A New Formulation of Neural Data Prefetching

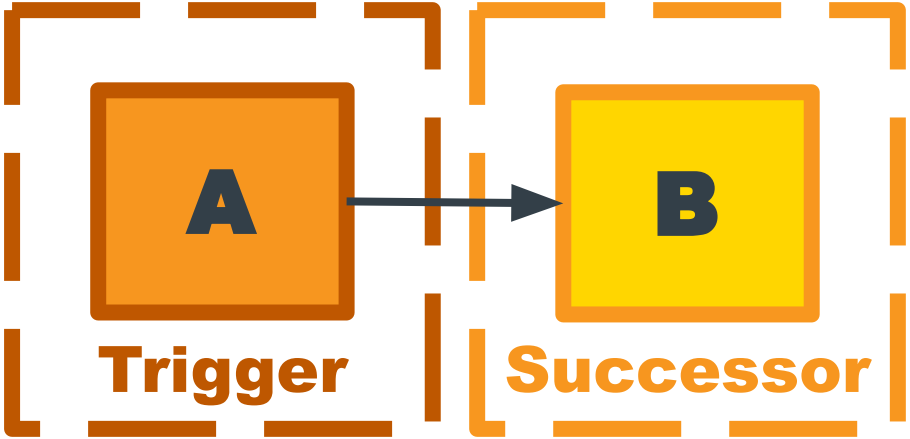
Quang Duong [1] Akanksha Jain [2] Calvin Lin [1]

[1] The University of Texas at Austin [2] Google

TEXAS
The University of Texas at Austin

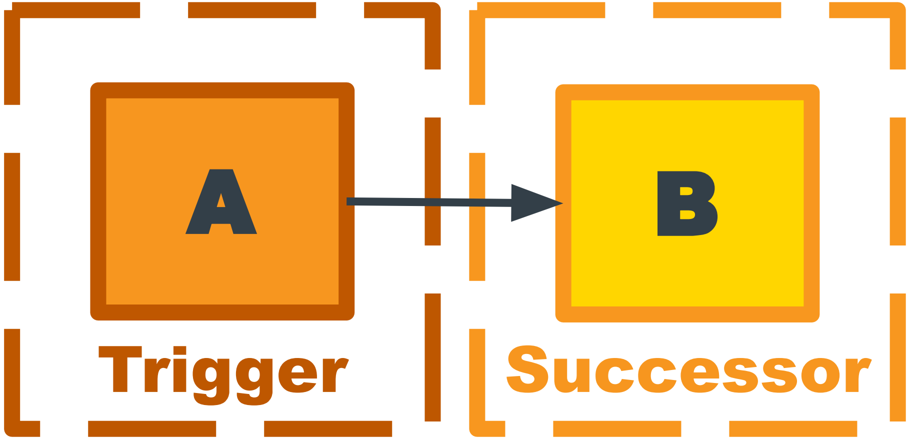
Temporal Prefetching [ISCA 1997]

- **Powerful Idea**
  - Correlate a trigger address A to successor address B



Temporal Prefetching [ISCA 1997]

- **Powerful Idea**
  - Correlate a trigger address **A** to successor address **B**
  - Capable of prefetching <u>any repeated stream of accesses</u>

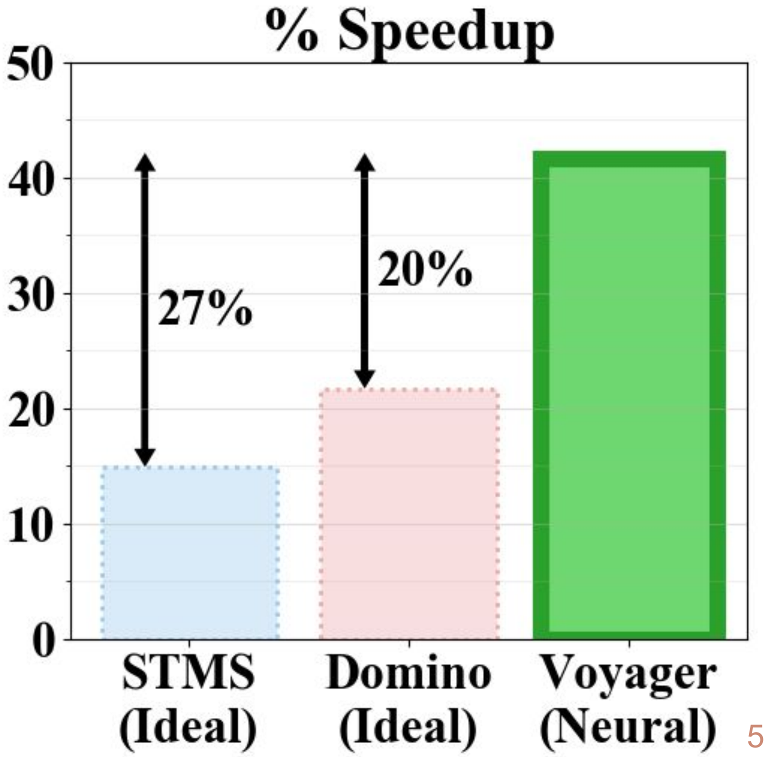


Voyager [ASPLOS 2021]

- First neural temporal prefetcher

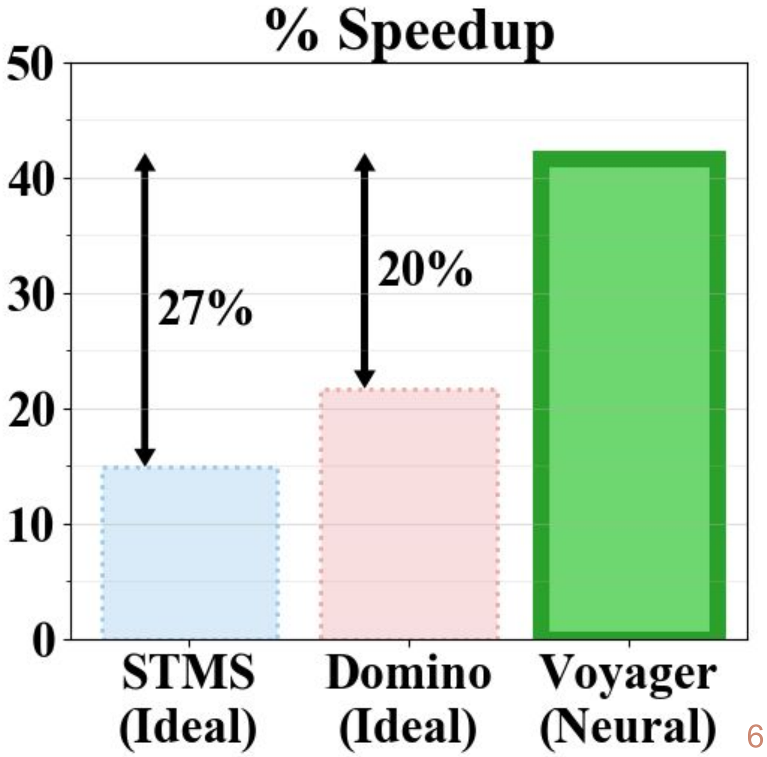
Voyager [ASPLOS 2021]

- First neural temporal prefetcher
- Outperforms **idealized** temporal prefetchers



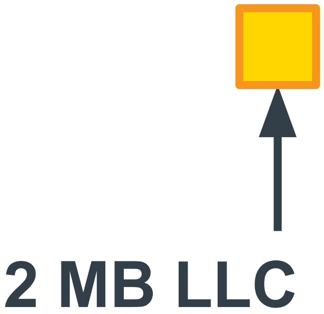
Voyager [ASPLOS 2021]

- First neural temporal prefetcher
- Outperforms **idealized** temporal prefetchers
- **Completely impractical**
  - Designed as a limit study



