POET: Training Neural Networks on Tiny Devices with Integrated Rematerialization and Paging

Slides are borrowed from the author

Shishir G. Patil

With Paras Jain, Prabal Dutta, Ion Stoica, Joseph Gonzalez

https://github.com/ShishirPatil/poet

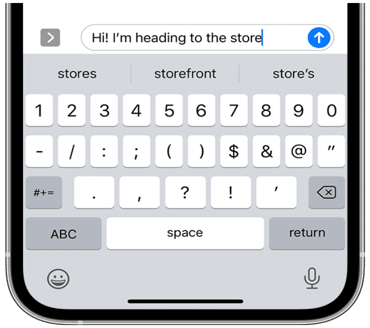
ICML
International Conference
On Machine Learning

riselab
UC Berkeley

BAIR
BERKELEY ARTIFICIAL INTELLIGENCE RESEARCH

Model Personalization Adapts Models by Training on User Data to Improve Accuracy

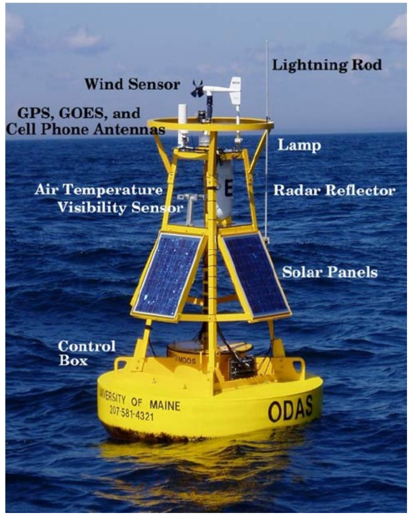
Privacy, no internet access



Autocompletion

Voice Recognition

Fitness Tracker



Ocean sensing

+ energy consumed by bulk data transmission can significantly reduce battery life

Model Fine-tuning – Train on Edge

Fine-tune on-device

Train

Key Challenge: Limited memory for DNN training!

Pros:

+ guarantees user’s privacy as all data stays on their device
+ enables offline device operation

Cons:

- cannot train modern DNNs on edge devices

Memory optimization techniques

- **Pruning**
  - They do not reduce the size of activations.
  - Accuracy trade-off
- **Quantization**
  - poor hardware support for quantized operations under 8 bits
    Accuracy trade-off
- **Rematerialization**
- **Paging**

Memory optimization techniques

- Pruning
  - They do not reduce the size of activations.
  - Accuracy trade-off
- Quantization
  - poor hardware support for quantized operations under 8 bits
    Accuracy trade-off
- **Rematerialization**
- **Paging**

Value preserving
Reduce activation

Insight

- Paging is very energy-intensive
- Rematerializing might consume lower energy
- Paging might be quicker.
  - Paging can be done in parallel with the compute. DMA technique
- This is because, on edge devices, **it is common practice to turn-off components that are not utilized (e.g., SD card, DMA, etc.)**

- For example,
- piecewise(cheap-to-compute but memory-intensive) → recompute
- conv, matmul(compute-intensive) → paging

Rematerialization & Paging in DNN training

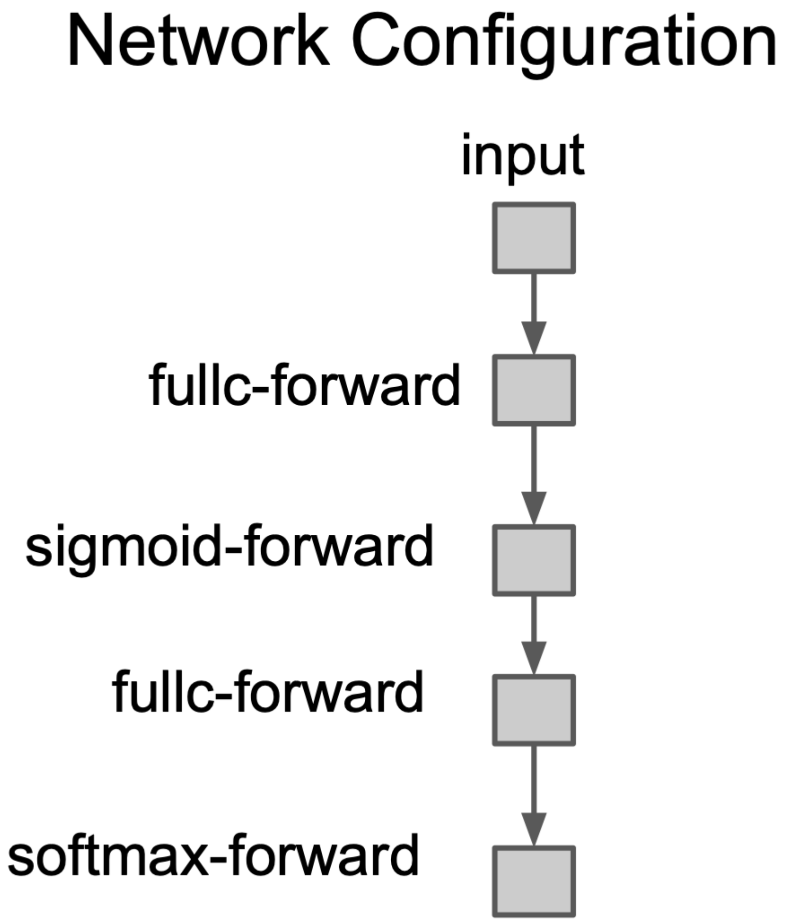
- **Sublinear & Revolve**
  - Strong assumption that models have uniform compute requirements. Heuristic so not optimal
- **Capuchin**
  - Paging as default. Rematerialization only when paging is not possible
- **Checkmate**
  - Optimal but static graph
  - Not energy-aware
  - No paging
- **POFO**
  - Not energy-aware
  - Assumes paging is asynchronous (e.g., CUDA) but this is not universally true for the edge devices we evaluate.

Comparison

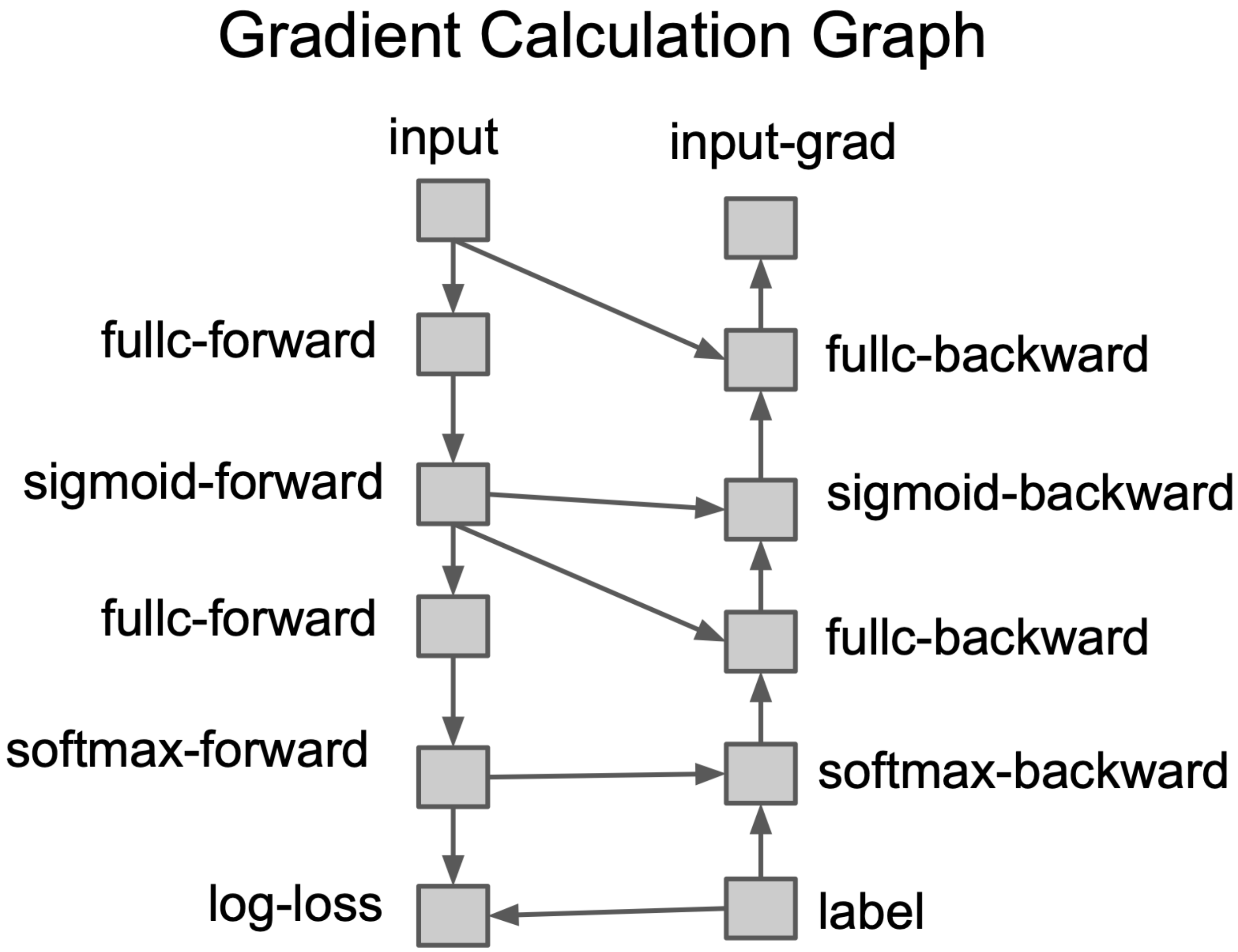
Method	General Graphs	Compute Aware	Memory Aware	Power Aware
Checkpoint all (PyTorch)	$\surd$	$\times$	$\times$	$\times$
Griewank & Walther (2000)	$\times$	$\times$	$\times$	$\times$
Chen et al. (2016) $\sqrt{n}$	$\times$	$\times$	$\times$	$\times$
Chen et al. (2016) greedy	$\times$	$\times$	$\sim$	$\times$
Checkmate (Jain et al., 2020)	$\surd$	$\surd$	$\surd$	$\times$
POFO (Beaumont et al., 2021)	$\times$	$\surd$	$\surd$	$\times$
DTR (Kirisame et al., 2021)	$\surd$	$\surd$	$\surd$	$\times$
POET (ours)	$\surd$	$\surd$	$\surd$	$\surd$

How to reduce the memory and energy requirements of ML training for modern DNN architectures within the constraints of edge devices?

