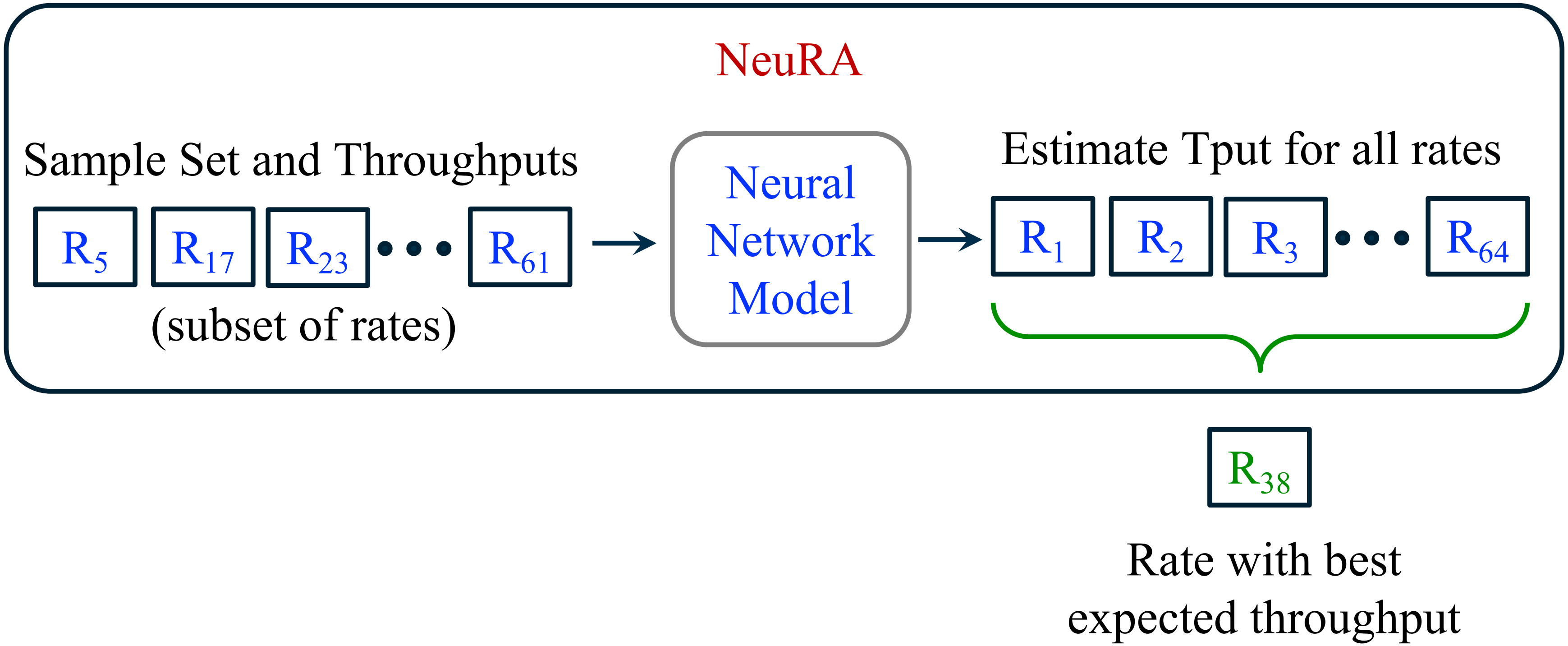
Input: Fixed set of rates and tputs, Output: expected tput of all rates

NeuRA's Resulting Neural Network Model



Weights on edges determined during training

NeuRA



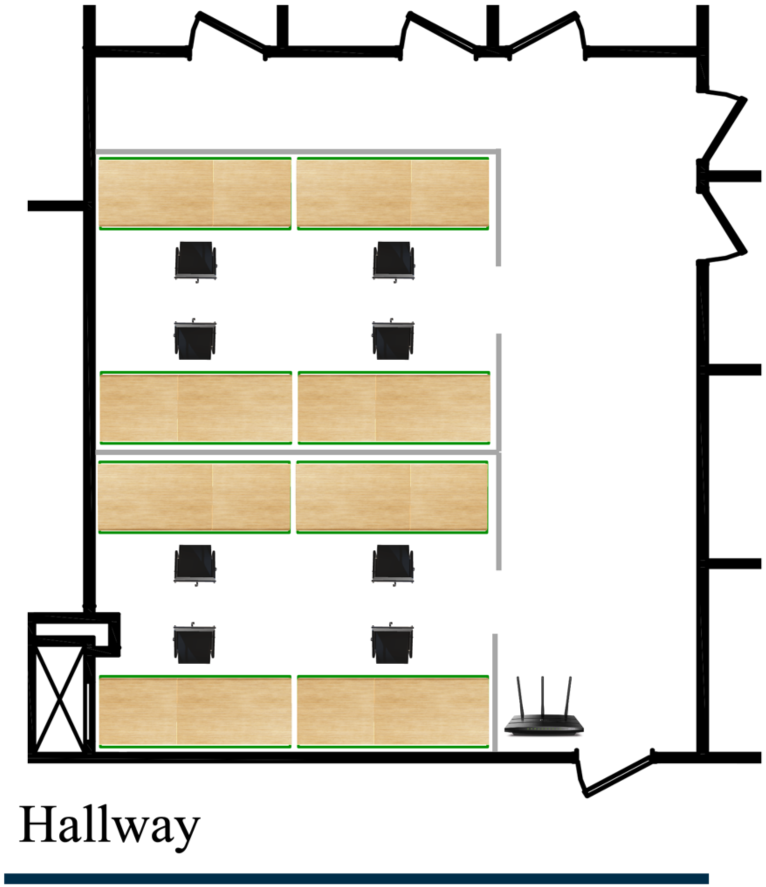
NeuRA
Sample Set and Throughputs
R5 R17 R23 • • • R61
(subset of rates)
Neural Network Model
Estimate Tput for all rates
R1 R2 R3 • • • R64
R38
Rate with best expected throughput

Evaluation Methodology

- Two separate models: 2.4 GHz and 5 GHz
- Two separate sets of traces for each: training and testing (evaluation)

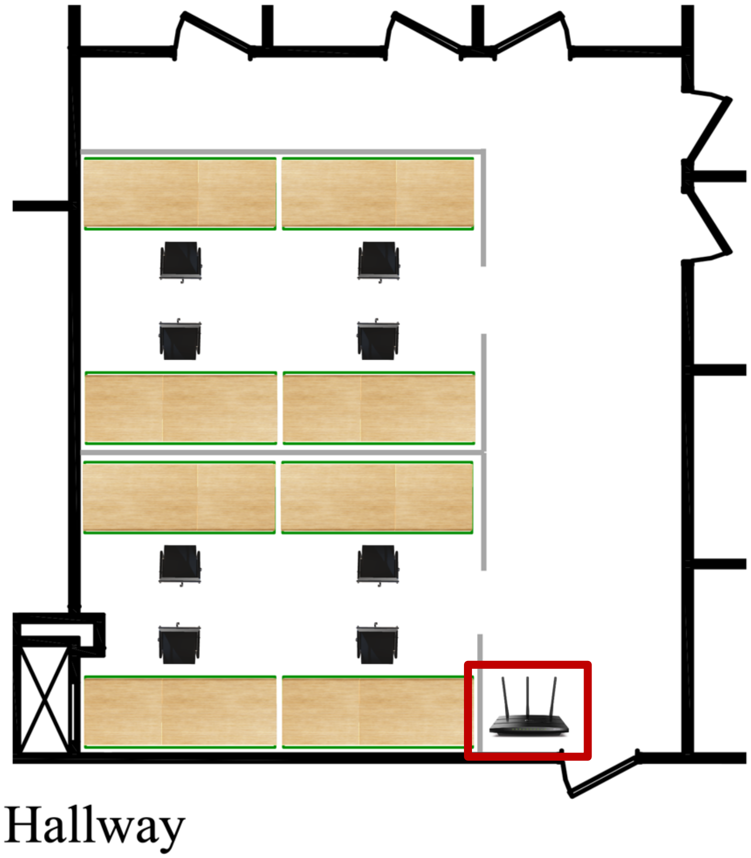
Config	Spectrum	# Streams	Channel Width	# Rates	Channel Condition
A	2.4 GHz	2	20 MHz	32	Congested
B	5 GHz	2	40 MHz	64	Unoccupied

