Kaleidoscope: An Efficient, Learnable Representation For All Structured Linear Maps

Tri Dao, Nimit Sohoni, Albert Gu, Matthew Eichhorn,
Amit Blonder, Megan Leszczynski, Atri Rudra, Christopher Ré

Stanford University
University at Buffalo The State University of New York

Structured linear maps with kaleidoscope matrices

- Structured linear maps (low-rank, sparse, DFT, conv…): ubiquitous in ML
- Challenge: they are hand picked
  $\rightarrow$ don't adapt to data, requires domain knowledge
- This talk:
  - Building on theory: how to learn a universal family, Kaleidoscope matrices
  - Applications: improved CNN channel shuffle, simplified speech pipeline…

Structured linear maps are ubiquitous in ML

You've heard of them before…

LDR
Hadamard
DFT
Toeplitz
Sparse
HD
Low-rank
Convolution
Fastfood
ACDC

+ Fast algorithms

+ Few parameters

- Some approximation

Picking the right structure:
→ good tradeoff for memory and speed

Hand-picked structured linear maps are ubiquitous in ML

But they're not easy to pick…

Challenges:

- Doesn't adapt to data
- Requires domain knowledge
- Different implementations

Goals:

- Learnable, integrate with end-to-end ML pipeline
- Expressive family to automate design choices
- Single efficient implementation, ↓ engineering effort

Is there a **learnable, expressive, efficient** representation for all structured linear maps?

Universal Representation for Structure: Outline

1. Background: How to parameterize **structured linear maps?**

    Fast algorithm ↔ Sparse matrix factorization

    Butterfly matrices

2. Kaleidoscope matrices: learnable end-to-end, expressive, and efficient

    Capture **all** structured linear maps (nearly tight # param. & run time).

3. Kaleidoscope matrices: replace **hand-crafted** structure

    Broad applications in vision, speech.

Universal Representation for Structure

1. Background: How to parameterize **structured linear maps?**

    Fast algorithm ↔ Sparse matrix factorization

    Butterfly matrices

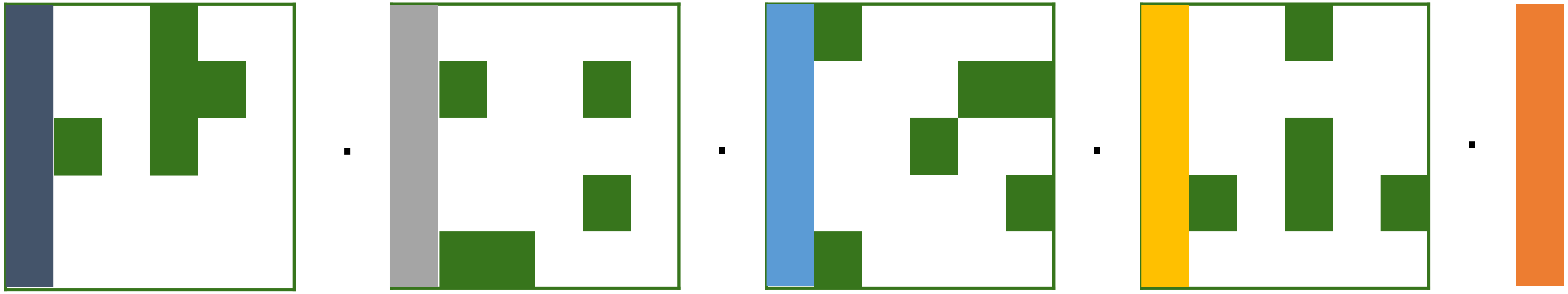
2. Kaleidoscope matrices: **learnable** end-to-end, **expressive, and efficient**

    Capture **all** structured linear maps (nearly tight # param. & run time).

3. Kaleidoscope matrices: replace **hand-crafted** structure

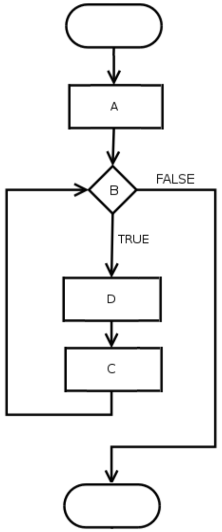
    Broad applications in vision, speech.

Sparse factorization → Fast algorithm

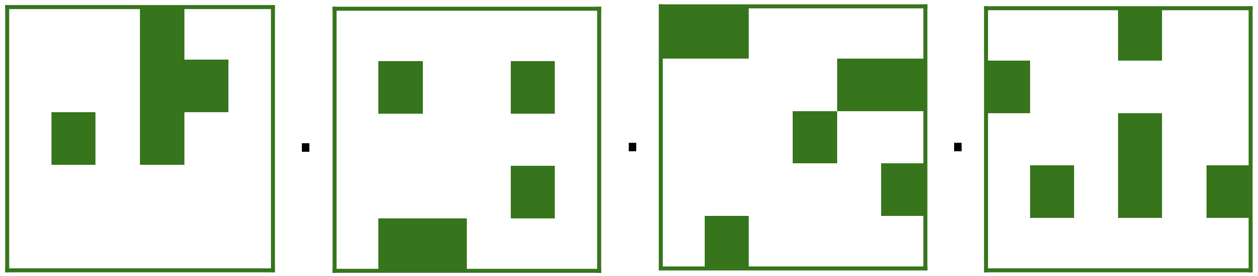
$Ax =$ 

Complexity: O(total nnz of factorization)

