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

Shishir G. Patil

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

https://github.com/ShishirPatil/poet
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

**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**

\[ \text{Value preserving Reduce activation} \]
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
• **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.
| Method                              | General Graphs | Compute Aware | Memory Aware | Power Aware |
|------------------------------------|----------------|---------------|--------------|-------------|
| Checkpoint all (PyTorch)           | ✓              | ×             | ×            | ×           |
| Griewank & Walther (2000)          | ×              | ×             | ×            | ×           |
| Chen et al. (2016) \(\sqrt{n}\)  | ×              | ×             | ×            | ×           |
| Chen et al. (2016) greedy          | ×              | ×             | ~            | ×           |
| Checkmate (Jain et al., 2020)      | ✓              | ✓             | ✓            | ×           |
| POFO (Beaumont et al., 2021)       | ×              | ✓             | ✓            | ×           |
| DTR (Kirisame et al., 2021)        | ✓              | ✓             | ✓            | ×           |
| POET (ours)                        | ✓              | ✓             | ✓            | ✓           |
How to reduce the memory and energy requirements of ML training for modern DNN architectures within the constraints of edge devices?
Computational graph

Network Configuration

input

fullc-forward

sigmoid-forward

fullc-forward

softmax-forward
Computational graph

Gradient Calculation Graph

- input
- input-grad
- fullc-forward
- fullc-backward
- sigmoid-forward
- sigmoid-backward
- fullc-forward
- fullc-backward
- softmax-forward
- softmax-backward
- log-loss
- label
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

Forward Pass

Backward Pass

RAM used

Peak RAM

Time

15
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

POET: Training Neural Networks on Tiny Devices
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

Paging:
Page-out to secondary storage and page-in Just-in-Time!
POET: Private Optimal Energy Training
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 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!
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’s integrated Rematerialization and Paging enables training with much smaller memory budgets which was previously not possible!
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.