Scenarios for Trace Collection



Office Environment
Graduate student offices / lab

Scenarios for Trace Collection

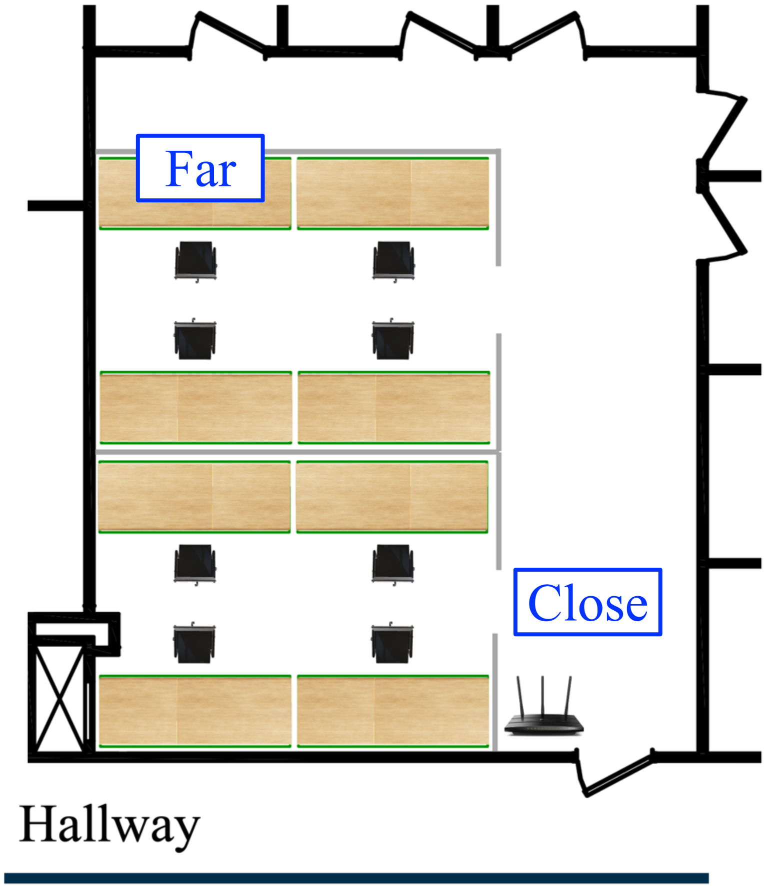


Office Environment
Graduate student offices / lab

Access Point
PC with ath9k WiFi (802.11n)

TP-Link WDN4800

Scenarios for Trace Collection: Training Data



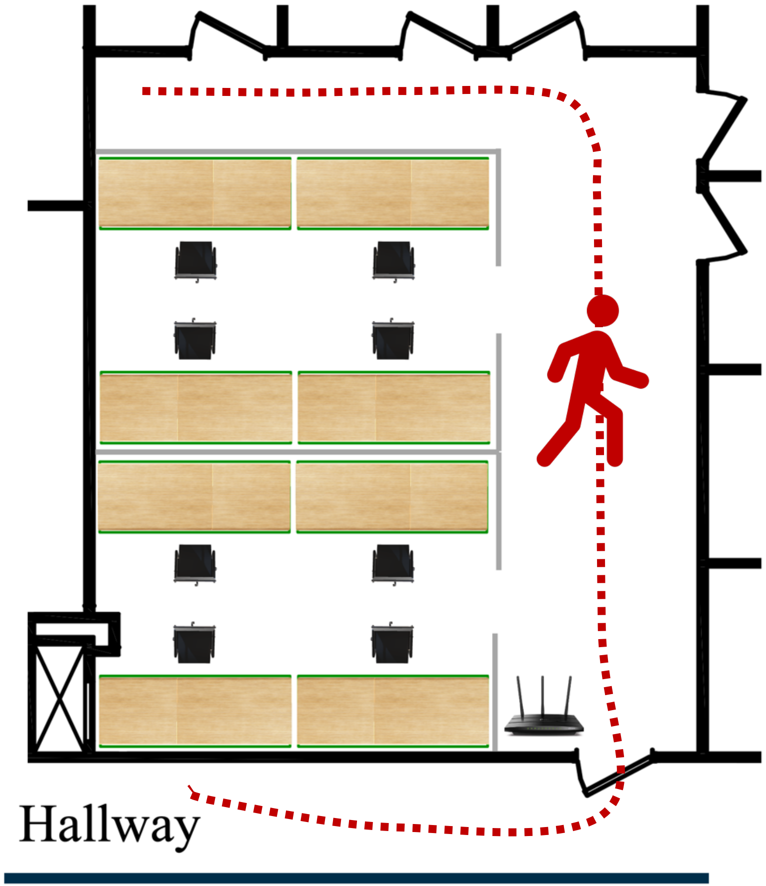
Stationary:
Close to AP ~1 m
Far from AP ~10 m