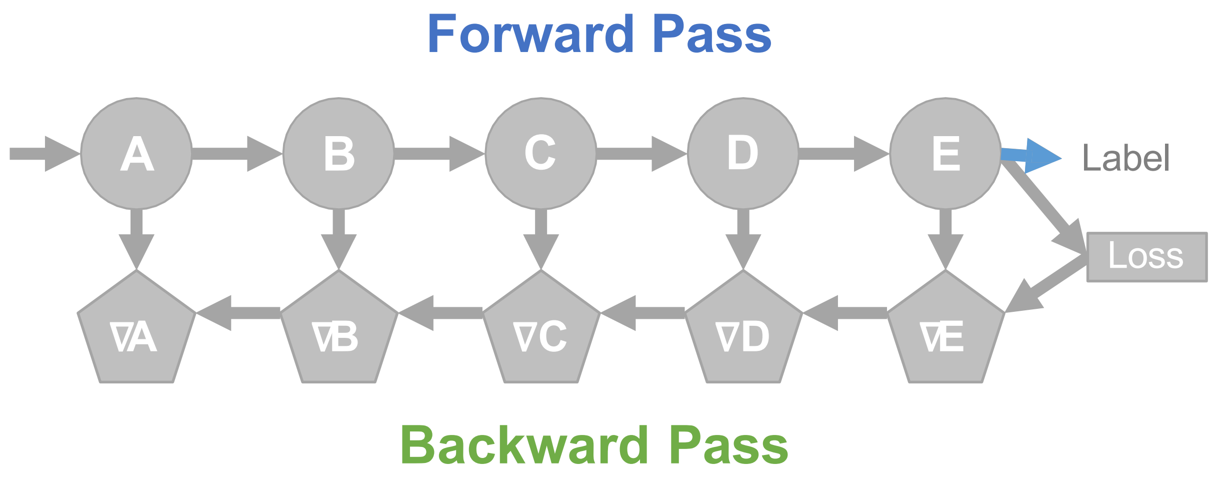
Computational graph

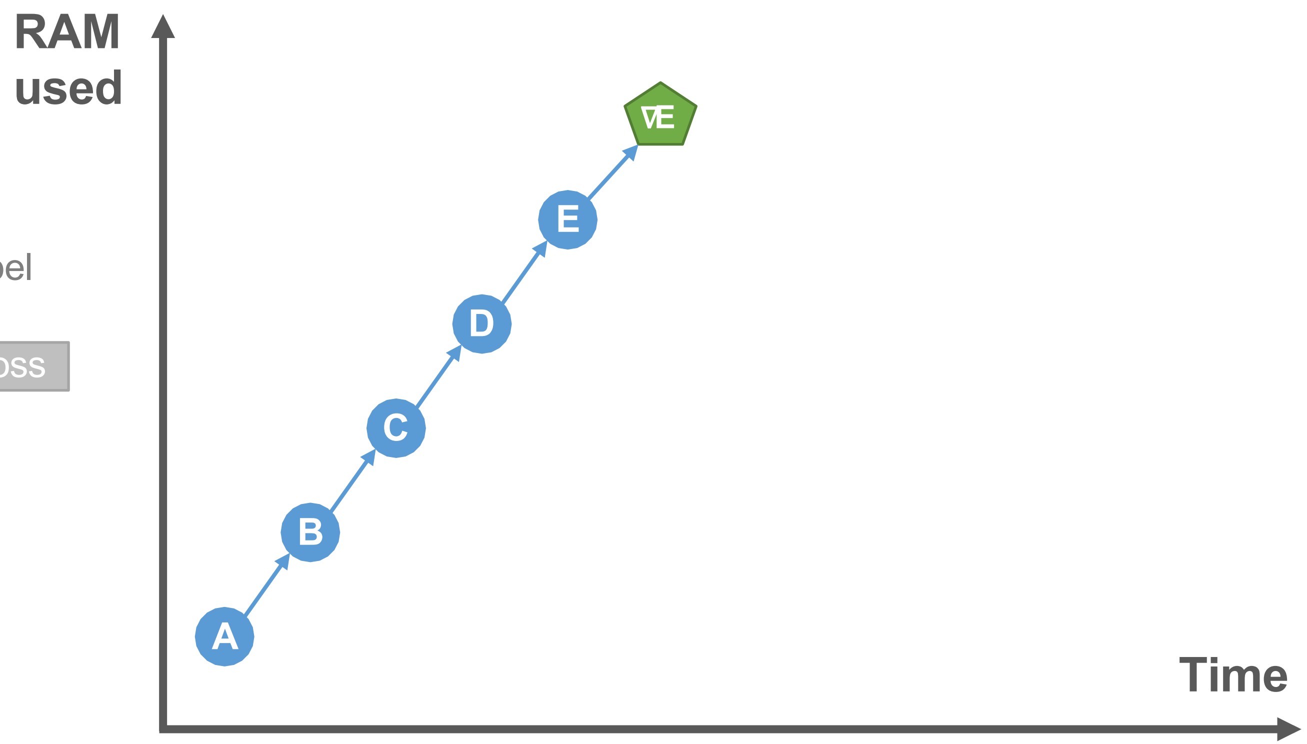


Computational graph

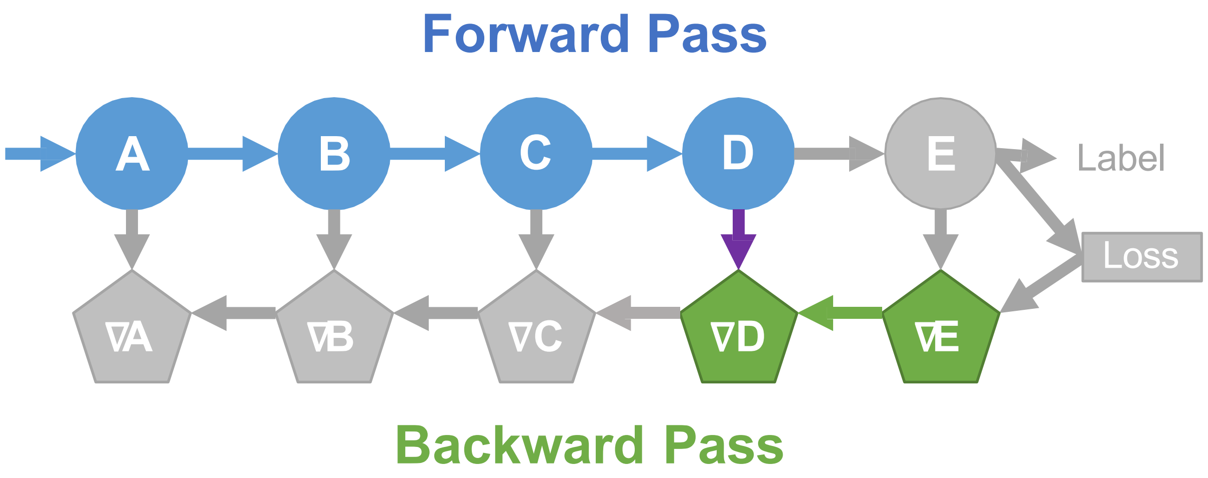


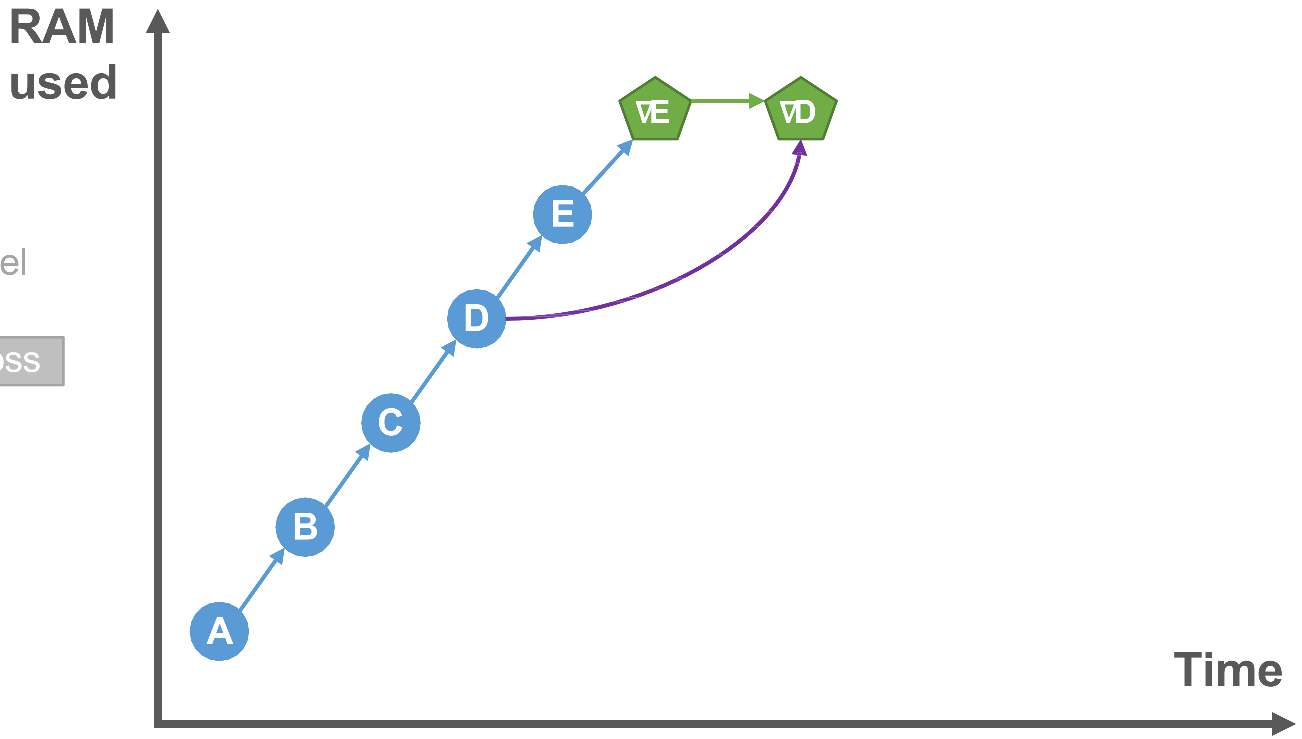
Training is Memory Intensive since Activation