Voyager’s Cost

Storage



Voyager’s Cost

Storage

Voyager’s Cost

Storage

Latency

Latency between LLC Accesses

Voyager's Cost

Storage

Voyager
114 MB
57x

Latency

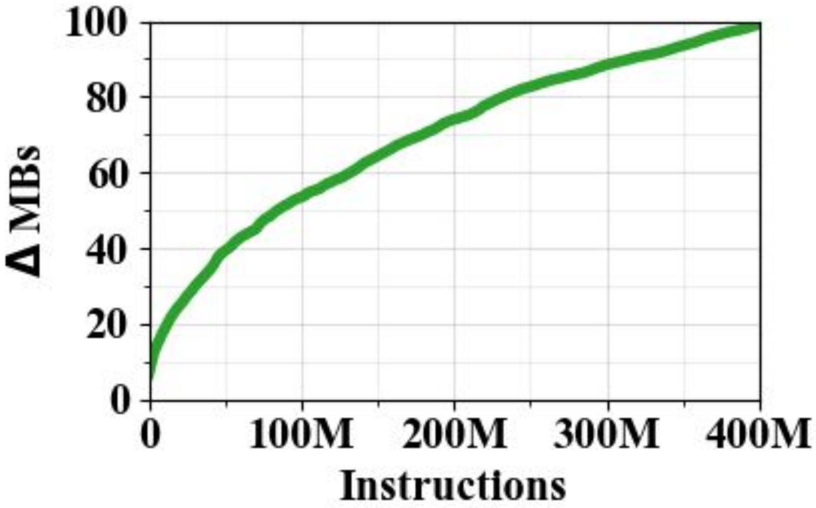
Voyager
81M FLOPs
200x

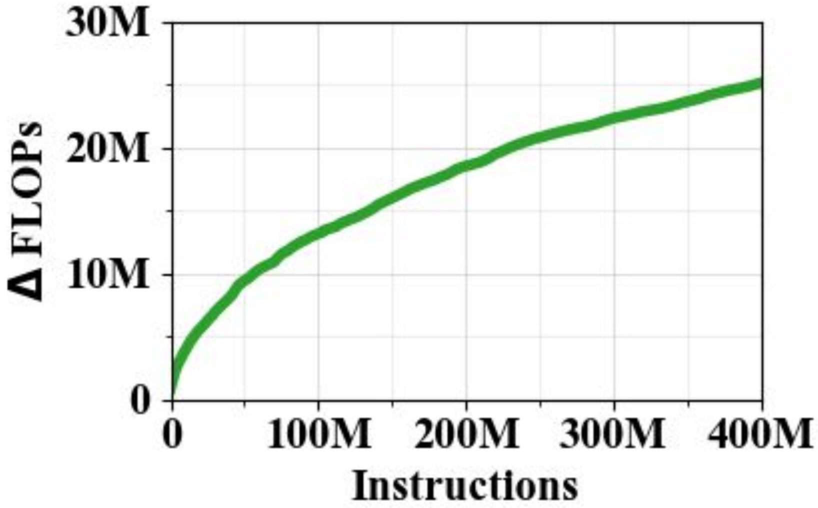
Standard ML Techniques to Reduce Cost

- **Model Compression Techniques**
  - Quantization
  - Pruning
  - Knowledge Distillation
  - Parameter Sharing
  - (Un)structured Sparsity
  - Ephemeral Sparsity
  - Dropout
  - Operator Factorization
  - Regularization
  - Neural Architecture Search
  - Low Rank Adaptation
  - Mixture of Experts

Standard ML Techniques NOT ENOUGH

- Storage and latency **grow with the memory footprint**





The Formulation is The Problem

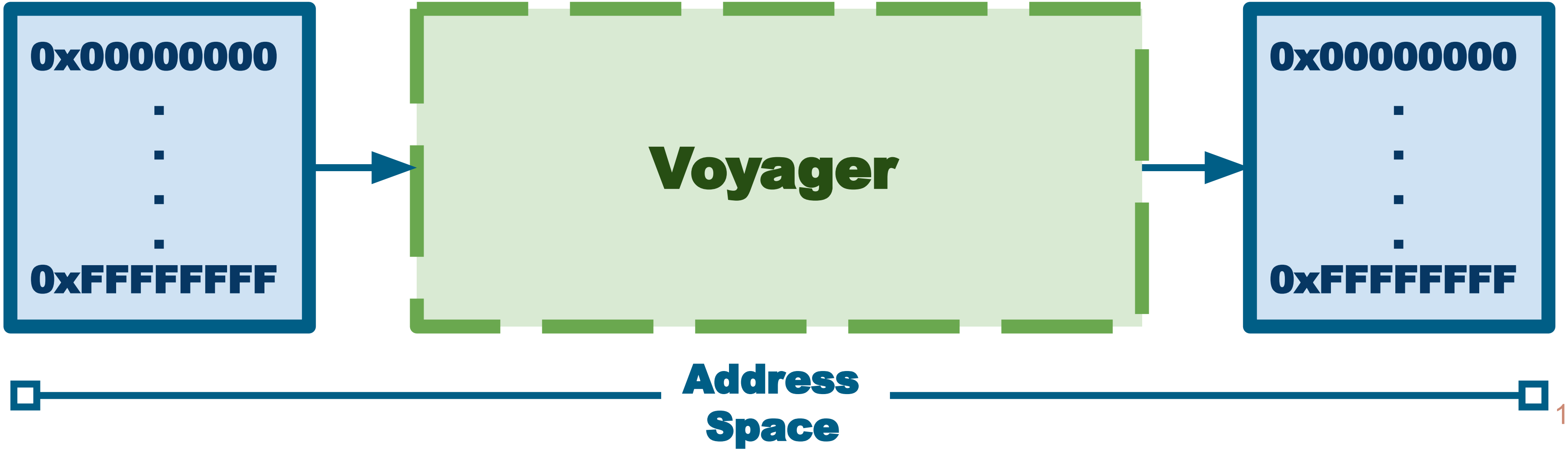
- Addresses are **not suitable as neural inputs / outputs**

The Formulation is The Problem

- Addresses are **not suitable as neural inputs / outputs**
  - Grows with the footprint

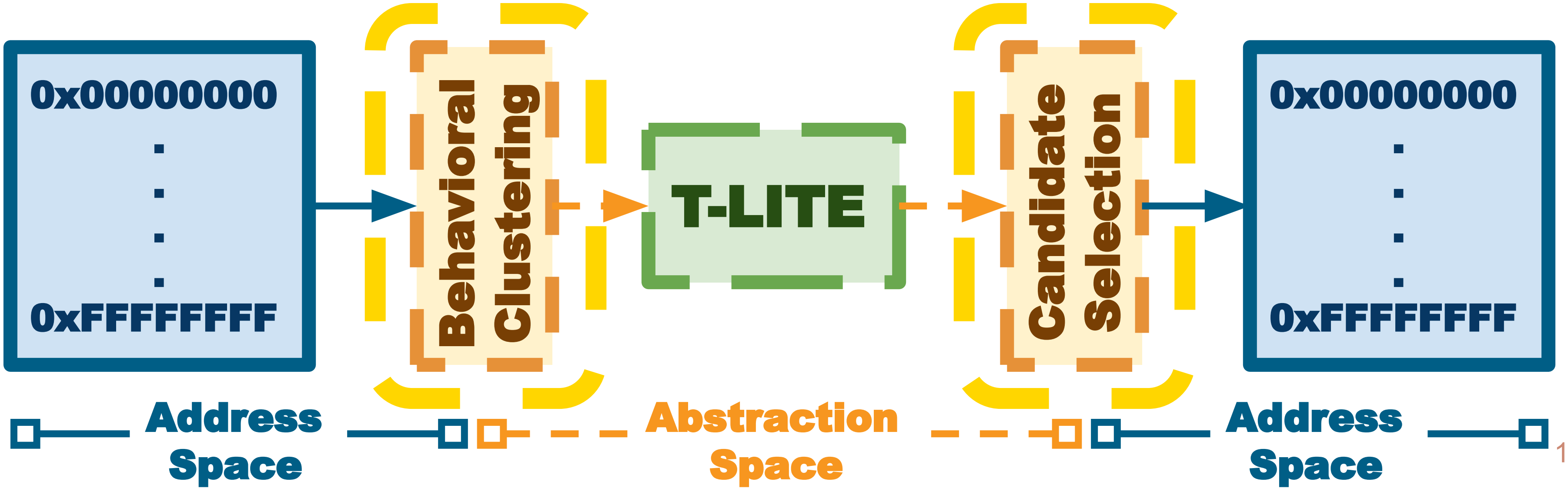
The Formulation is The Problem

- Addresses are **not suitable as neural inputs / outputs**
  - Grows with the footprint + correlations useless across runs