Laptop with TL-WDN4200 USB device

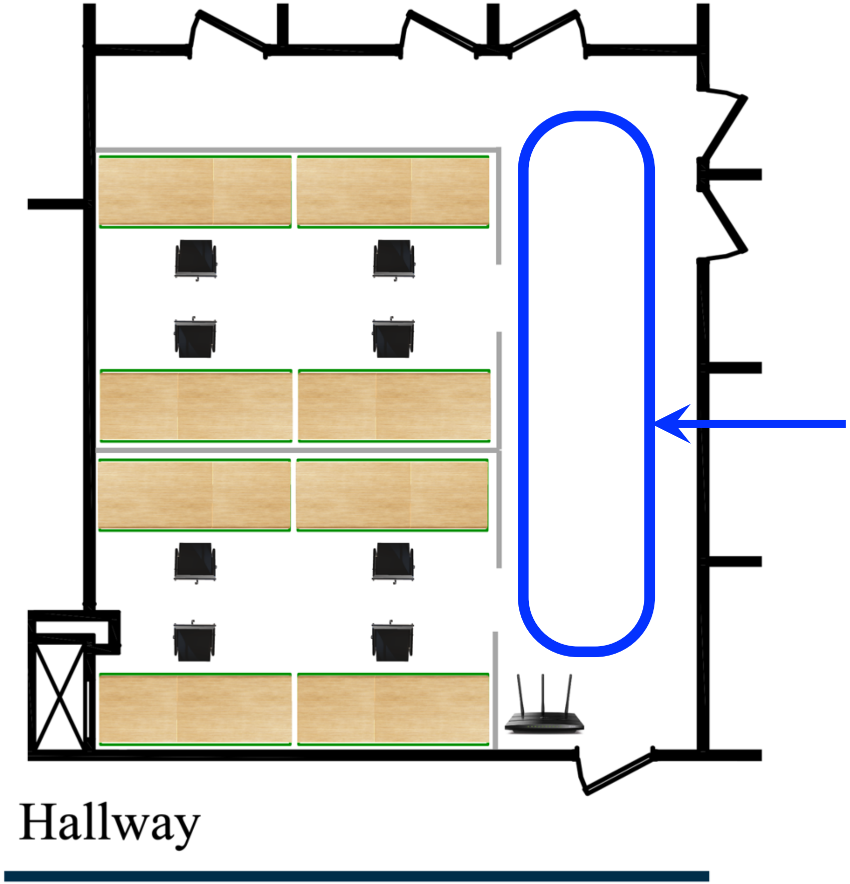
Samsung Galaxy Note 5

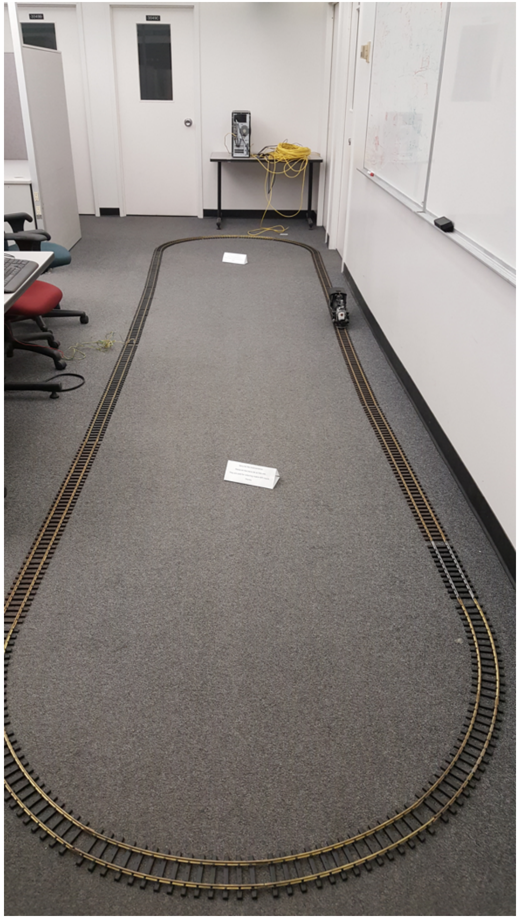
Scenarios for Trace Collection: Training Data

Mobile: Walking



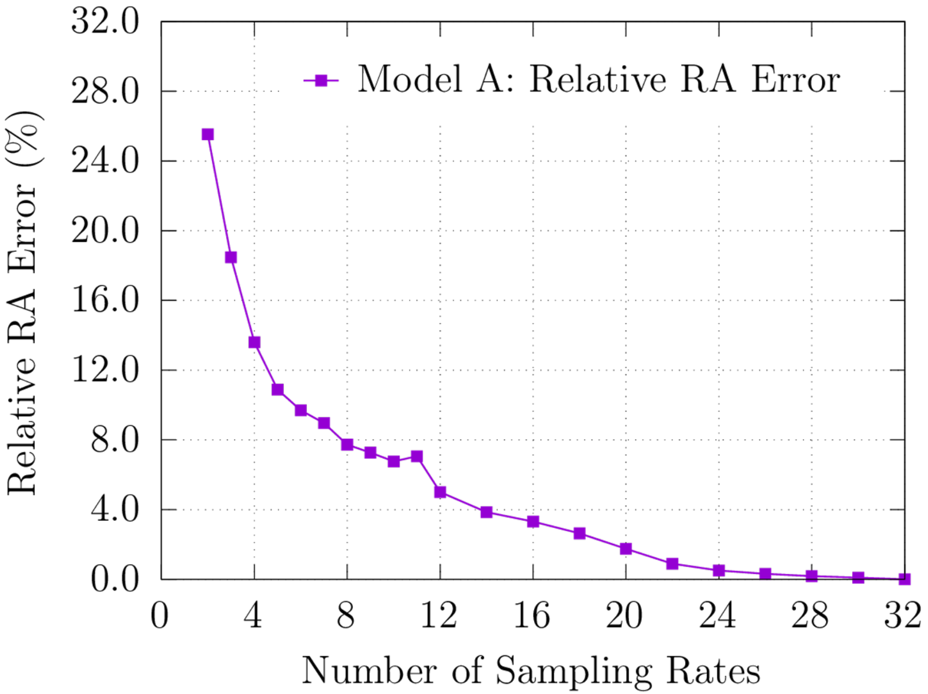
Scenarios for Trace Collection: Training Data





