**ECHO**: Compiler-based GPU Memory Footprint Reduction for LSTM RNN Training

Bojian Zheng\(^1,2\), Nandita Vijaykumar\(^1,3\), Gennady Pekhimenko\(^1,2\)

**Key Results**: \(3 \times\) memory footprint reduction \(\rightarrow 1.35 \times\) faster training

**ECHO** and the MXNet GPU memory profiler are both **open-sourced**

Why LSTM RNN Training is Inefficient?

Training throughput is limited by the GPU memory capacity

Feature maps dominate the GPU memory footprint of the NMT model

Prior works fail to address 2 key challenges:

1. Estimation of memory footprint
2. Runtime overhead

Selective Recomputation
**Echo**: A Selective Recomputation Graph Compiler Pass

- Open-sourced and integrated in MXNet
  - [https://issues.apache.org/jira/browse/MXNET-1450](https://issues.apache.org/jira/browse/MXNET-1450)
- Fully **Automatic & Transparent**
  - Requires NO changes in the training source code
- Addresses 2 key challenges: Estimation of
  1. memory footprint: Bidirectional Dataflow Analysis
  2. runtime overhead: Layer-Specific Optimizations
**ECHO**’s Effect on Memory and Performance

**Baseline:** NO Recomputation, **Mirror:** T. Chen et al. [1], **ECHO:** Our Work

IWSLT15 EN-VI Dataset, Single RTX 2080 Ti GPU

- Baseline $B = 128$
- Mirror $B = 128$
- Echo $B = 128$
- Echo $B = 256$

- **Memory Consumption (GB):**
  - 11 GB Memory Capacity
  - Better

- **Throughput (samples/s):**
  - 1.00 $\times$
  - 0.74 $\times$
  - 0.32 $\times$
  - 0.52 $\times$
  - 1.00 $\times$
  - 0.83 $\times$
  - 0.99 $\times$
  - 1.27 $\times$

- **Performance:**
  - $2 \times$

**DOUBLE WIN!**

Memory Footprint $\downarrow$
Performance $\uparrow$