Our New Formulation

- Hide addresses by inserting layers of abstraction



Voyager’s Cost vs Our Approach

Storage

Voyager's Cost vs Our Approach

Storage

Voyager's Cost vs Our Approach

Latency

Voyager
81M FLOPs
200x

Voyager's Cost vs Our Approach

Latency

Overview

Overview

- **Twilight**
  - Significantly reduced cost (**10.8× smaller + 988× faster**)
  - Still impractical
    - Compare against Voyager [ASPLOS 21]

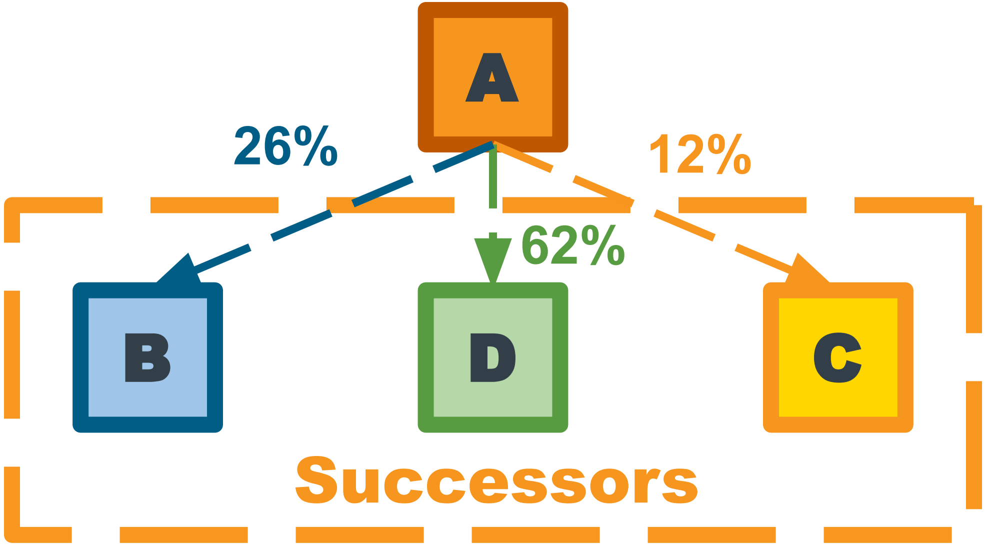
Overview

- Twilight
  - Significantly reduced cost (**10.8× smaller + 988× faster**)
  - Still impractical
    - Compare against Voyager [ASPLOS 21]
- Twilight-LITE (T-LITE for short)
  - Efficiency-focused derivation of Twilight
  - Near-practical (**142× smaller + 1421× faster**)
    - Compare against Triage [MICRO 19]

Twilight

Key Insight: Sparse Connectivity

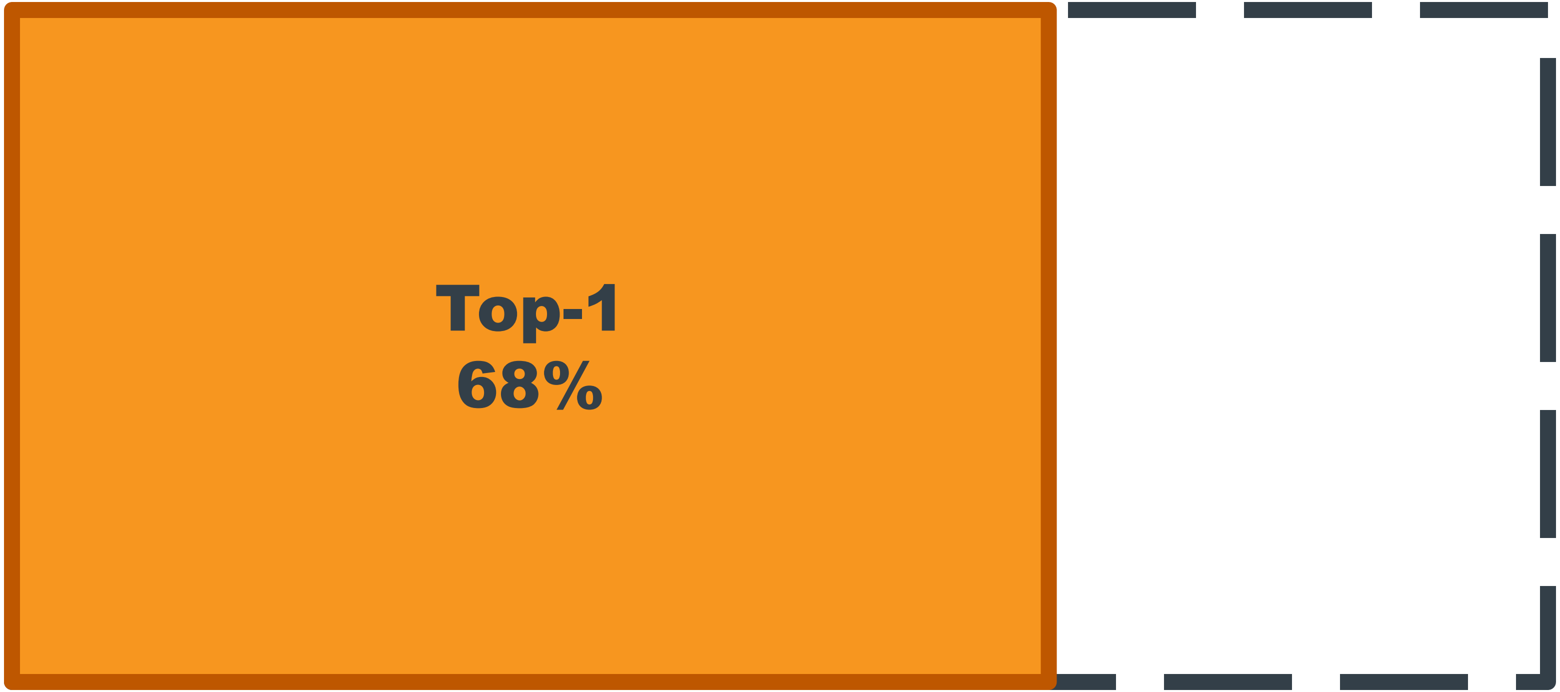
- Every address **A** is often followed by just a handful of successor addresses **{B, C, D, …}**