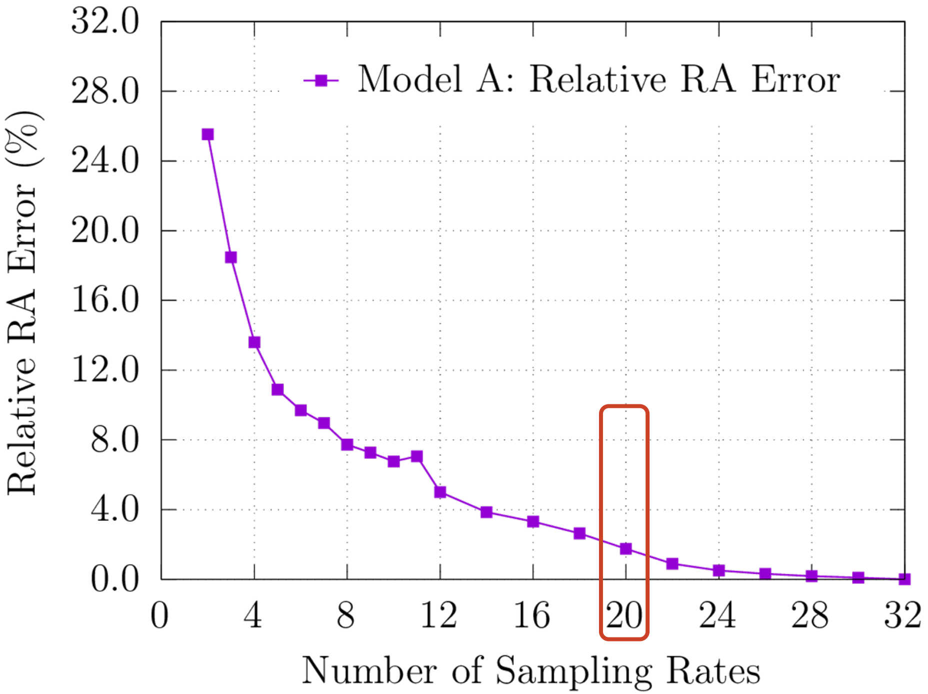
Mobile: Toy Train
Fast and slow

Relative Rate Adaptation Error



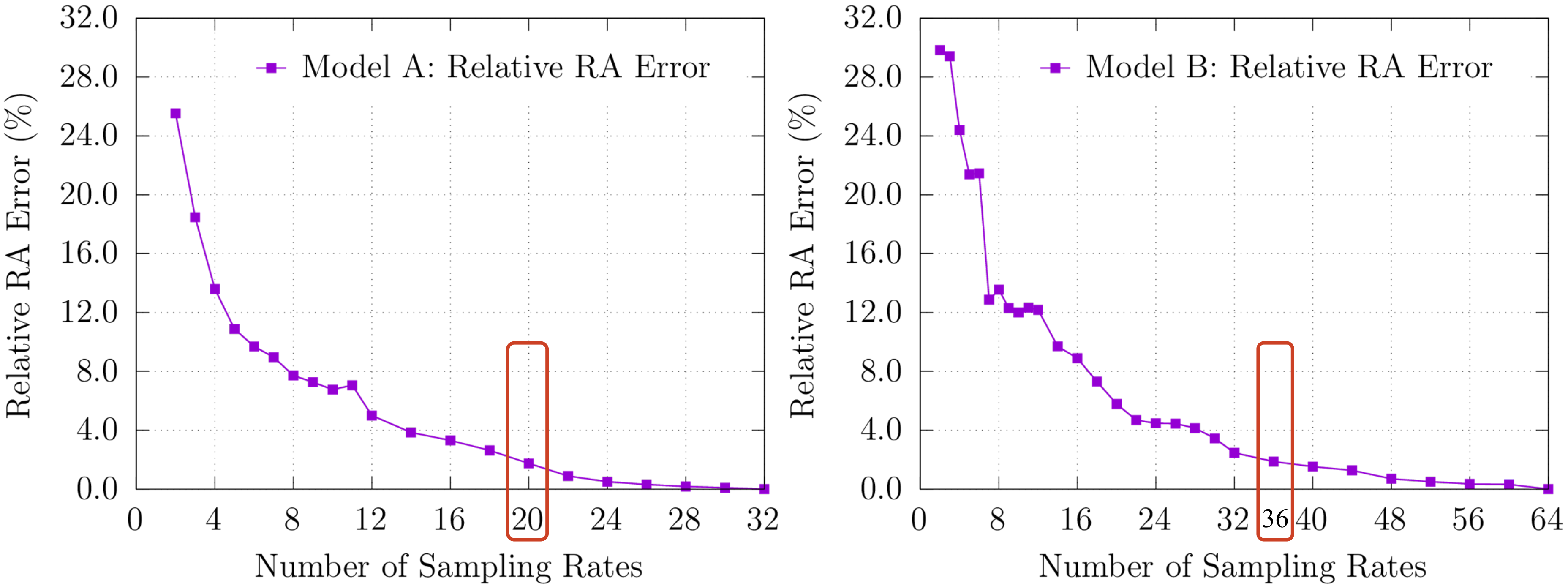
- Rate adaptation using model (avg. error on testing dataset)

Relative Rate Adaptation Error



- Rate adaptation using model (avg. error on testing dataset)

Relative Rate Adaptation Error



- Rate adaptation using model (avg. error on testing dataset)

Evaluation: Algorithms

Rate Adaptation Algorithms

- Minstrel HT
- NeuRA
- Intel iwl-mvm-rs
- Minstrel HT w/o LGI Sampling

Frame Aggregation Algorithms

- Minstrel HT + PNOFA
- Minstrel HT + OSOFA

Both

- STRALE
- Offline Statistically Optimal

Evaluation: Algorithms

Rate Adaptation Algorithms

- Minstrel HT
- NeuRA
- Intel iwl-mvm-rs
- Minstrel HT w/o LGI Sampling

Evaluation: Algorithms

Rate Adaptation Algorithms

- Minstrel HT
- NeuRA
- Intel iwl-mvm-rs
- Minstrel HT w/o LGI Sampling

- Most widely used algorithm
- 100's of millions of devices
- In Linux
- Use as a basis for comparison

Evaluation: Algorithms

Rate Adaptation Algorithms

- Minstrel HT
- NeuRA
- Intel iwl-mvm-rs
- Minstrel HT w/o LGI Sampling

- From this work

Evaluation: Algorithms

Rate Adaptation Algorithms

- Minstrel HT
- NeuRA
- Intel iwl-mvm-rs
- Minstrel HT w/o LGI Sampling

- Another practical widely used alg
- Used in recent Intel chipsets
- Described in and code ported from

  [Grünblatt, et al. MSWiM, 2019]

Evaluation: Algorithms

Rate Adaptation Algorithms

- Minstrel HT
- NeuRA
- Intel iwl-mvm-rs
- Minstrel HT w/o LGI Sampling

- From "relationships" paper

  [Abedi and Brecht, MSWiM, 2016]

- Proof of concept for relationships
- Samples SGI rates, estimates LGI

Evaluation: Algorithms

Rate Adaptation Algorithms

- Minstrel HT
- NeuRA
- Intel iwl-mvm-rs
- Minstrel HT w/o LGI Sampling

Frame Aggregation: all maximize number of frames
Except: NeuRA in 5 GHz (PNOFA)

Evaluation: Algorithms

- Practical Near Optimal Frame Aggregation

- Offline Statistically Optimal Frame Aggregation

PNOFA paper
[Abedi et al, MSWiM, 2020]

Frame Aggregation Algorithms

- Minstrel HT + PNOFA
- Minstrel HT + OSOFA

Evaluation: Algorithms

- Adjusts Frame Length and Rate

  [Byeon et al. INFOCOM 2017]

Both

- STRALE
- Offline Statistically Optimal

Evaluation: Algorithms

Both

- STRALE
- Offline Statistically Optimal

Offline Statistically Optimal: FA and RA Algorithm

Key contribution

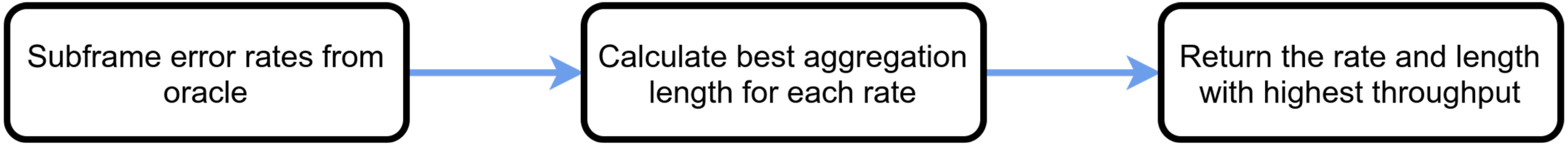
Statistically optimal frame length and rate

Upper bound on throughput of practical RA and FA algorithms

Previously weak understanding of how well algorithms were doing

- Only relative to each other
- No idea of how much room there is for improvement
- When do we stop creating new algorithms?

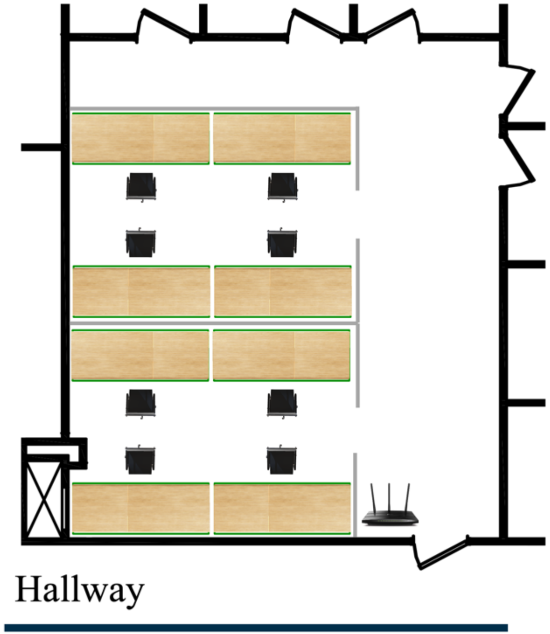
Offline Statistically Optimal: FA and RA Algorithm



Trace-Based Evaluation

- T-SIMn: trace-driven simulator [Abedi et al. MSWiM, 2016]
- Trace-based: all algorithms see the same channel conditions
  Differences are due to algorithms not changes in the channel
- Can implement Offline Statistically Optimal (look ahead in trace)

