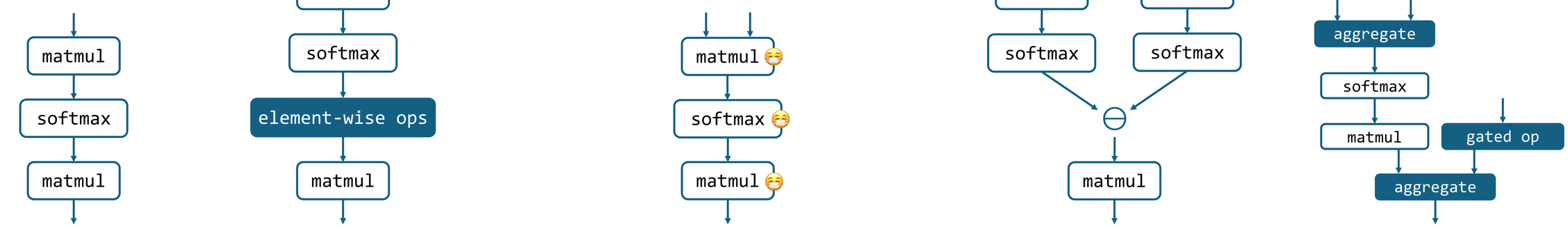
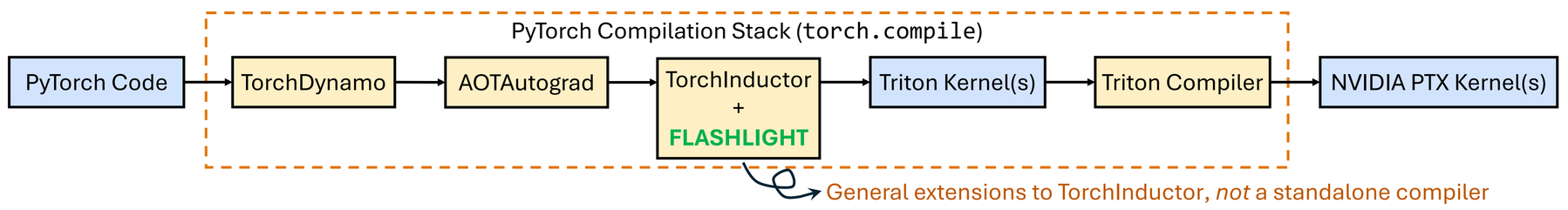
FLASHLIGHT: PyTorch Compiler Extensions to Accelerate Attention Variants

Bozhi You, Irene Wang, Zelal Su Mustafaoglu, Abhinav Jangda, Angélica Moreira, Roshan Dathathri, Divya Mahajan, Keshav Pingali

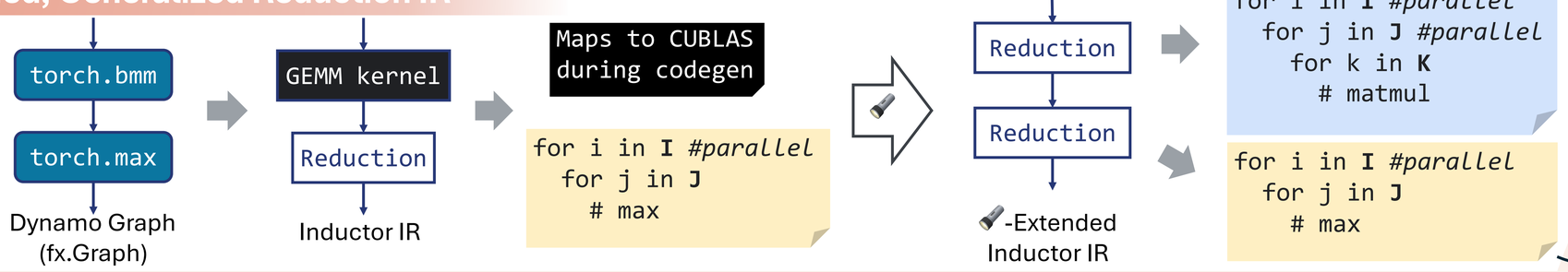
The University of Texas at Austin · Georgia Tech · Microsoft



FLASHLIGHT **generates** optimized, fused kernels from idiomatic PyTorch code



1 Unified, Generalized Reduction IR



2 Algebraic Transformation

$$\text{softmax}(x) = \frac{e^{x-\max x}}{\sum e^{x-\max x}}$$

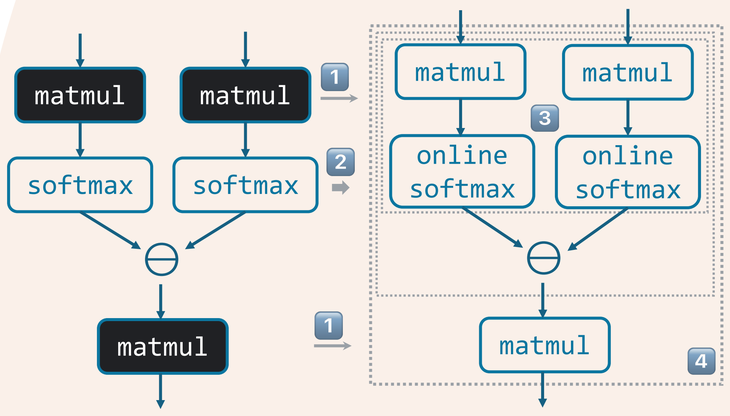
Homomorphism: $h(f(a,b)) = g(h(a), h(b))$