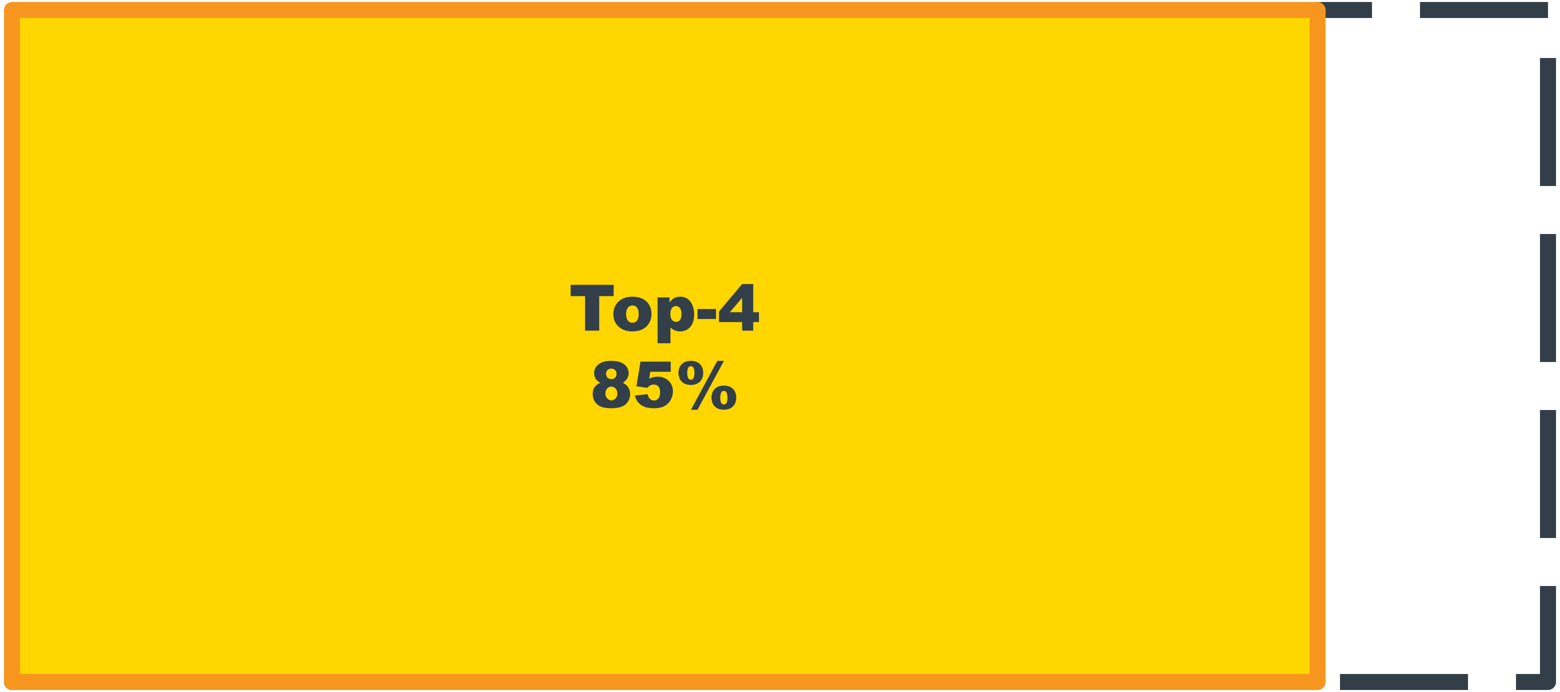
Sparse Connectivity in Practice

Temporal Prefetching Opportunity

Sparse Connectivity in Practice



Sparse Connectivity in Practice



Sparse Connectivity in Practice

Top-20
94%

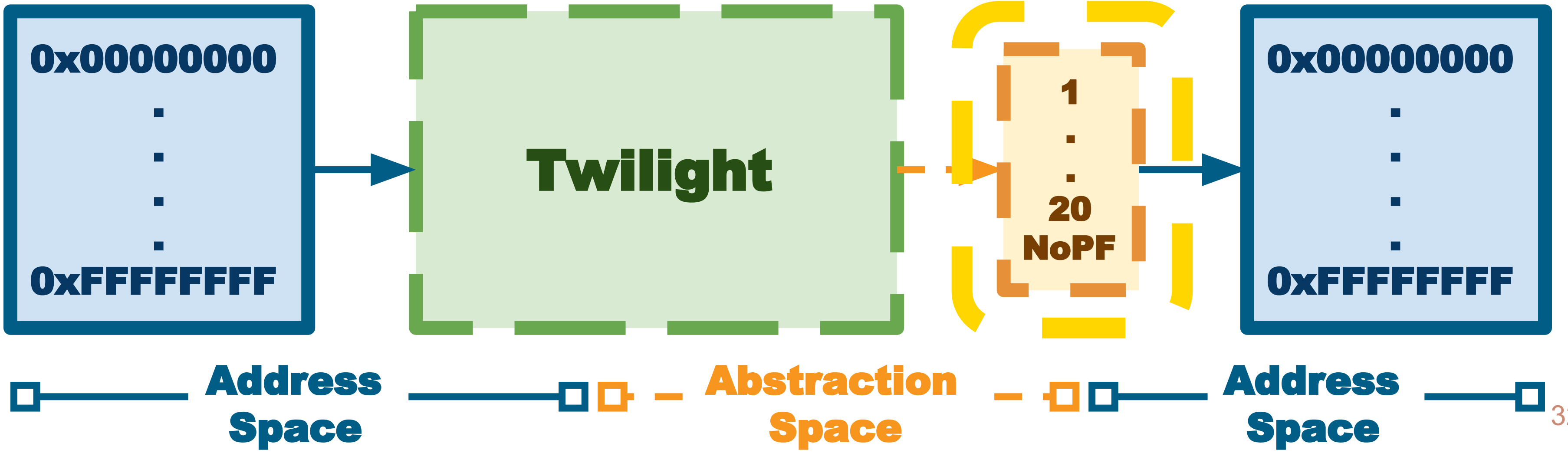
Sparse Connectivity in Practice

Candidate Selection

- Given the top-N most frequent successors **{B, C, D, ...}**, select which of them to prefetch (or to not prefetch)

Candidate Selection

- Given the top-N most frequent successors **{B, C, D, ...}**, select which of them to prefetch (or to not prefetch)



Twilight-LITE (T-LITE)

Problem

- Voyager has a neural encoding for each unique address in the memory footprint = **High Storage Cost**

Intuition

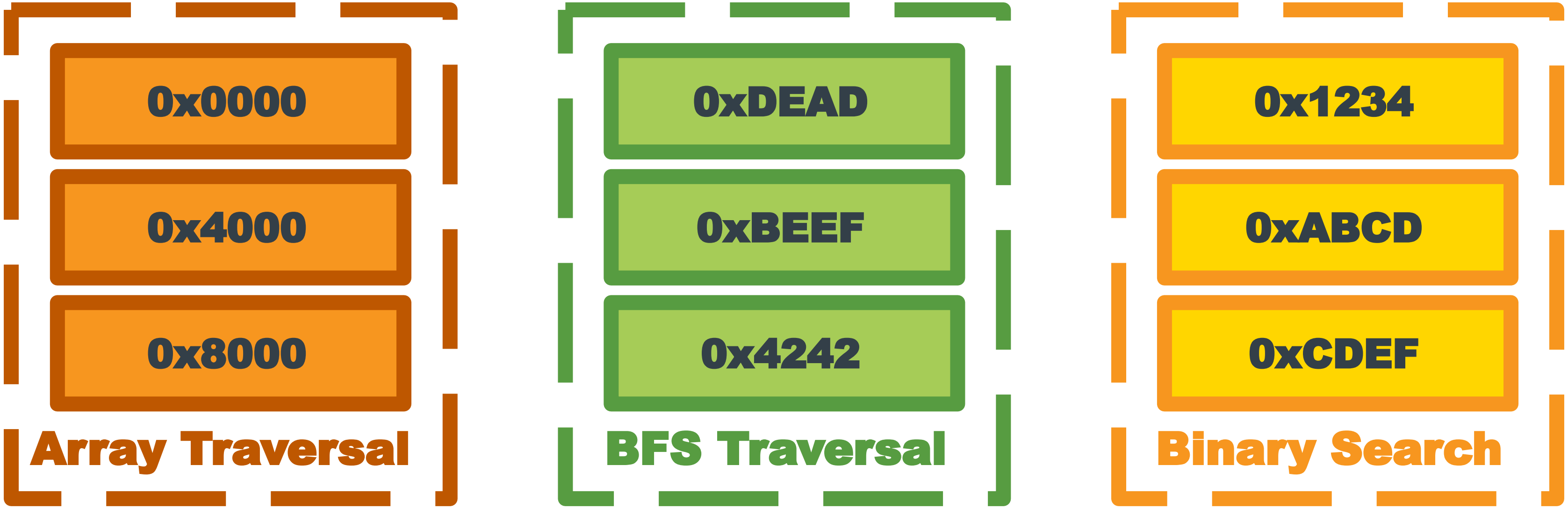
- Data of a given type have similar access patterns