Different Traces and Scenarios for Testing



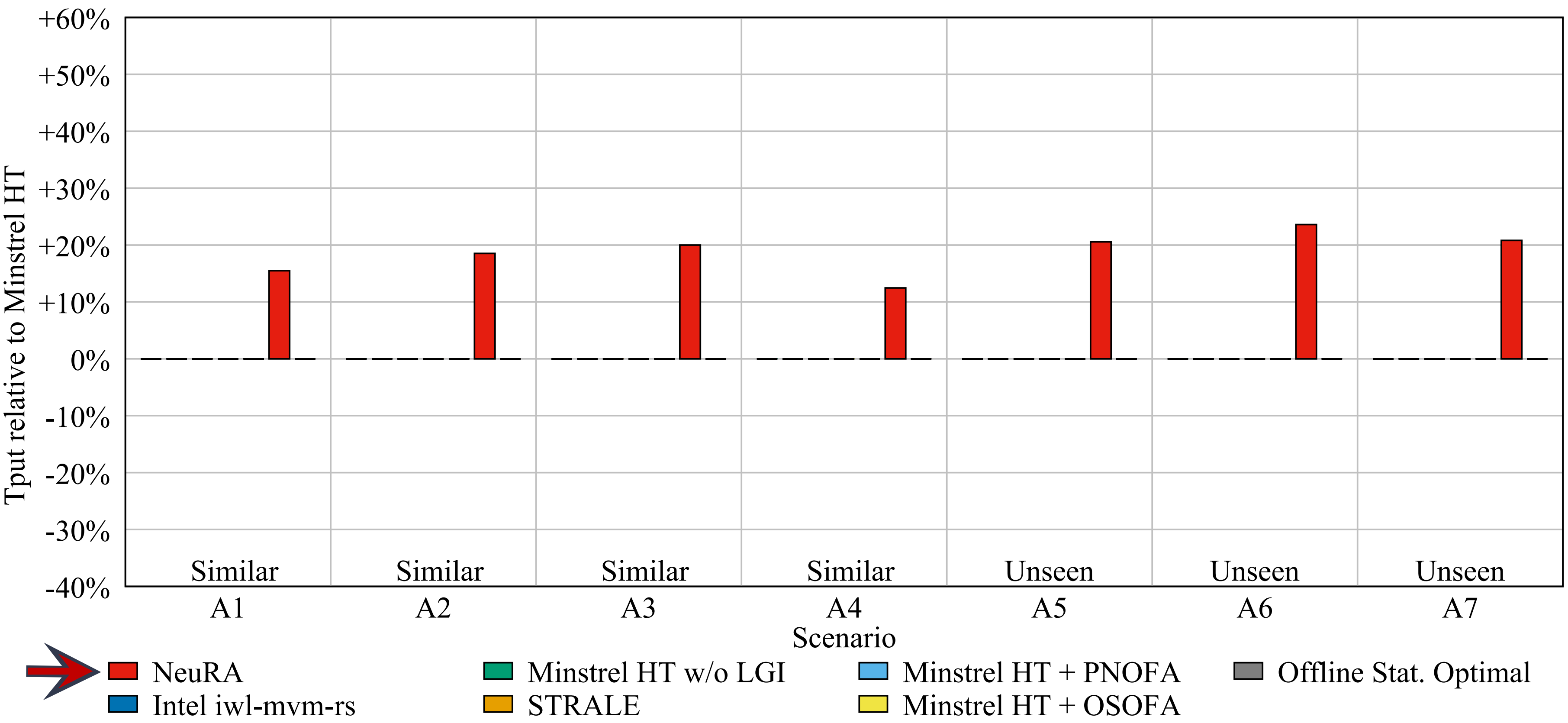
- All new traces
- Some similar setting as training
- Previously unseen scenarios
  - 2 new devices
  - New mobility patterns (extreme movement)
- 7 scenarios for each model
- 5 - 20 minutes each
- Stationary and mobile

Intel 8265 laptop WiFi card Walking

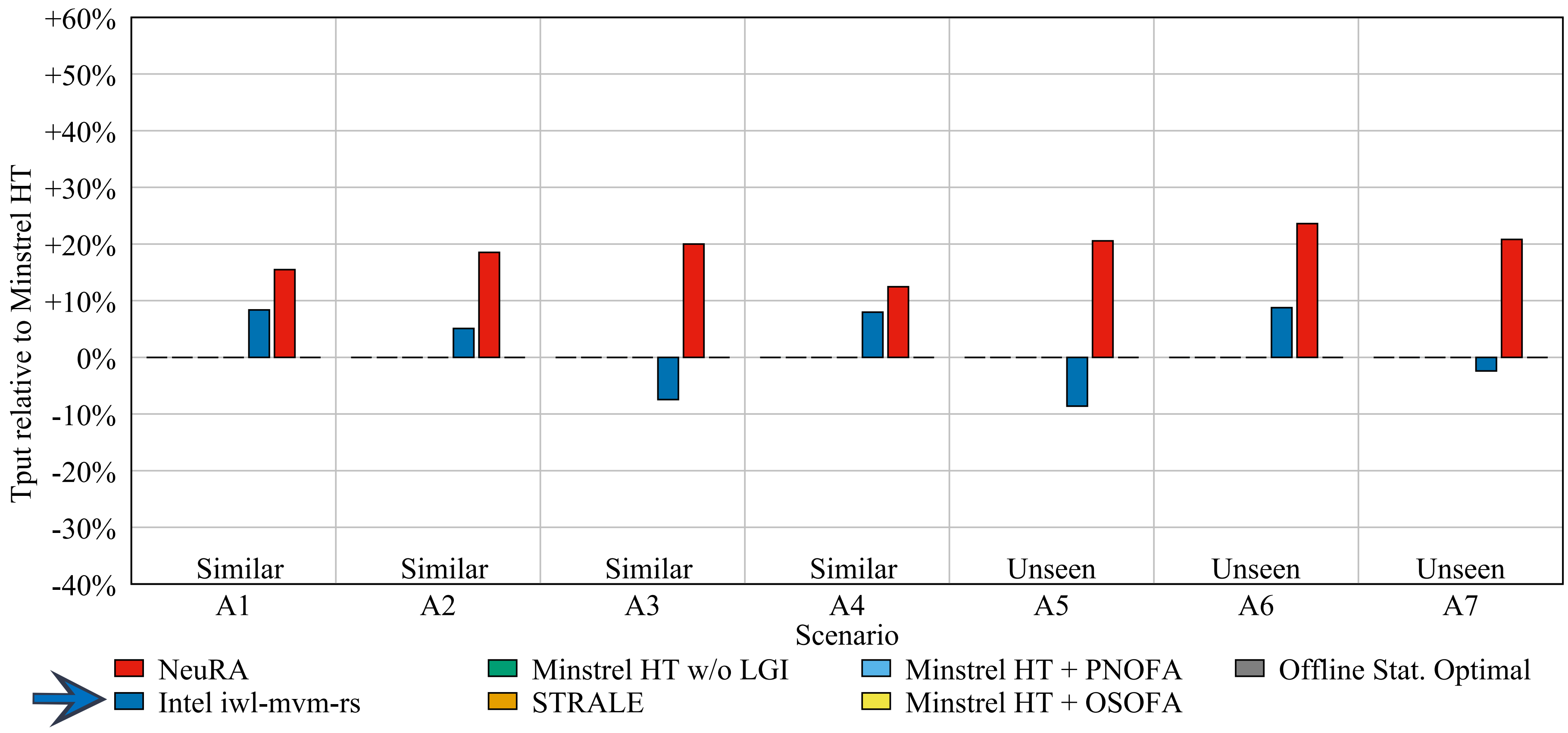
Huawei P20 (EML-L09C) Walking and Stationary