from Forward Pass Need to be Stored for Backpropagation



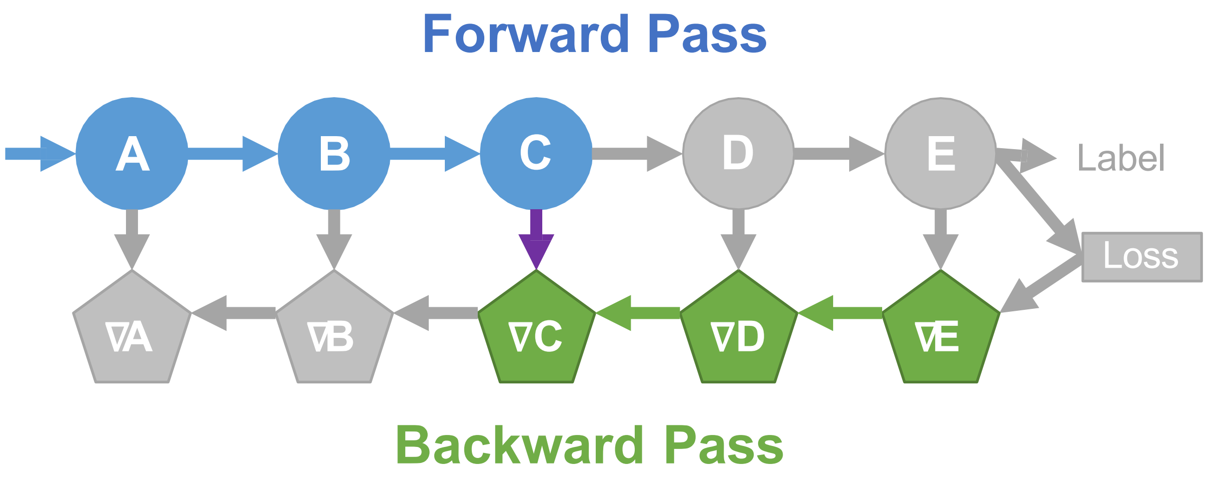


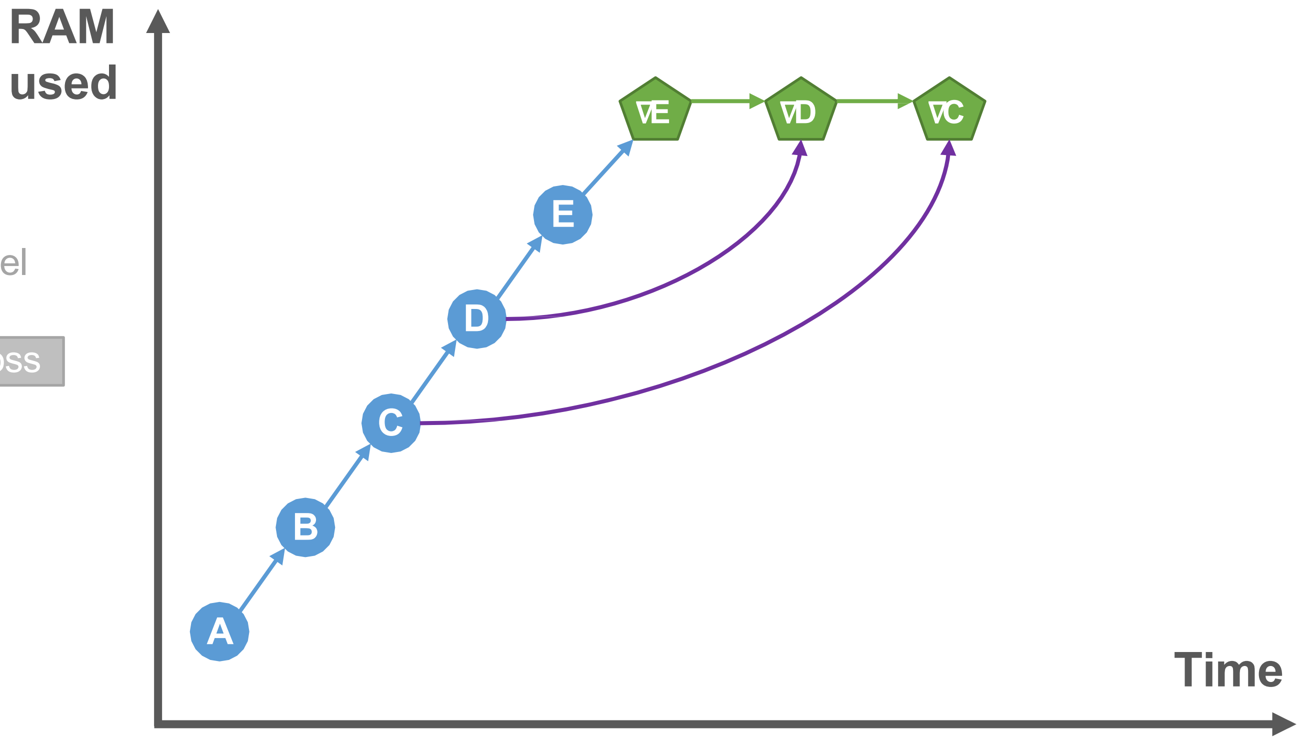
Training is Memory Intensive since Activation from Forward Pass Need to be Stored for Backpropagation