Intuition

- Data of a given type have similar access patterns
  - Per-Address Neural Encodings = **Redundant + Wasteful**

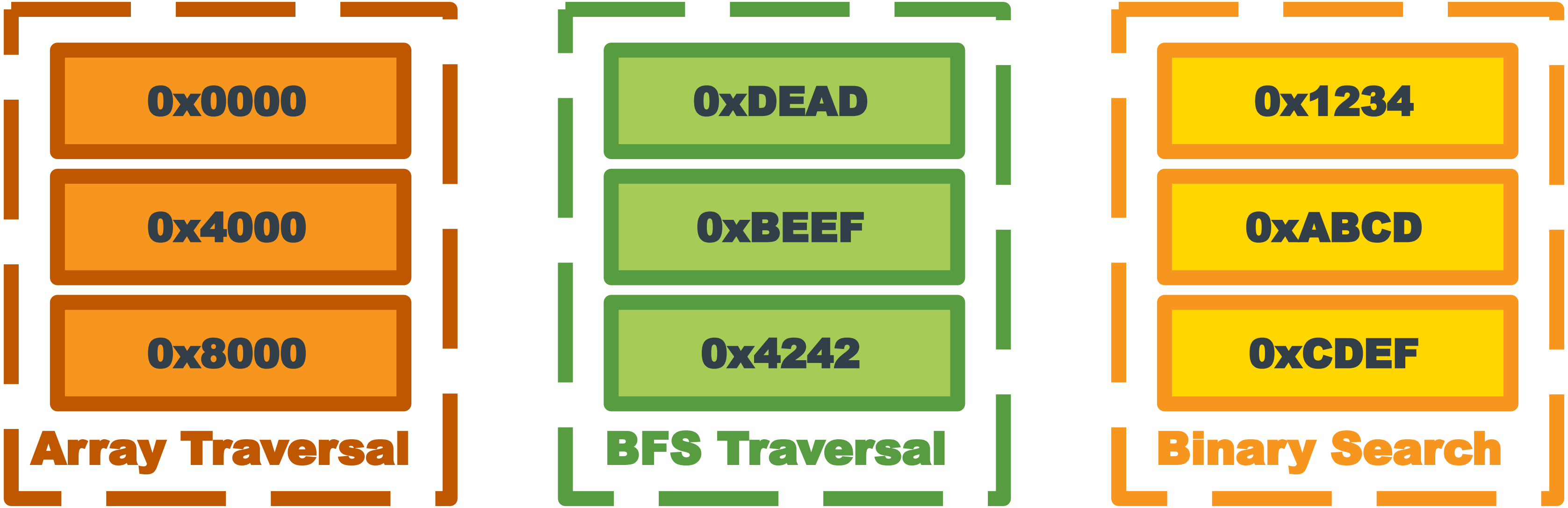
Behavioral Clustering

- Group addresses based on their prefetching behavior



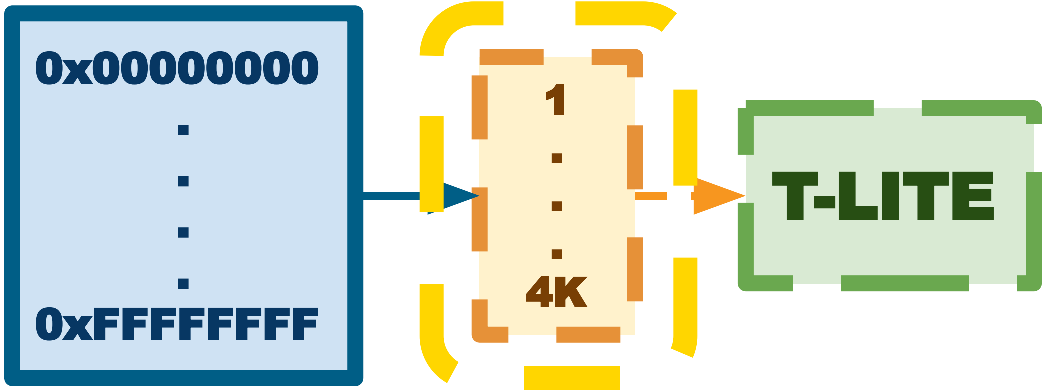
Behavioral Clustering

- Group addresses based on their prefetching behavior
  - Each cluster shares one neural encoding



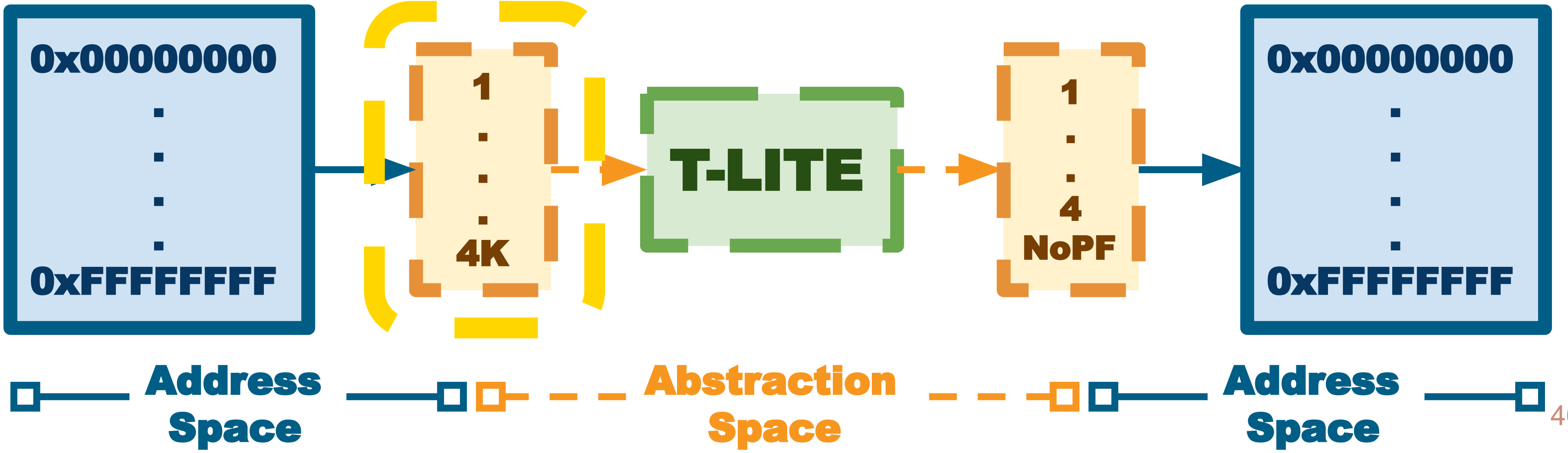
Behavioral Clustering

- Insulates T-LITE from addresses on the input side



Behavioral Clustering

- Insulates T-LITE from addresses on the input side
  - **Constant Storage Cost**