Traces from WiFi experiments collected using real-world conditions

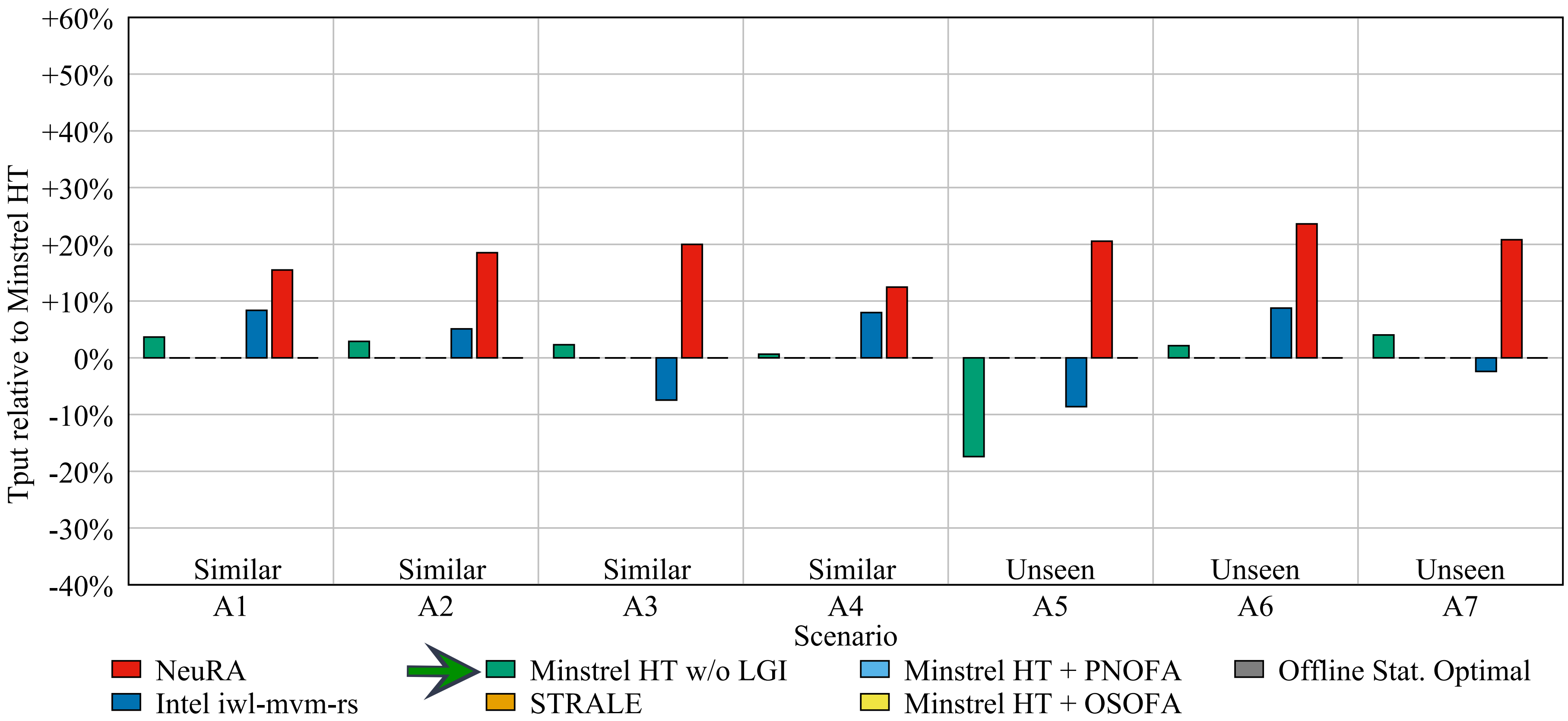
Trace-Based Evaluation (Model A, 2.4 GHz)



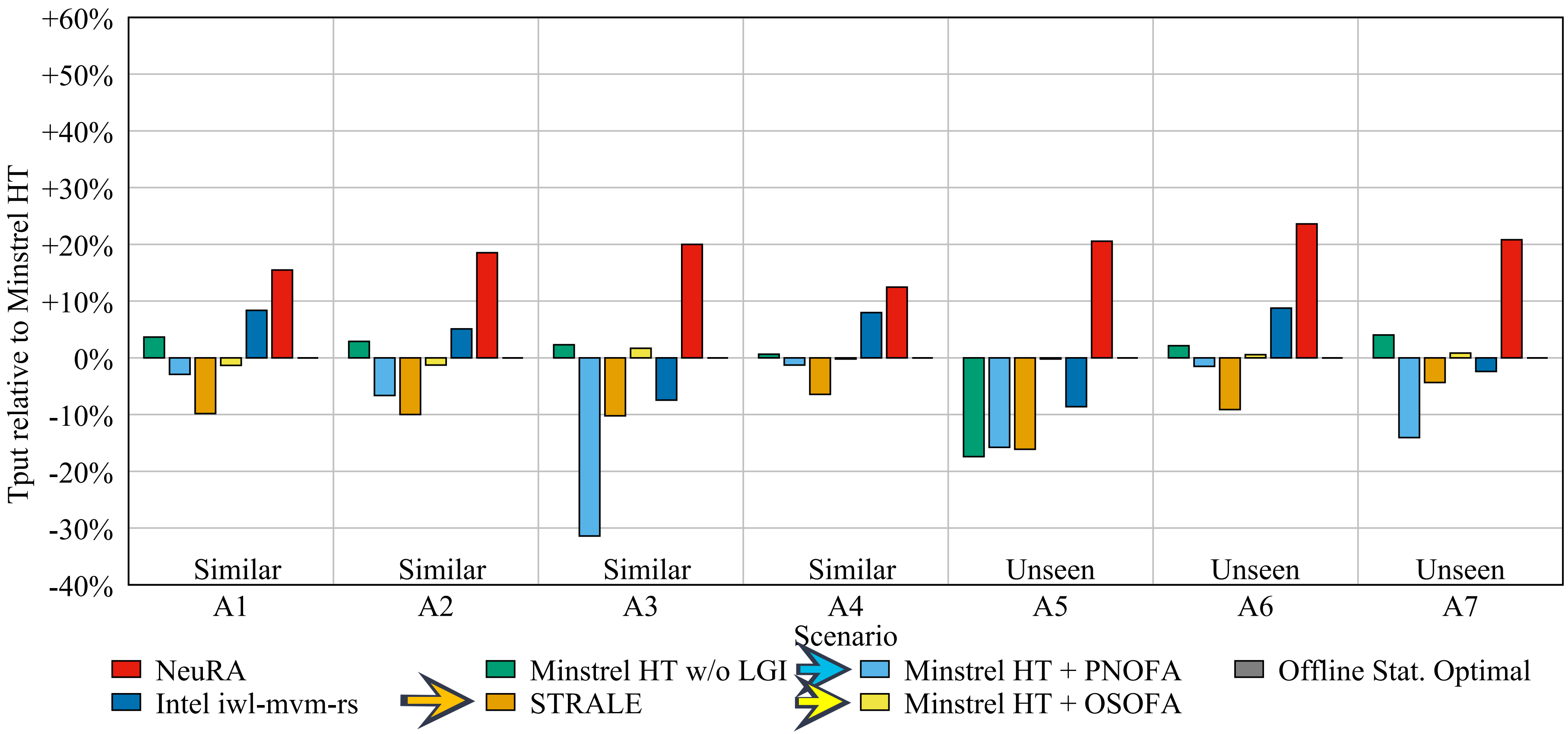
Trace-Based Evaluation (Model A, 2.4 GHz)