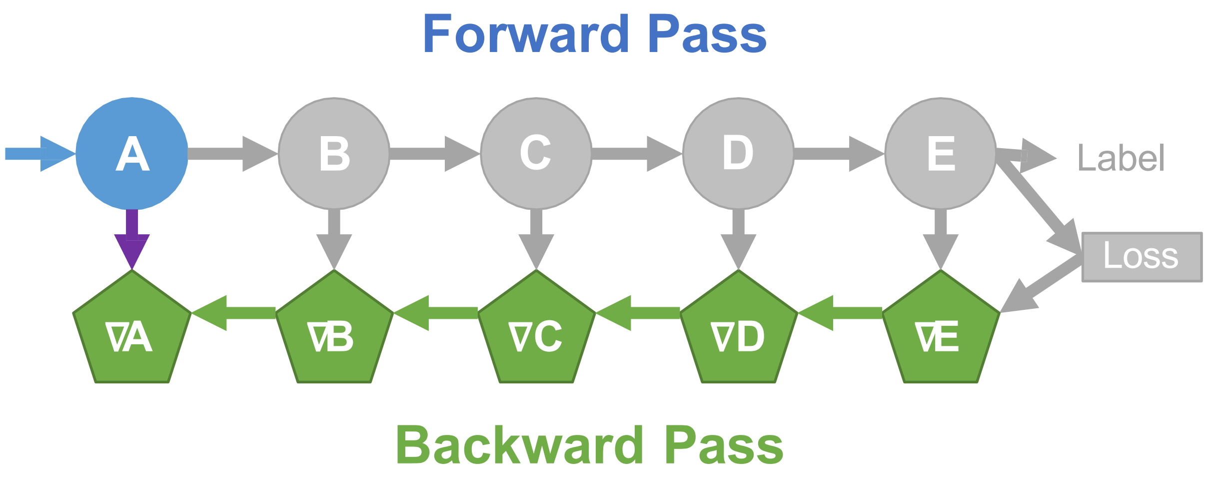


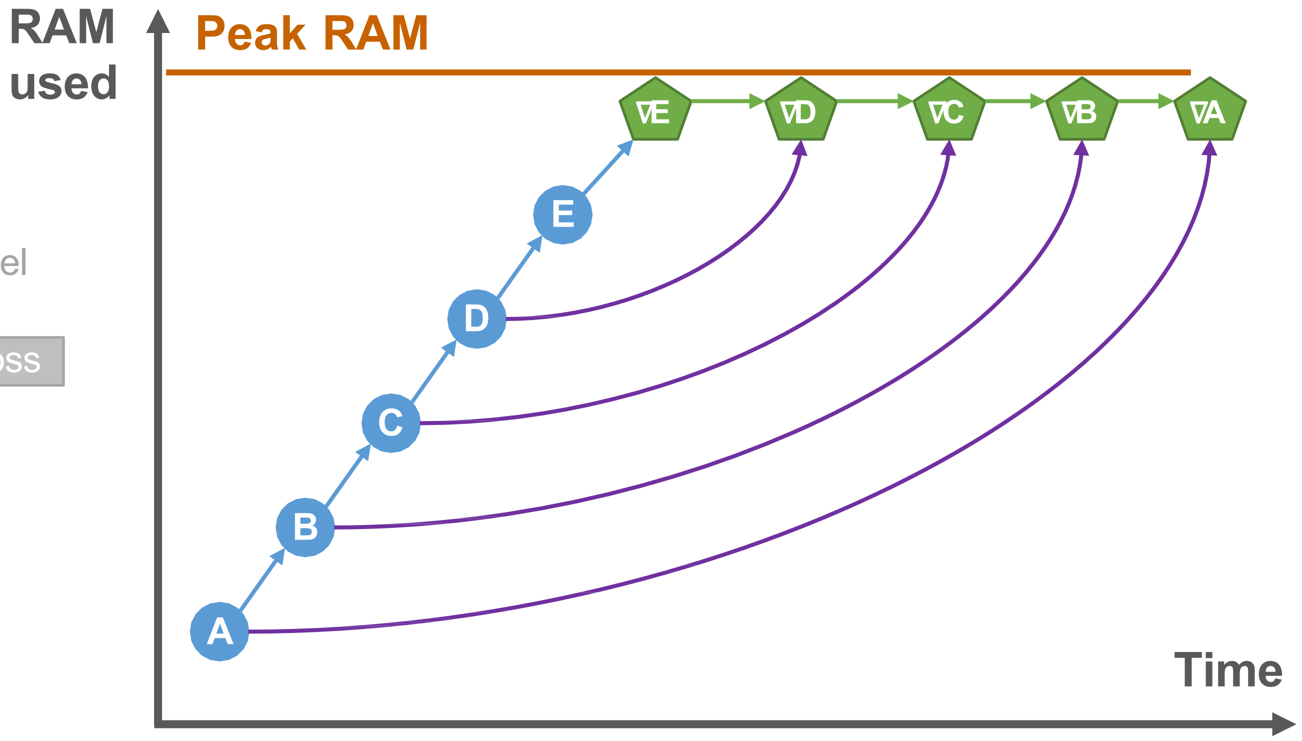
Training is Memory Intensive since Activation from Forward Pass Need to be Stored for Backpropagation



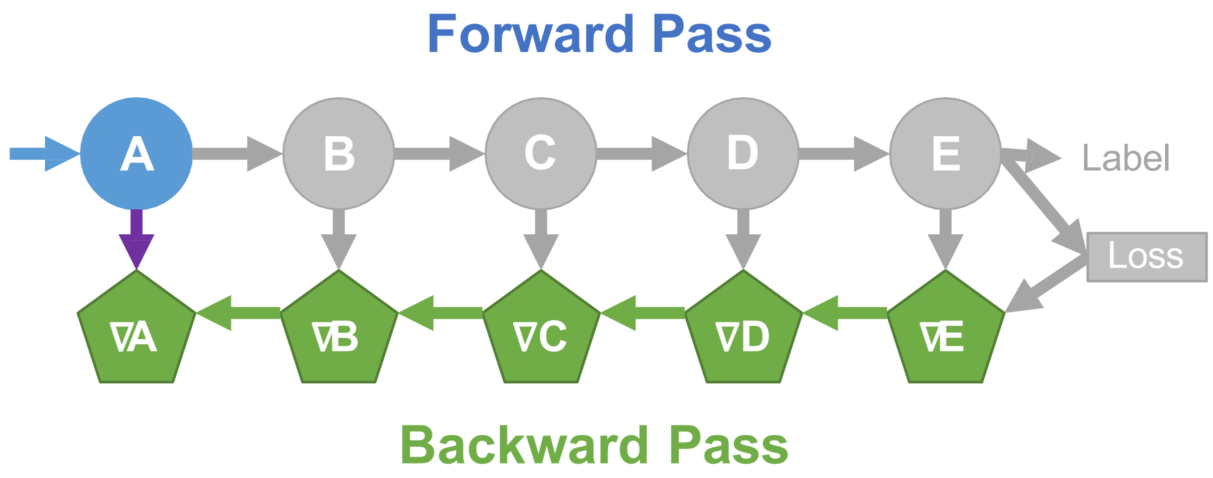


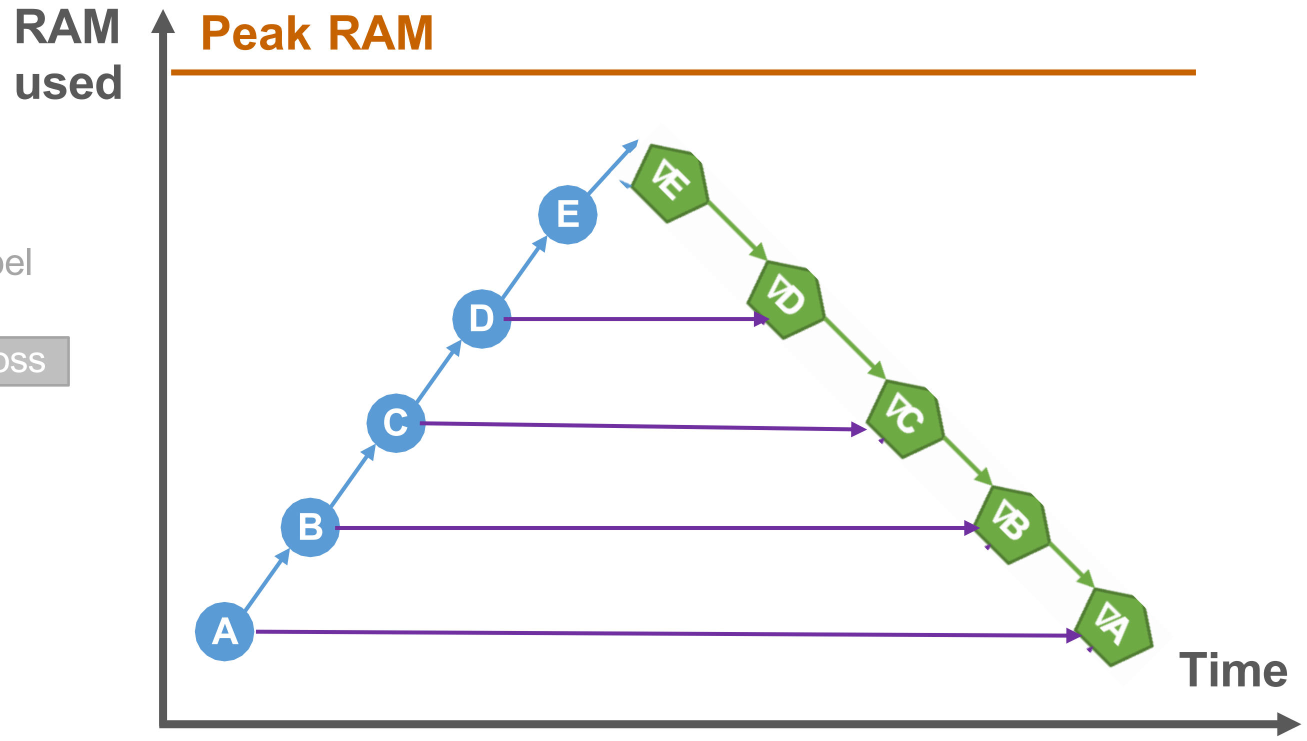
Training is Memory Intensive since Activation from Forward Pass Need to be Stored for Backpropagation