Evaluation

Twilight vs Voyager

- Unconstrained Evaluation
  - Purely comparing predictive power

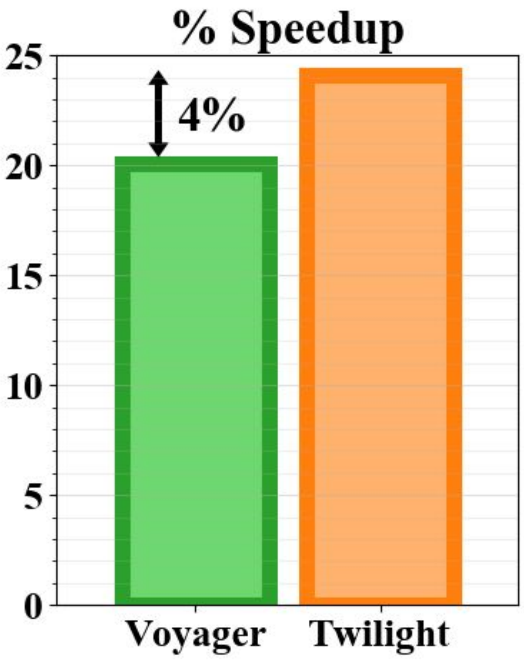
Twilight vs Voyager

- Unconstrained Evaluation
  - Purely comparing predictive power
- Compared to Voyager, Twilight:
  - has **988× shorter prediction latency**
  - requires **10.8× less neural model storage**

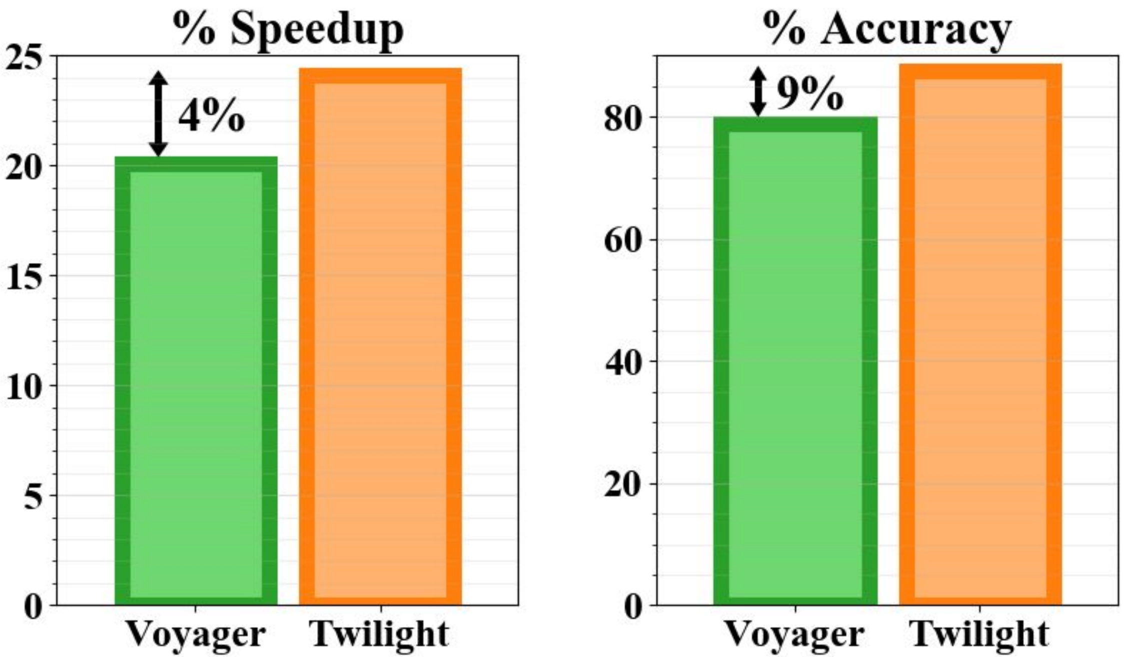
Twilight vs Voyager

- Unconstrained Evaluation
  - Purely comparing predictive power
- Compared to Voyager, Twilight:
  - has **988× shorter prediction latency**
  - requires **10.8× less neural model storage**
- <u>How much does Twilight give up for this compression?</u>

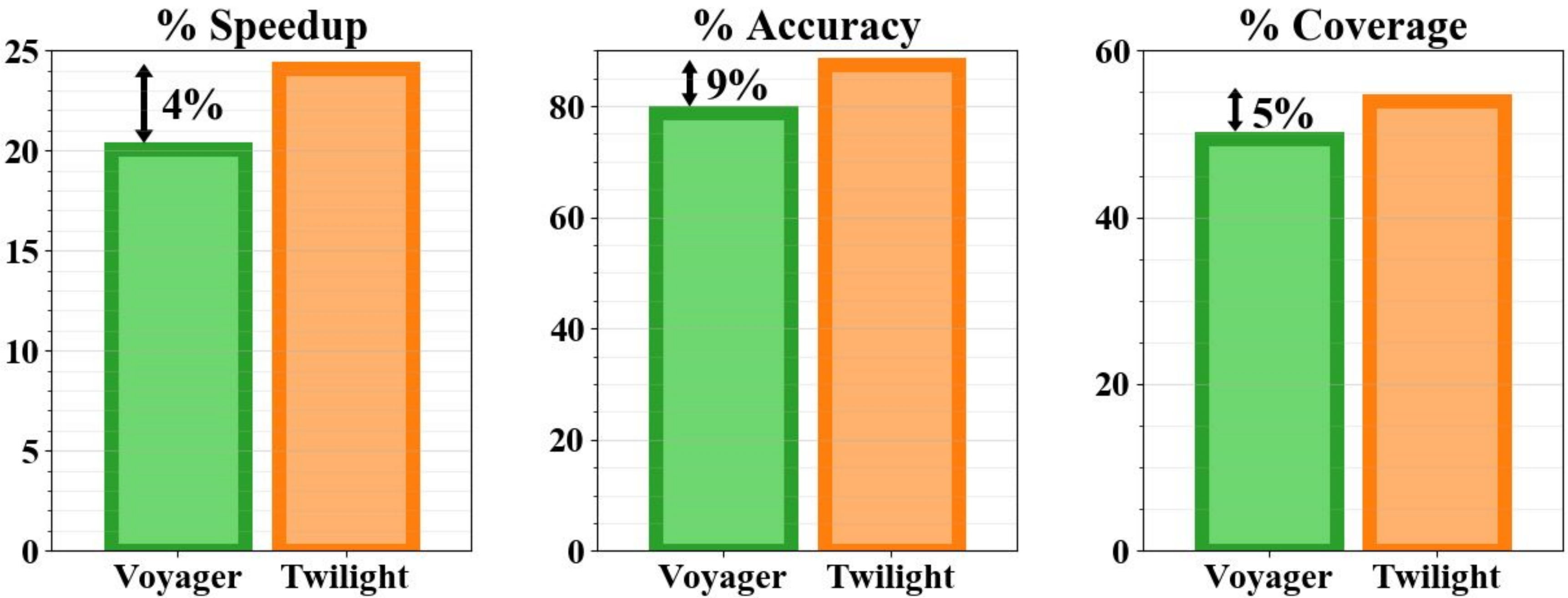
Twilight Evaluation



Twilight Evaluation



Twilight Evaluation



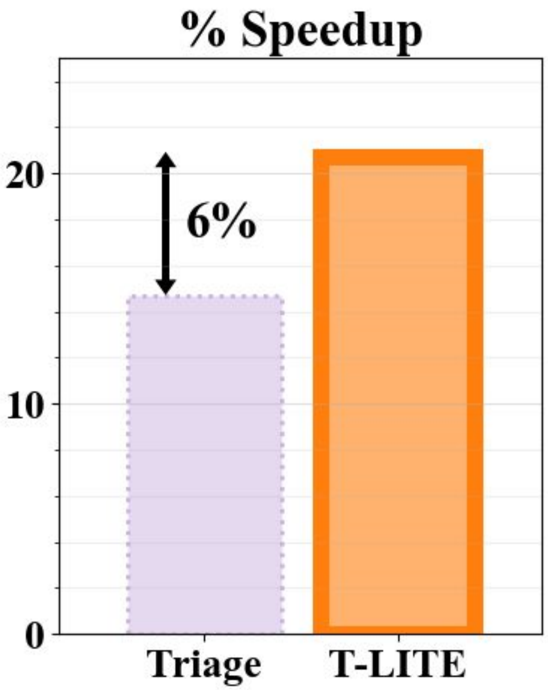
T-LITE vs Practical Prefetchers

- More Realistic Evaluation
  - Metadata storage cost
  - T-LITE prediction latency