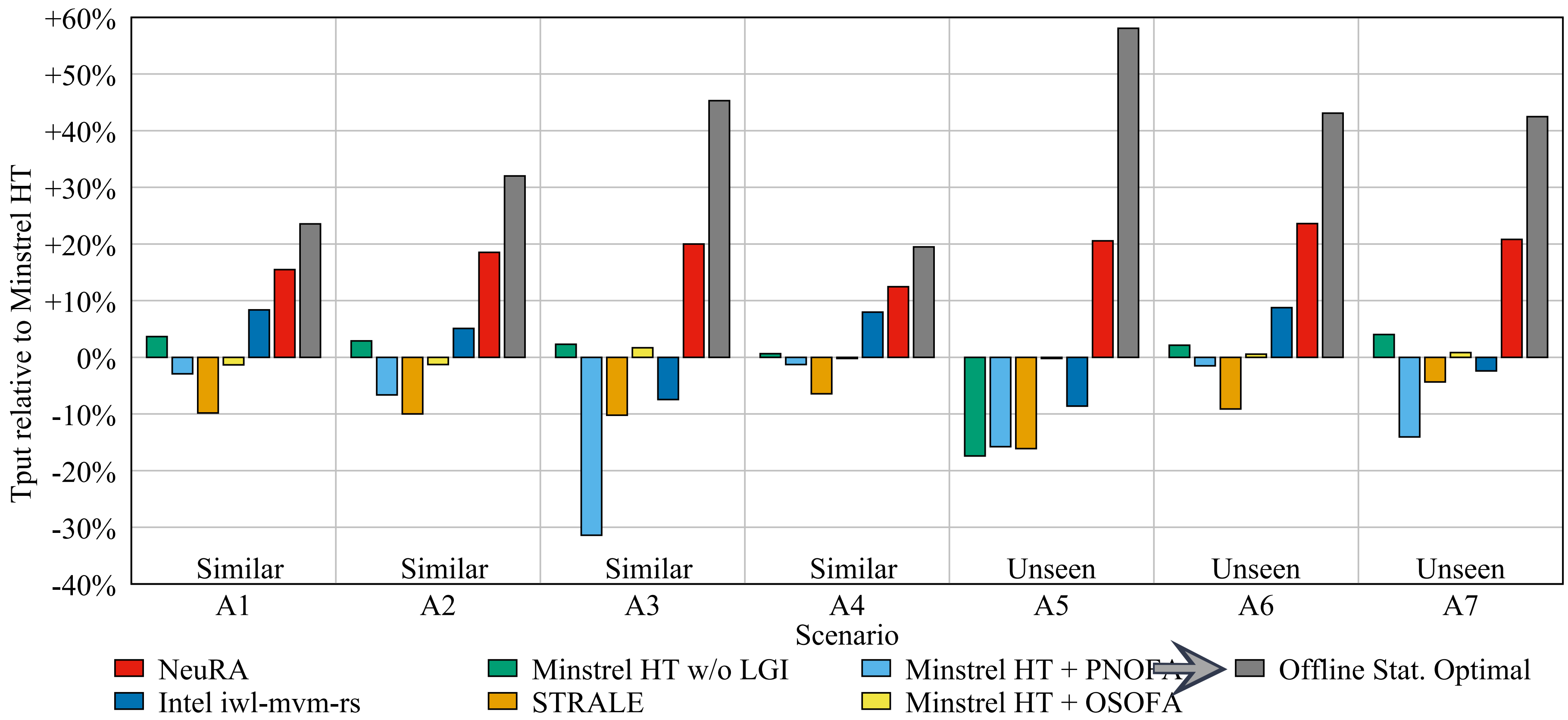
Trace-Based Evaluation (Model A, 2.4 GHz)



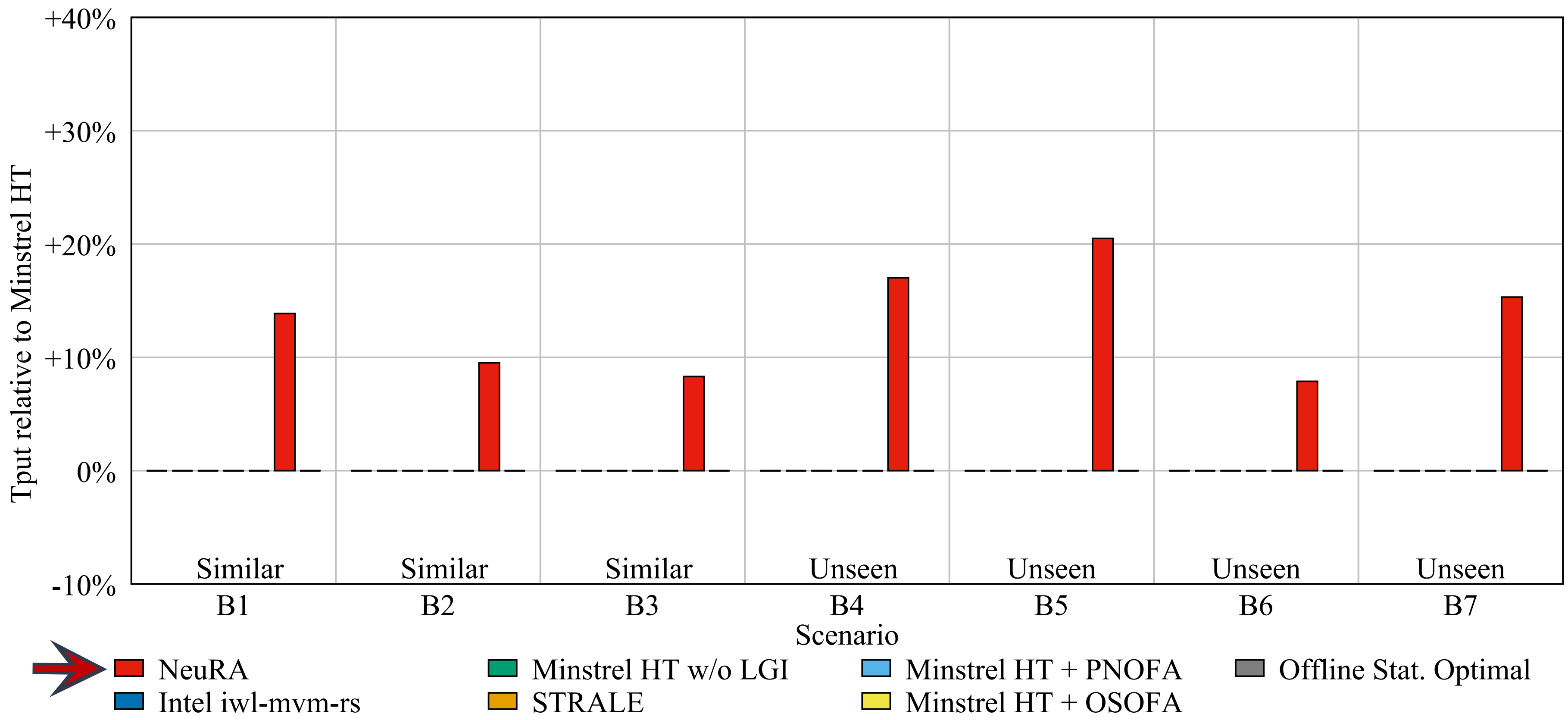
Trace-Based Evaluation (Model A, 2.4 GHz)