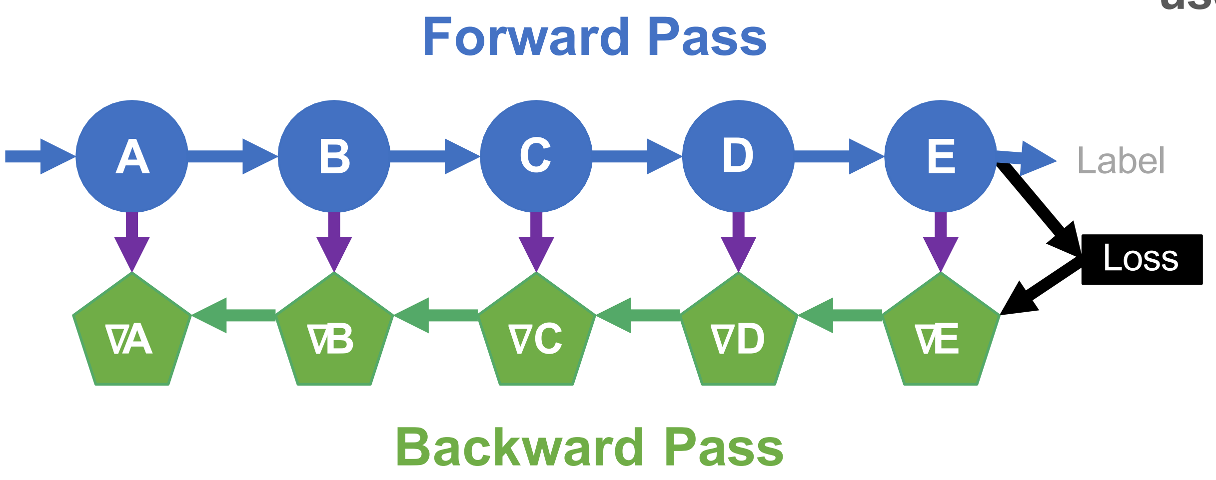


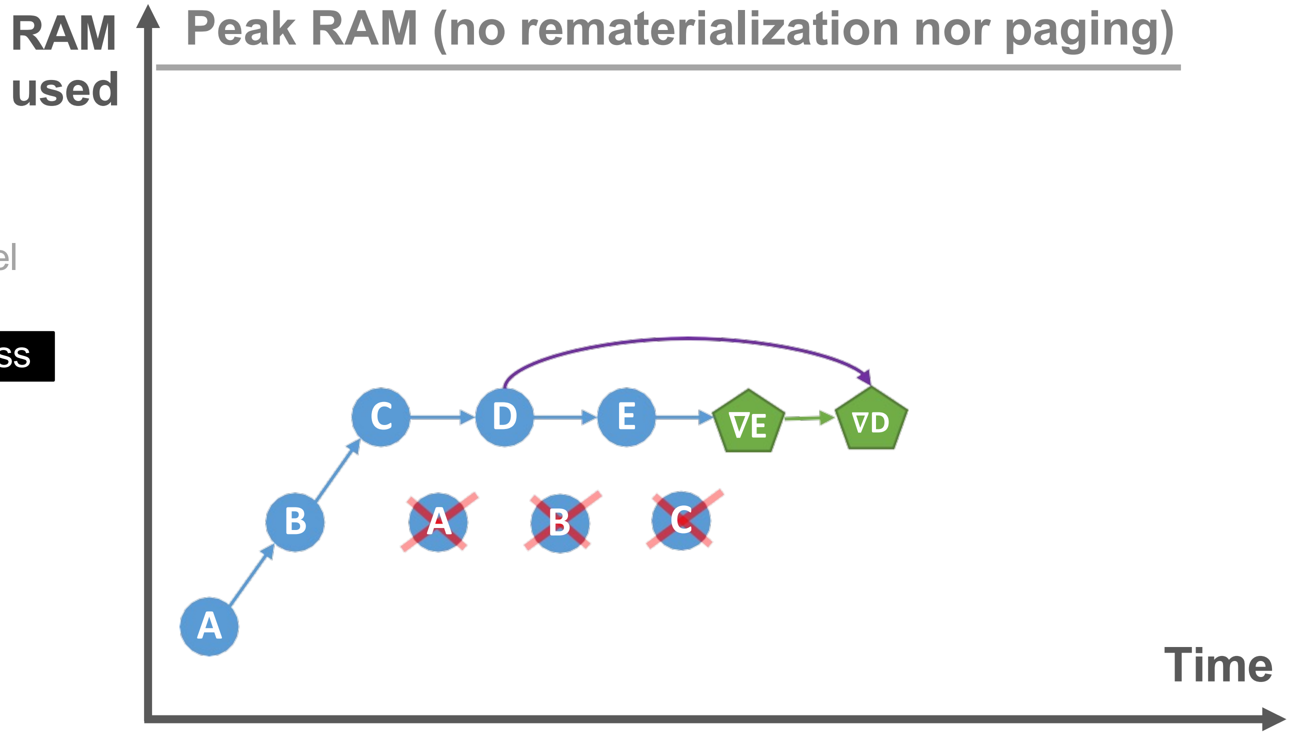
Training is Memory Intensive since Activation from Forward Pass Need to be Stored for Backpropagation





Rematerialization and Paging: Two Techniques to Reduce Memory Consumption

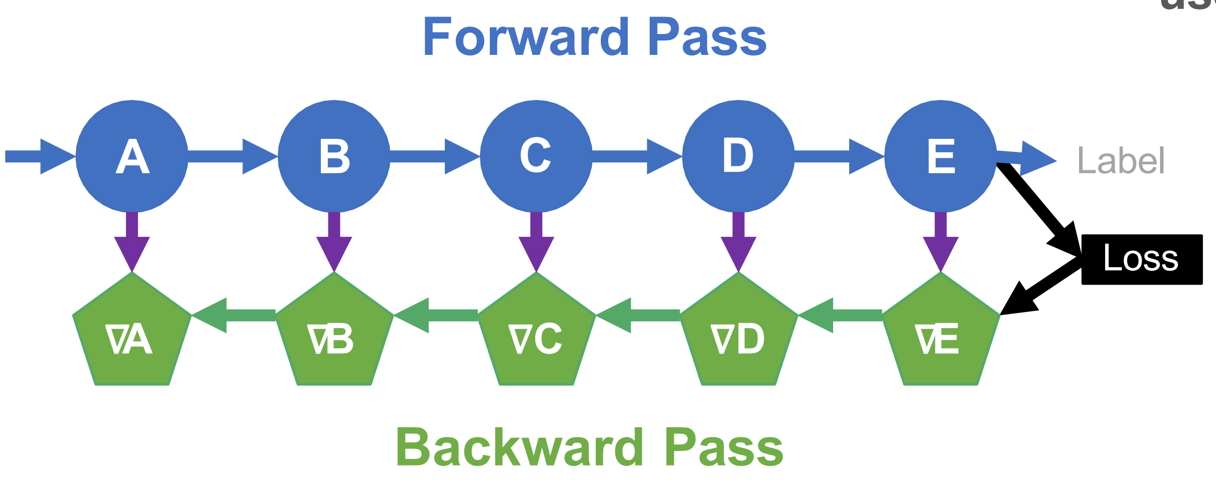


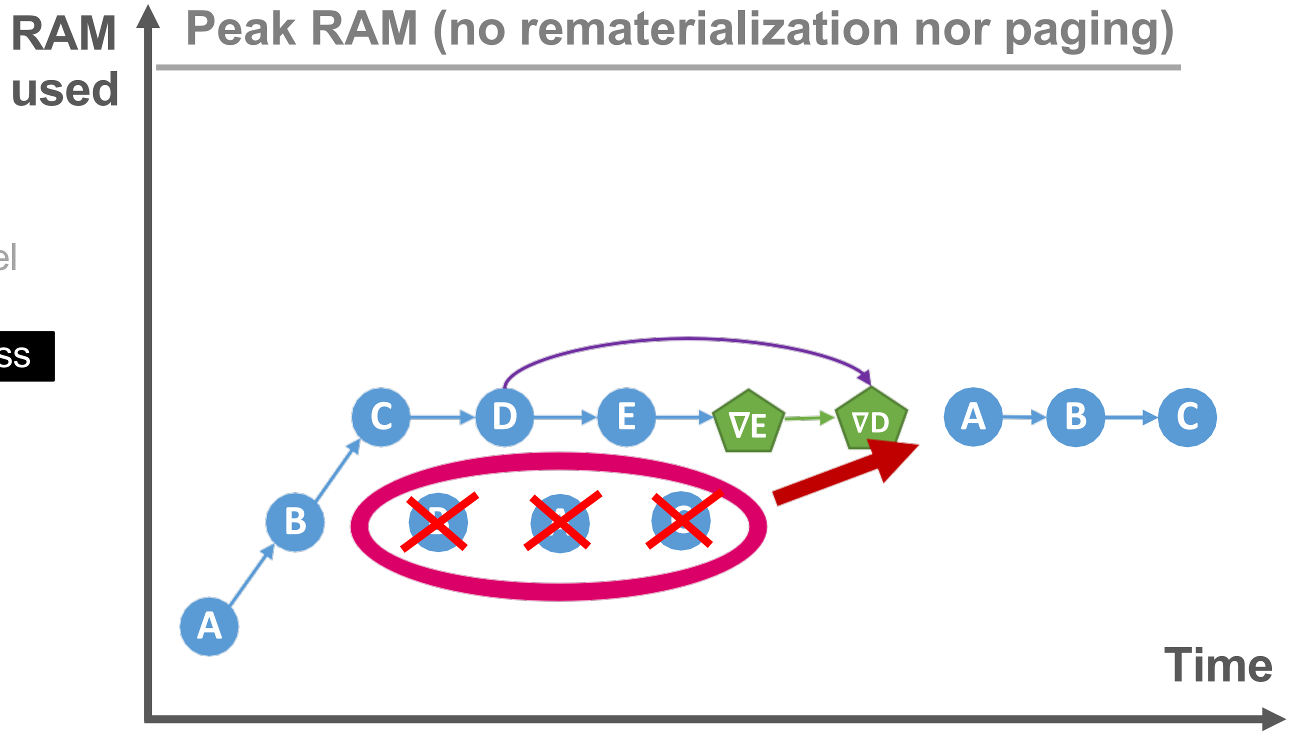


Rematerialization:

Free early & recompute

Rematerialization and Paging: Two Techniques to Reduce Memory Consumption

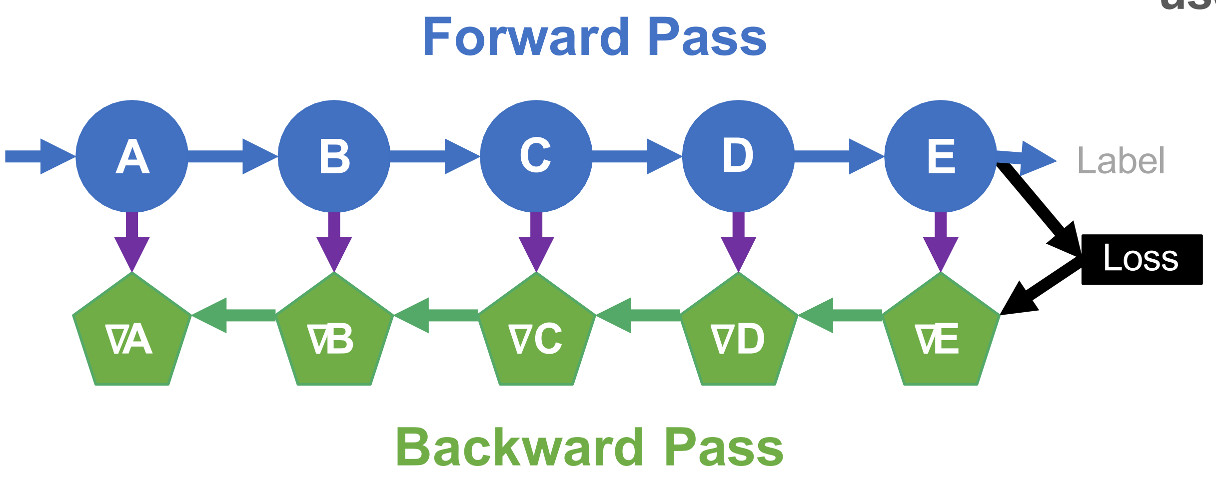


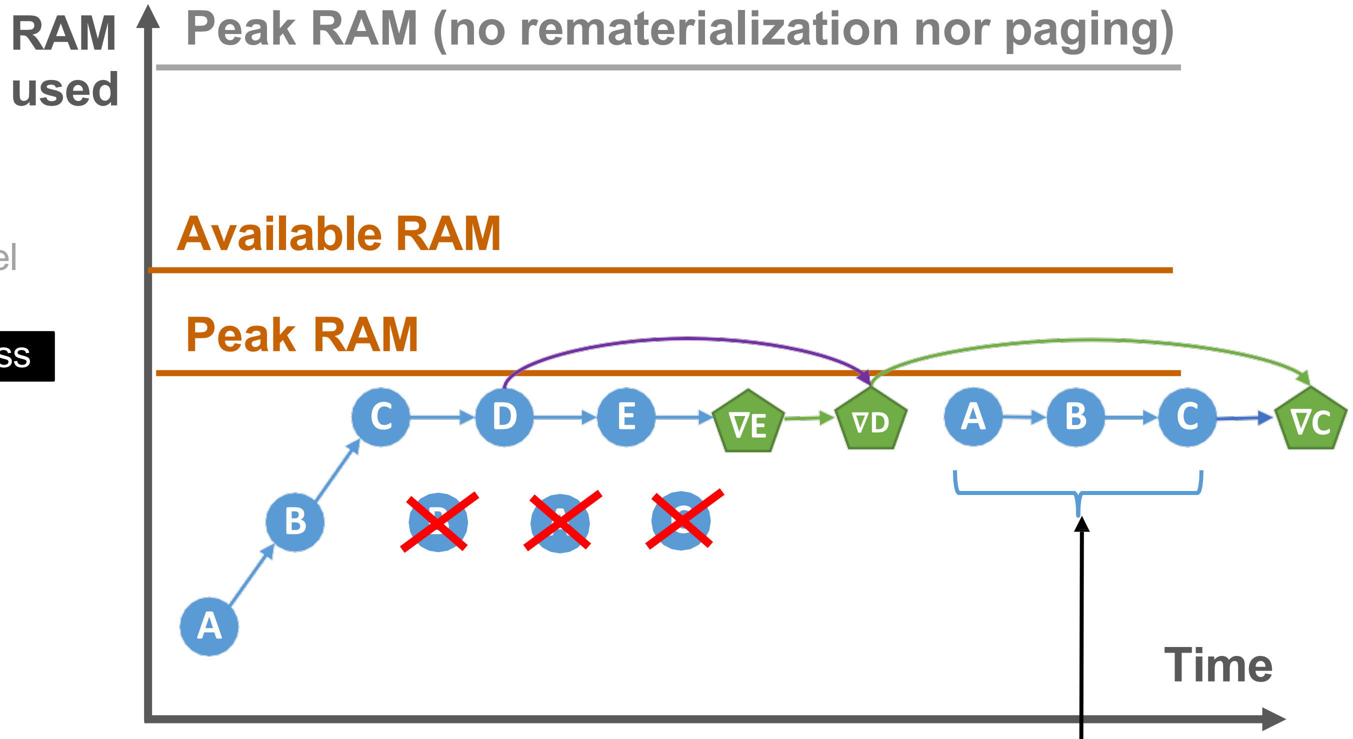


Rematerialization:

Free early & recompute

Rematerialization and Paging: Two Techniques to Reduce Memory Consumption



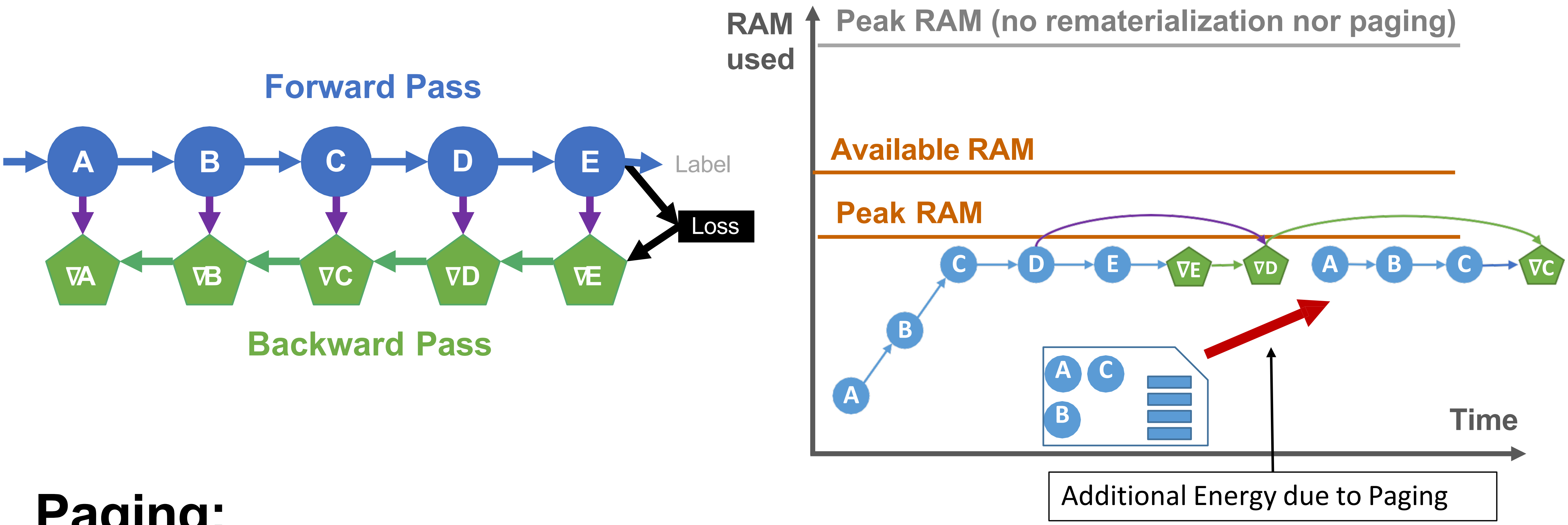


Additional Energy and runtime due to recomputation!

Rematerialization:

Free early & recompute

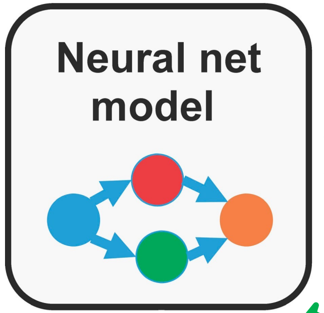
Rematerialization and Paging: Two Techniques to Reduce Memory Consumption



Paging:

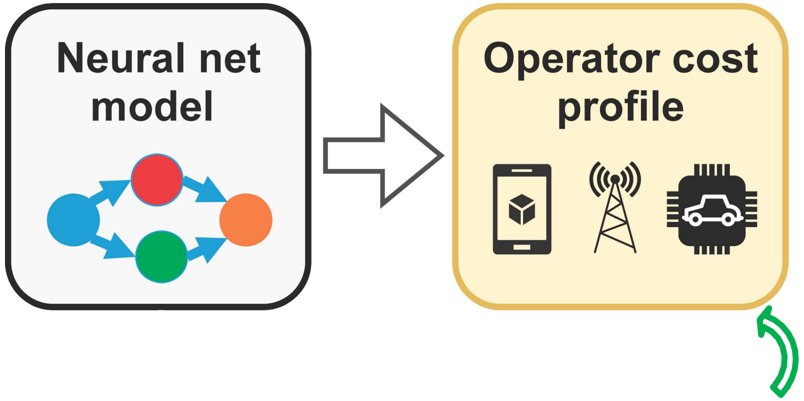
Page-out to secondary storage and page-in Just-in-Time!