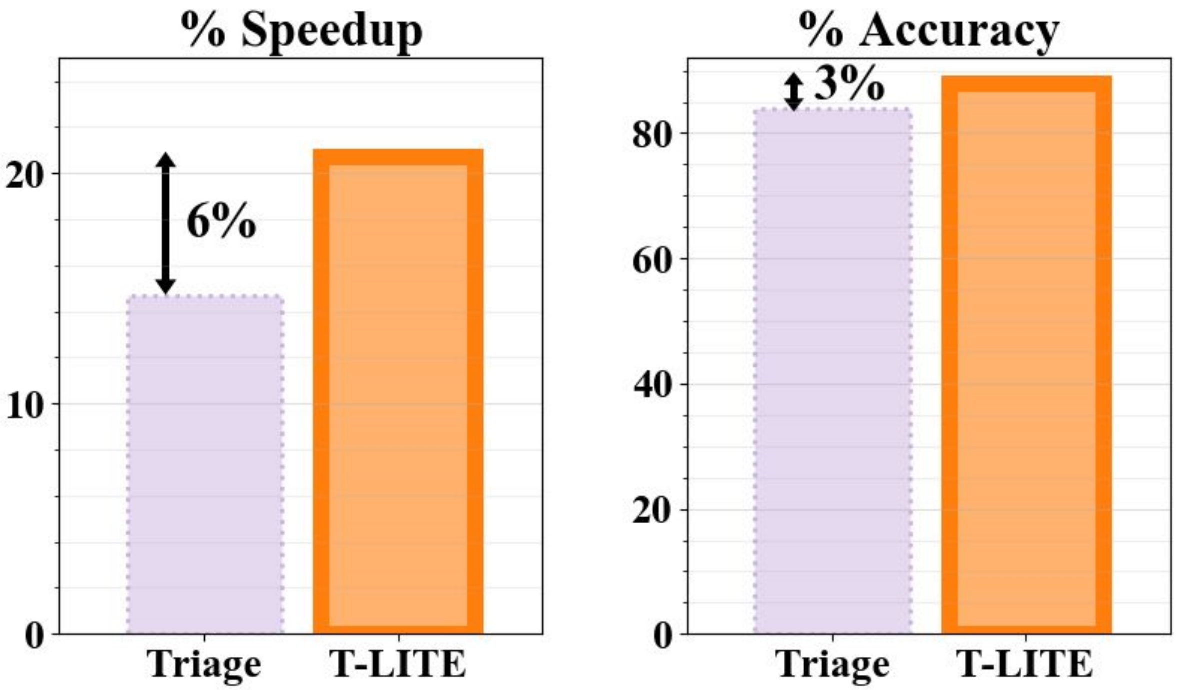
T-LITE vs Practical Prefetchers

- More Realistic Evaluation
  - Metadata storage cost
  - T-LITE prediction latency
- Baseline:
  - Triage [MICRO 2019]

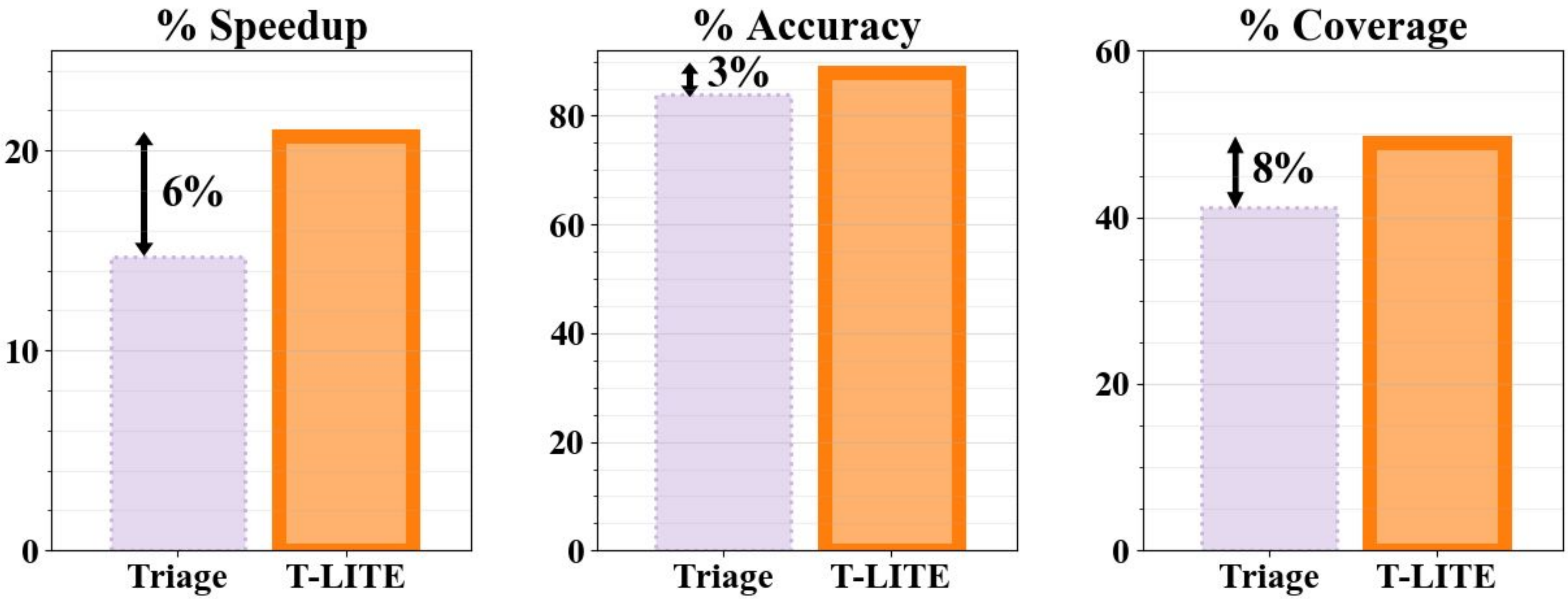
T-LITE vs Practical Prefetchers



T-LITE vs Practical Prefetchers



T-LITE vs Practical Prefetchers



Cross-Input Evaluation

- Since T-LITE is trained offline, useful deployment **requires T-LITE to work across program inputs** by

Cross-Input Evaluation

- Since T-LITE is trained offline, useful deployment **requires T-LITE to work across program inputs** by
  - Adapting to new addresses