$\exp(\text{sub}(a,b)) = \text{div}(\exp(a), \exp(b))$: $e^{a-b} = \frac{e^a}{e^b}$

```
m = -inf
for j in J
  m = max(m, x[p1])
```

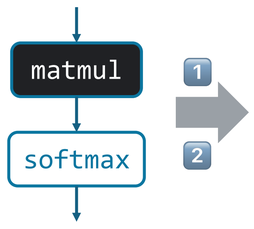
```
s = 0
for j in J
  s = s + exp(x[p1] - m)
```

```
m = -inf
s = 0
for j in J
  m_ = max(m, x[p1])
  s = s * exp(m - m_)
  s = s + exp(x[p1] - m_)
  m = m_
```



$$\text{DiffAttn}(Q[2], K[2], V) = \left(\text{softmax}\left(\frac{Q_1 K_1^T}{\sqrt{d_k}}\right) - \lambda\,\text{softmax}\left(\frac{Q_2 K_2^T}{\sqrt{d_k}}\right)\right) V$$

3 Dimension Demotion




```
for i in I #parallel
  for j in J #parallel
    for k in K
      # matmul
```

```
for i in I #parallel
  for j in J #reduction
    # softmax
```

```
for i in I #parallel
  for j in J #reduction
    for k in K
      # matmul
```

```
for i in I #parallel
  for j in J #reduction
    # softmax
```

```
for i in I #parallel
  for j in J
    for k in K
      # matmul
    # softmax
```

Tradeoff: sacrifices parallelism for I/O efficiency

4 Tiling-Aware Dimension Elimination

```
for i in I #parallel
  for j in J
    for k in K
      # matmul
    # softmax
```

```
for i in I #parallel
  for h in H
    for j in J
      # matmul
```

```
for ti in TI #parallel
  for tj in TJ
    for tk in TK
      # tile of matmul
    # tile of softmax
```

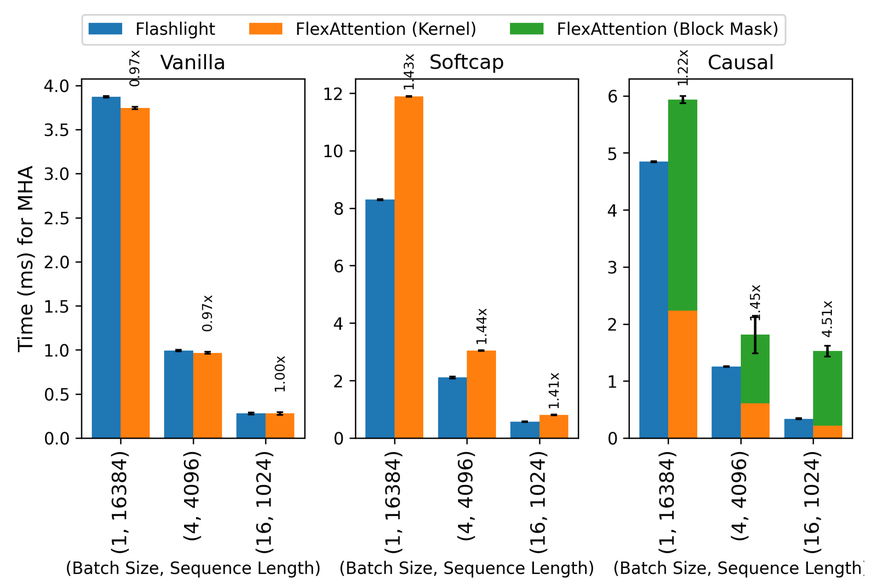
```
for ti in TI #parallel
  for th in TH
    for tj in TJ
      # tile of matmul
```

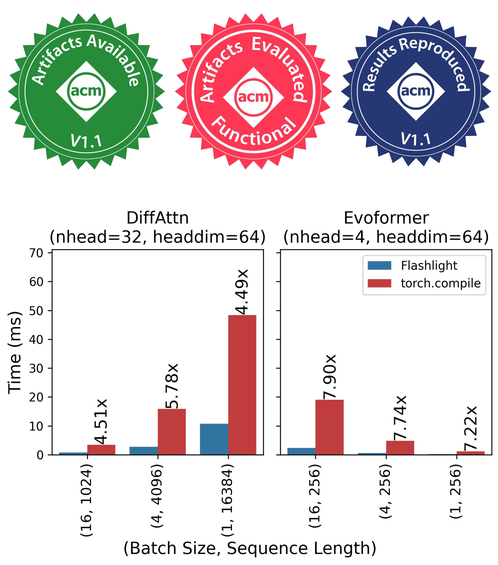
```
for ti in TI #parallel
  for tj in TJ
    for tk in TK
      # tile of matmul
    # tile of softmax
```

```
for ti in TI #parallel
  # |TH| = 1
  for tj in TJ
    # tile of matmul
```

```
for ti in TI #parallel
  for tj in TJ
    for tk in TK
      # tile of matmul
    # tile of softmax
    # tile of matmul
```

When H (head dim of V) is small, set `tile_size_of_H` = H so that $|TH| = 1$ to drop TH loop





Read the paper for more results
- on A100
- for GQA
- for more variants
- for more configurations
- against FlashInfer

Paper

Code

General extensions to PyTorch compiler (`torch.compile`)
- Generates I/O-efficient Triton kernels from idiomatic PyTorch code
- Moves the optimization burden from the user to the compiler

Superior performance
- Always faster than default PyTorch Compiler
- For FlexAttention-supported variants: similar performance
- Beyond FlexAttention: **5x** faster for Evoformer
  - Improves end-to-end inference latency for AlphaFold by **6-9%**