POET: Private Optimal Energy Training



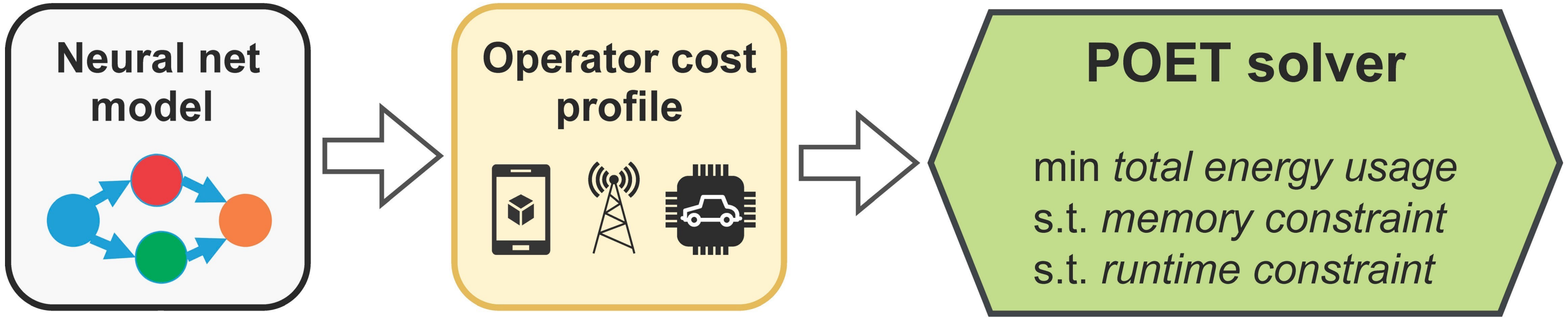
{ML model, memory and
runtime constraints}

POET: Private Optimal Energy Training



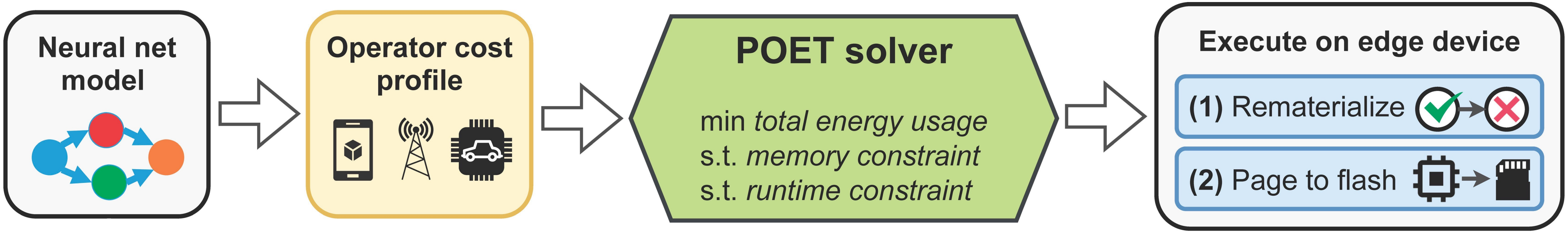
Accurate cost profile of ML operators on target edge platform

POET: Private Optimal Energy Training



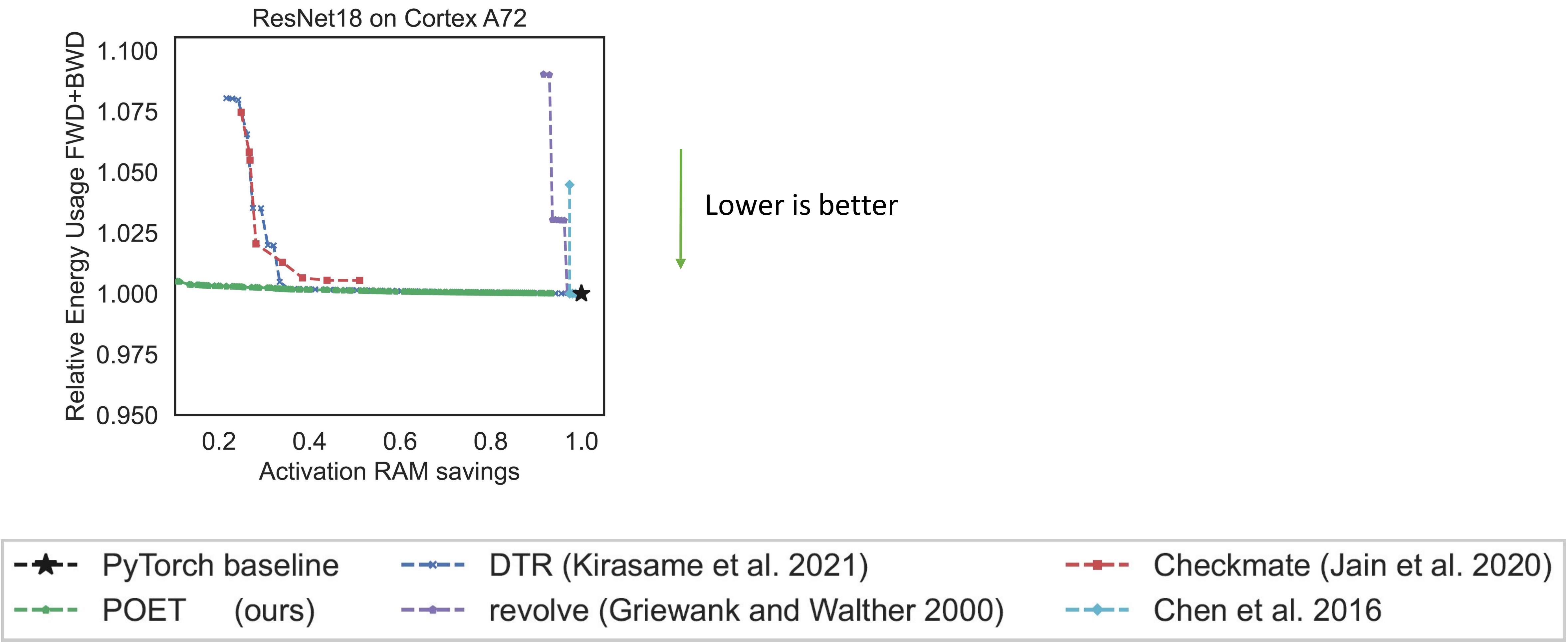
Incorporate *memory* and *runtime* constraints into a Mixed Integer Linear Program (MILP) formulation

POET: Private Optimal Energy Training

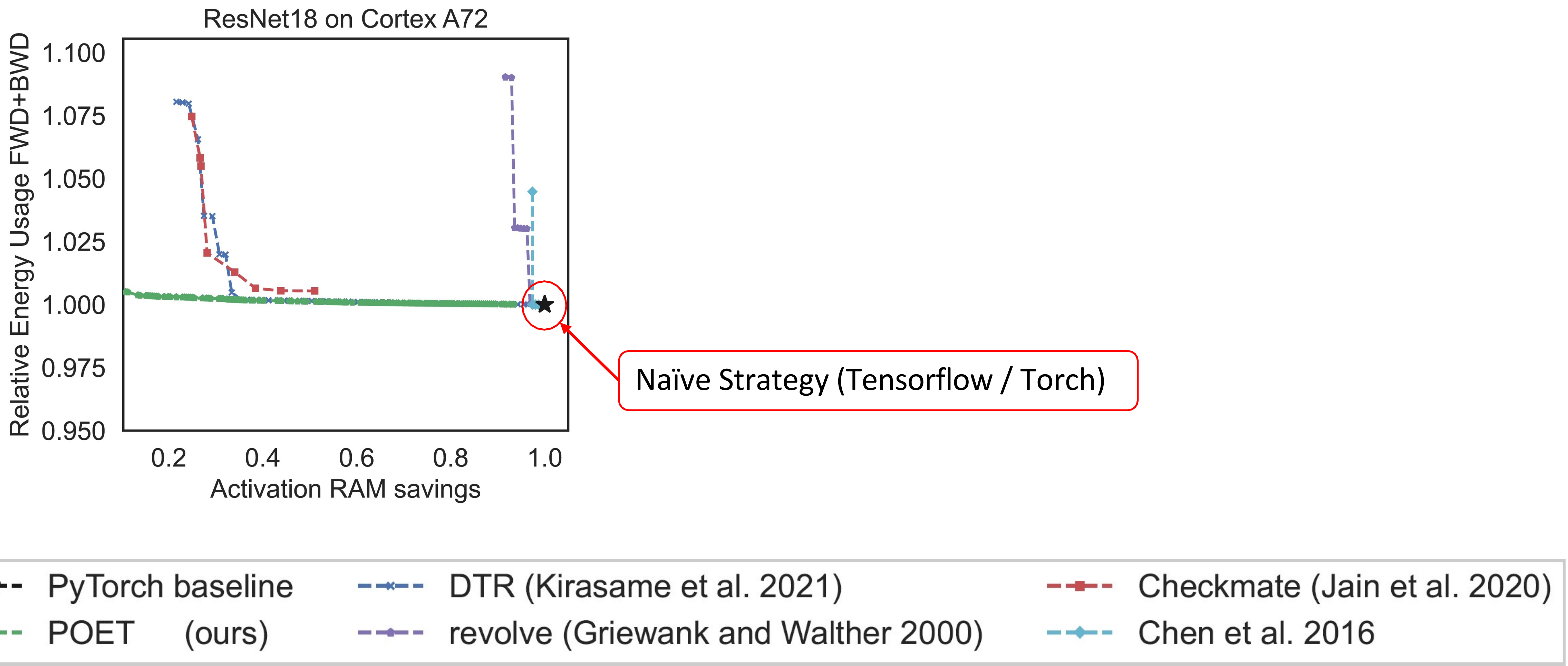


POET finds a provably optimal solution through integrated rematerialization and paging.