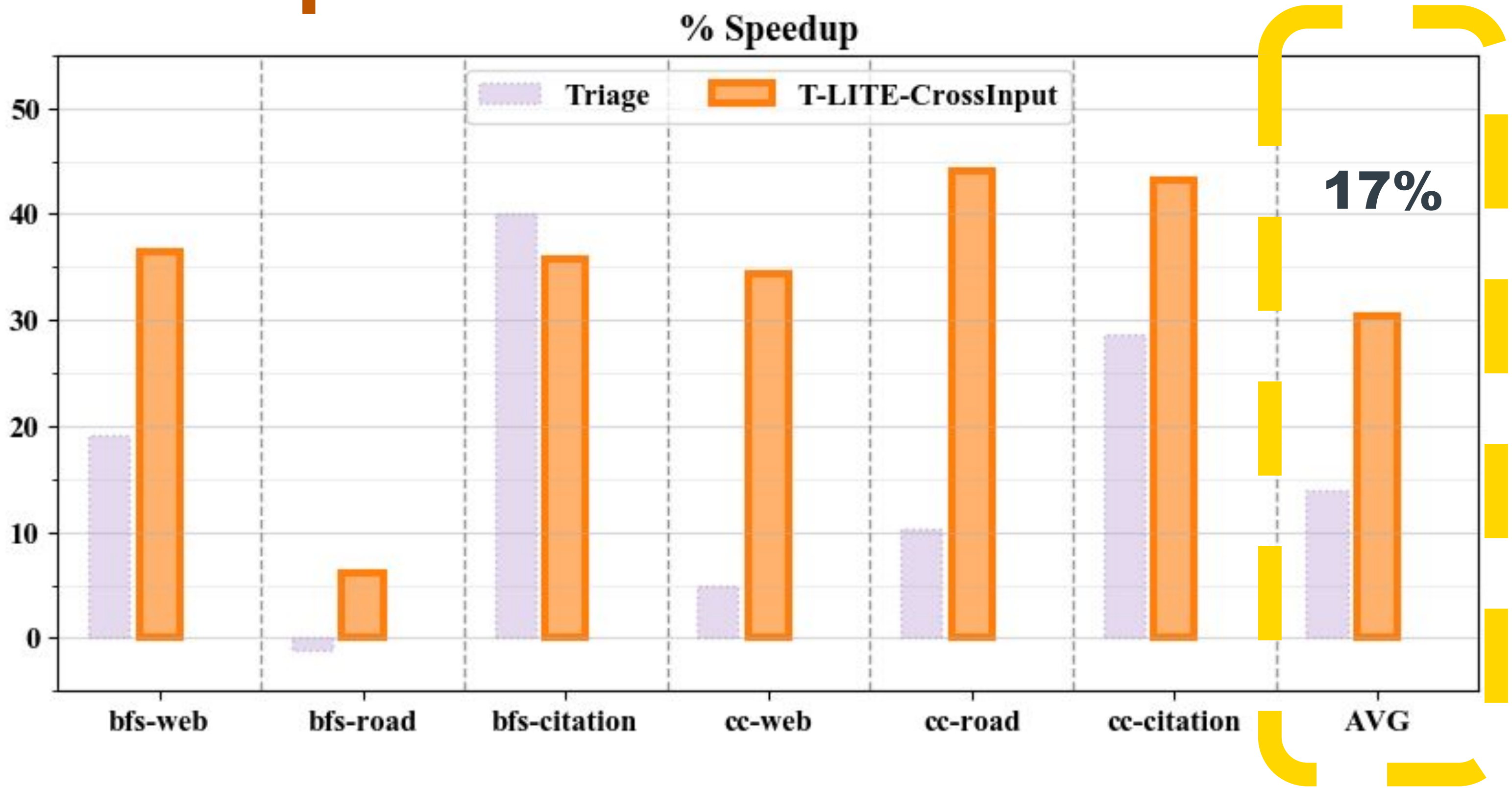
Cross-Input Evaluation

- Since T-LITE is trained offline, useful deployment **requires T-LITE to work across program inputs** by
  - Adapting to new addresses
  - Learning new correlations

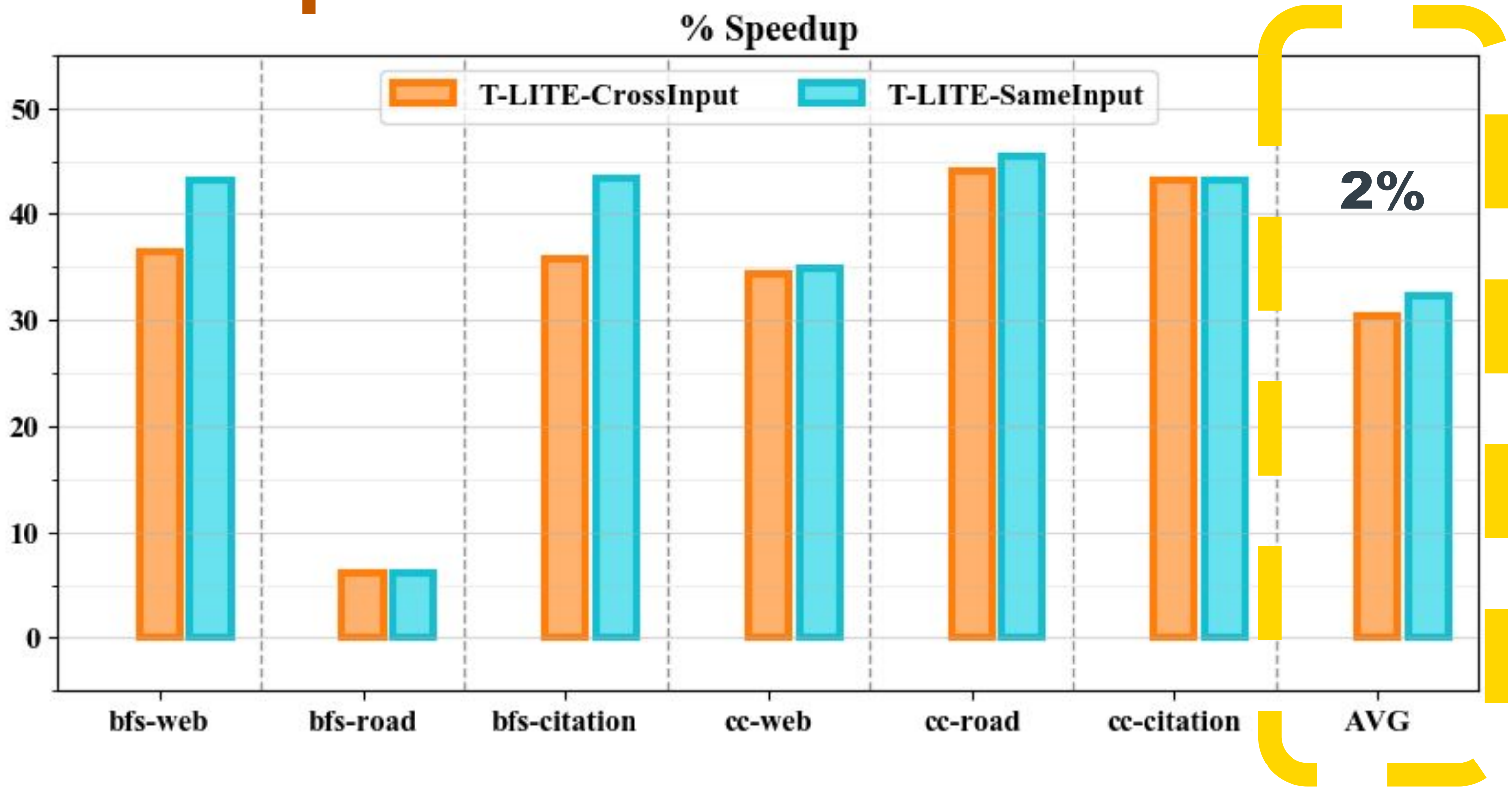
Cross-Input Evaluation

- Since T-LITE is trained offline, useful deployment **requires T-LITE to work across program inputs** by
  - Adapting to new addresses
  - Learning new correlations
- Evaluate on GAP traces across diverse input domains: **road**, **citation**, **web**
  - <u>Train, validate, and evaluate on different domains to eliminate data leakage</u>

Cross-Input Evaluation



Cross-Input Evaluation



Conclusion

- We make neural temporal prefetching **near-practical**
  - 142× less storage
  - 1421× faster prediction
  - No longer grows with memory footprint

Conclusion

- We make neural temporal prefetching **near-practical**
  - 142× less storage
  - 1421× faster prediction
  - No longer grows with memory footprint
- The key is our reformulation of the problem
  - <u>We abstract away from the address space</u>

Questions?