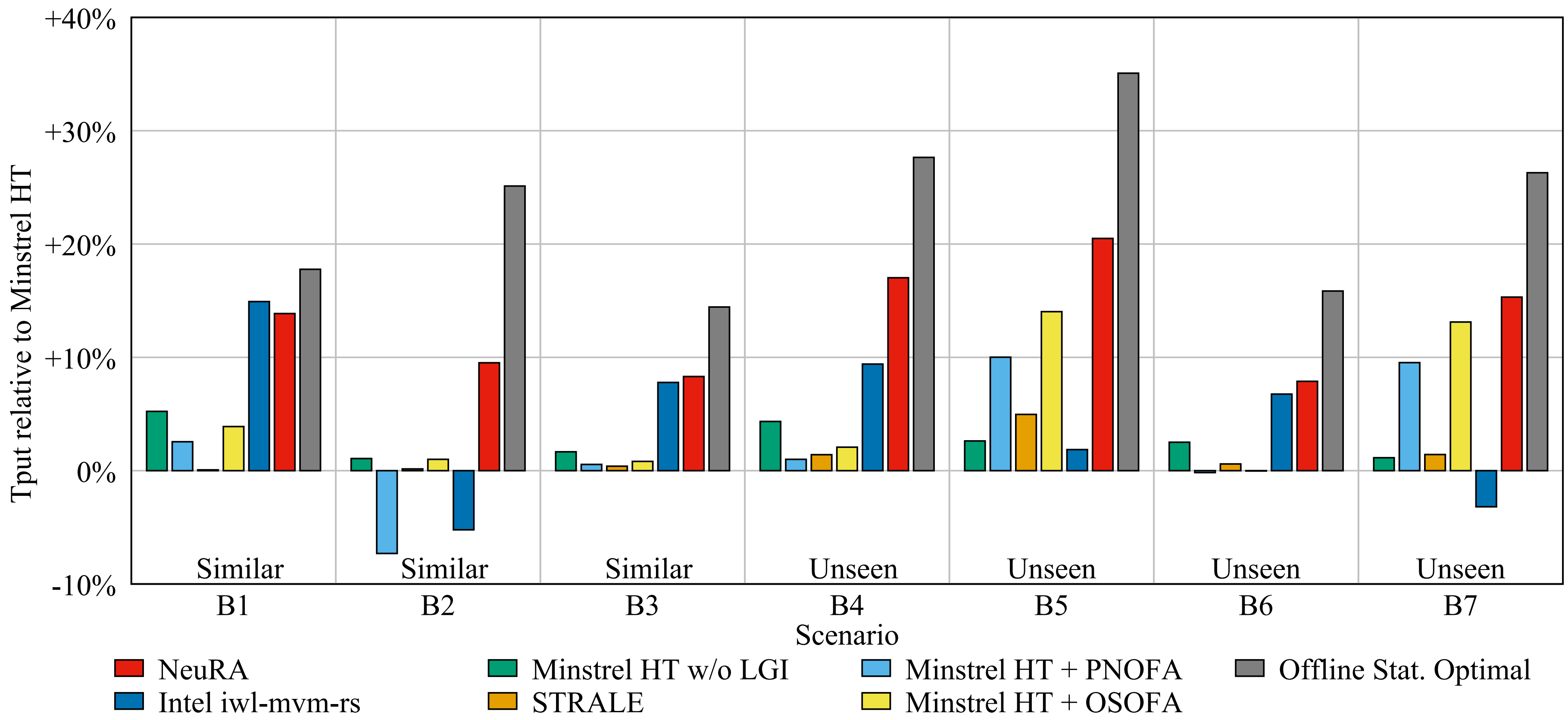
Trace-Based Evaluation (Model A, 2.4 GHz)



Trace-Based Evaluation (Model B, 5 GHz)



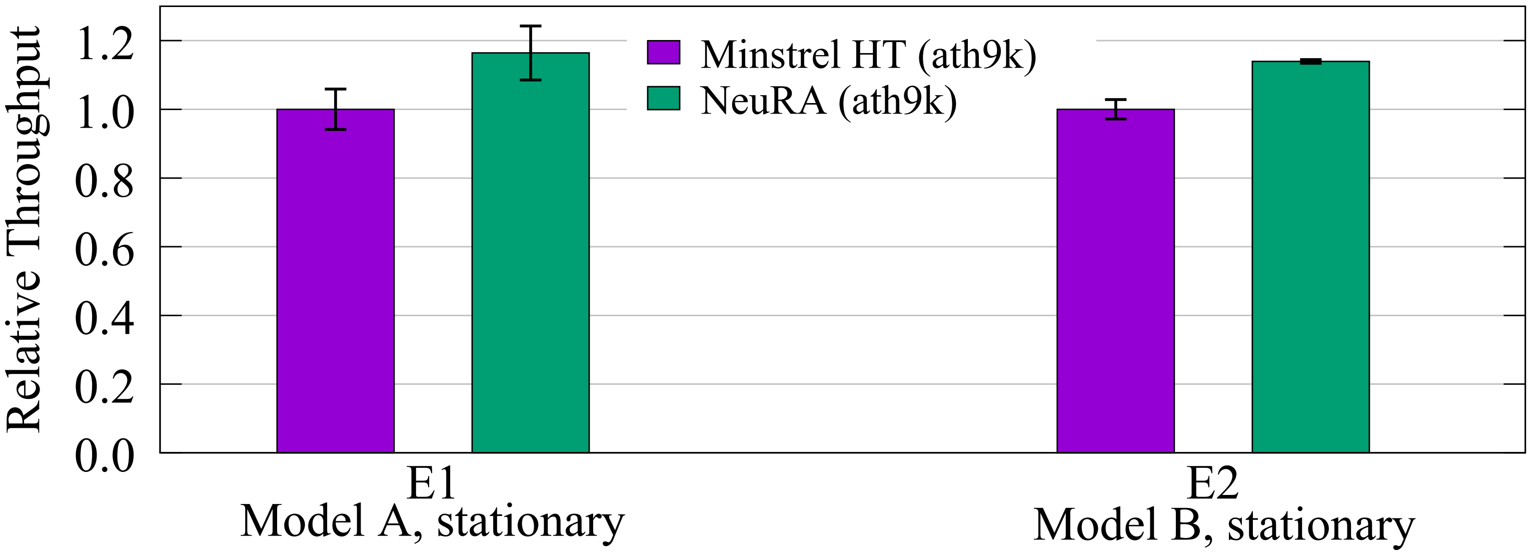
Trace-Based Evaluation (Model B, 5 GHz)



Summary of Trace-Driven Evaluation

- NeuRA
  - Up to 24% higher tput than Minstrel HT (16% on average)
  - Up to 32% higher tput than Intel iwl-mvm-rs (13% on average)
  - Reduces gap between Minstrel HT and upper bound by half
  - Remaining gap not overly large

Real-World Prototype (in Linux)



- CPU: 20% of a 800 MHz core

Conclusions

NeuRA

- Use predictions from neural network model, reduce sampling overhead
- Generalized model improves throughput on unseen scenarios
- Low processing overhead to improve throughput in real world
- Potentially greater impact with more rates (802.11ax: up to 768!)

Offline Statistically Optimal Algorithm

- Obtain upper bound on throughput (NeuRA is not that far from opt)

Simulator, Traces, Algorithms to be made available
https://cs.uwaterloo.ca/~brecht/neura