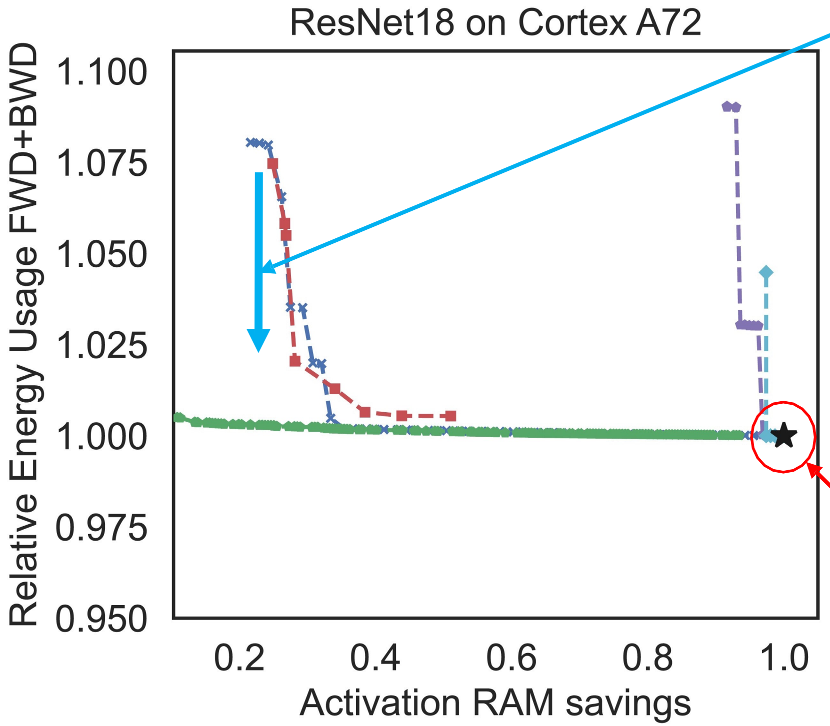
Result: POET lowers energy consumption and allows training large models previously not possible!



Result: POET lowers energy consumption and allows training large models previously not possible!



Result: POET lowers energy consumption and allows training large models previously not possible!

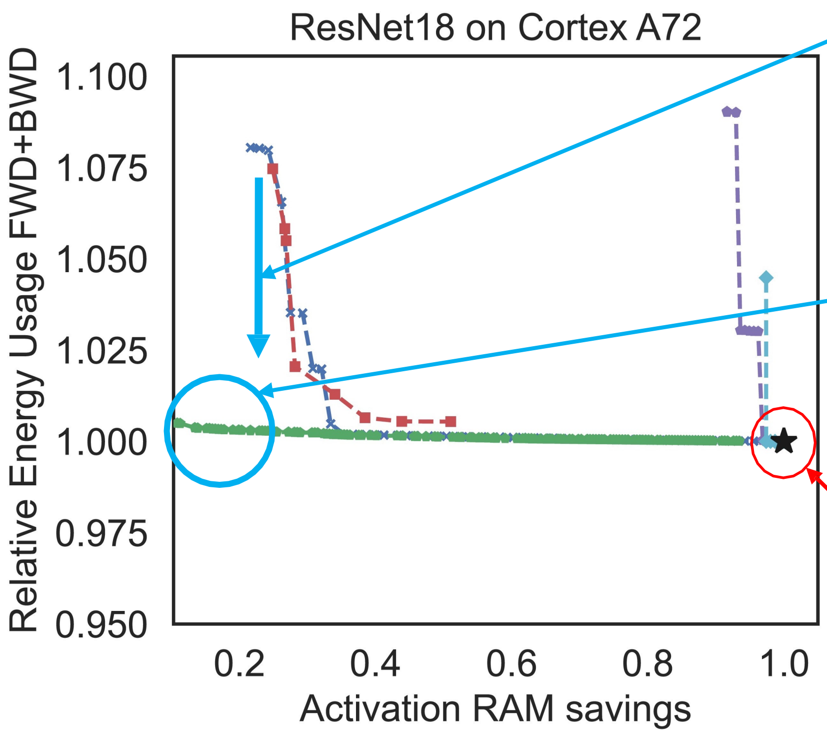


POET lowers energy consumption at all budgets

Naïve Strategy (Tensorflow / Torch)

PyTorch baseline
POET (ours)
DTR (Kirasame et al. 2021)
revolve (Griewank and Walther 2000)
Checkmate (Jain et al. 2020)
Chen et al. 2016

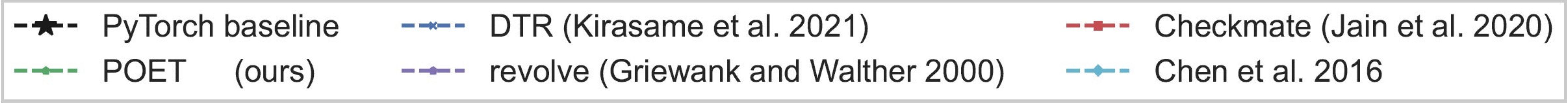
Result: POET lowers energy consumption and allows training large models previously not possible!



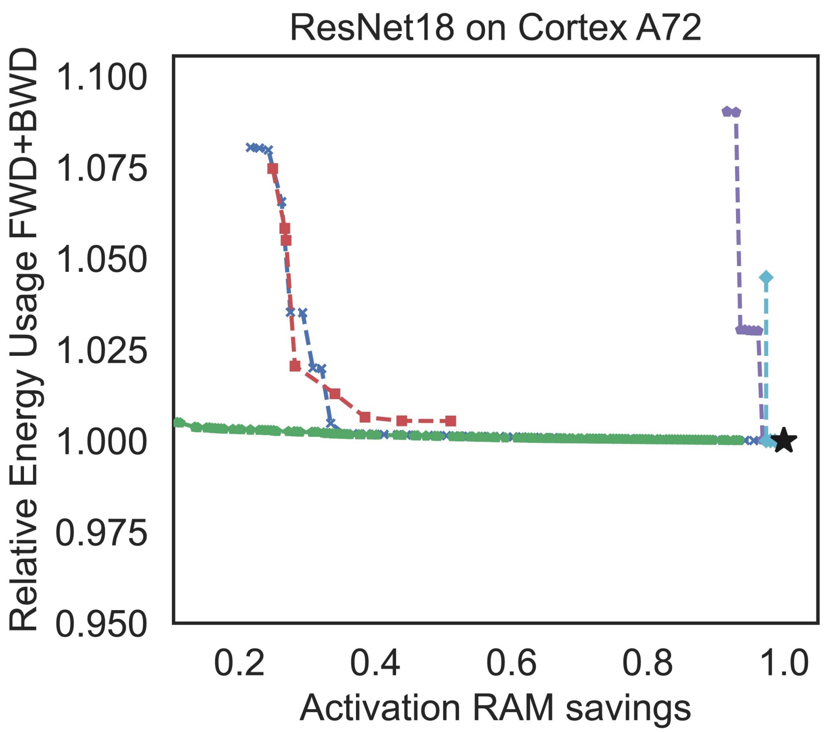
POET lowers energy consumption at all budgets

POET's integrated Rematerialization and Paging enables training with much smaller memory budgets which was previously not possible!

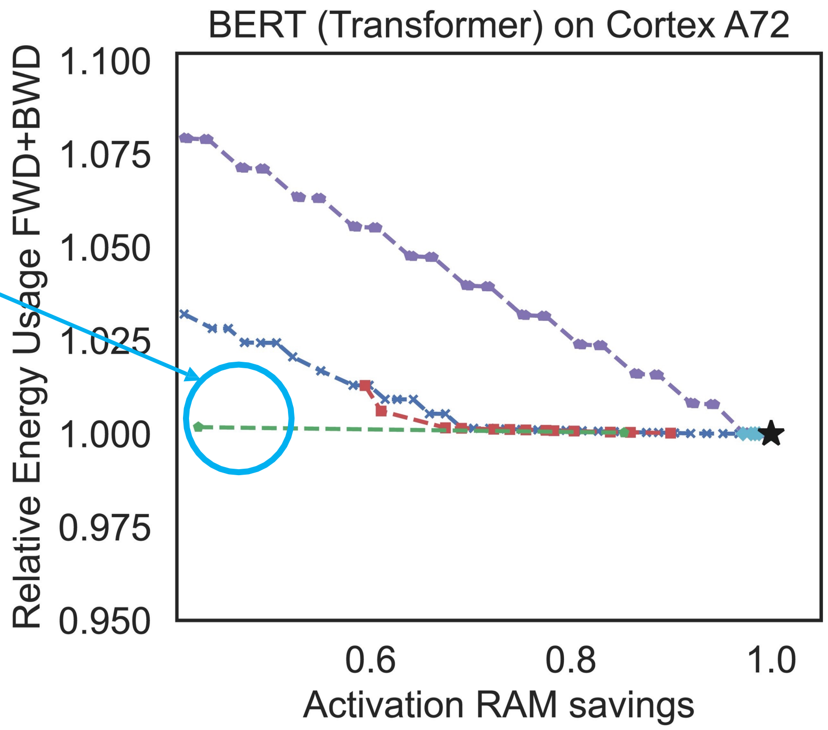
Naïve Strategy (Tensorflow / Torch)

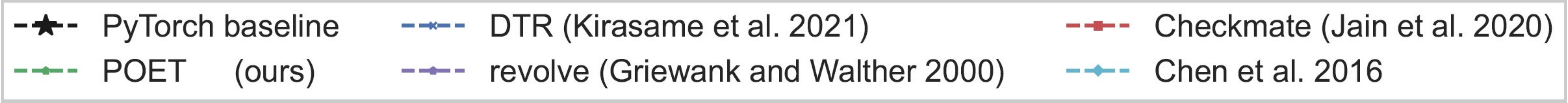


Result: POET lowers energy consumption and allows training large models previously not possible!



POET's integrated Rematerialization and Paging enables training with much smaller memory budgets which was previously not possible!





Conclusion

- POET enables training SOTA DNN models locally on memory-constrained edge devices.
- POET's fine grained profiling results in accurate cost profiles.
- POET's MILP formulation finds the optimal training schedule through integrated **rematerialization** and **paging.**