Fast algorithm → Sparse factorization



A =



Any A that has
algorithm for A x with
S arithmetic operations
(e.g. add/mult)

Factorization with total nnz O(S)

[Burgisser et al., 2013;
De Sa et al., 2018]

De Sa, C., Gu, A., Puttagunta, R., Ré, C., Rudra, A. A Two-Pronged Progress in Structured Dense Matrix Vector Multiplication. SODA, 2018.

Universal Representation for Structure: Outline

1. Background: How to parameterize **structured linear maps?**

    Fast algorithm ↔ Sparse matrix factorization

    Butterfly matrices

Still difficult to learn

Need inductive bias

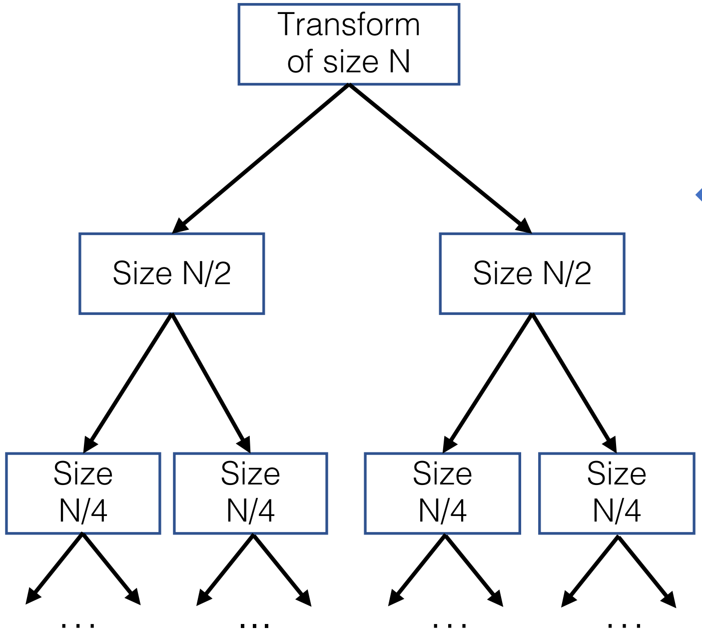
2. Kaleidoscope matrices: **learnable** end-to-end, **expressive, and efficient**

    Capture **all** structured linear maps (nearly tight # param. & run time).

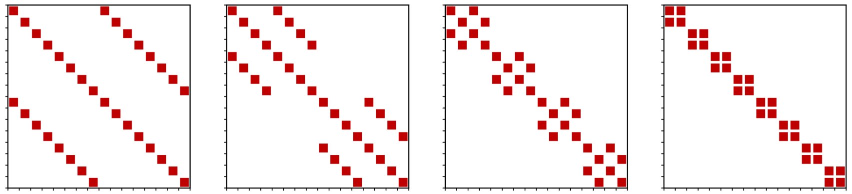
3. Kaleidoscope matrices: replace **hand-crafted** structure

    Broad applications in vision, speech.

Divide-and-Conquer → Butterfly matrices



$$\left(B_N \begin{bmatrix} B_{N/2} & 0 \\ 0 & B_{N/2} \end{bmatrix} \cdots \begin{bmatrix} B_2 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & B_2 \end{bmatrix}\right)$$



[Parker, 1995; Matthieu & LeCun, 2014; Dao et al., 2019]

Recursive divide-and-conquer

[De Sa et al., 2018]

- Trainable with gradient descent on nonzero entries of butterfly matrix.

Captures **recursive divide-and-conquer** structure

Universal Representation for Structure

1. Background: How to parameterize **fast linear maps?**

    Fast algorithm ↔ Sparse matrix factorization

    Butterfly matrices

2. Kaleidoscope matrices: learnable end-to-end, expressive, and efficient

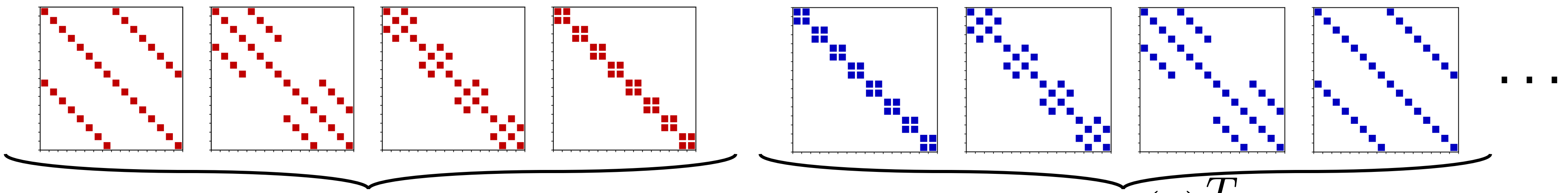
    Capture **all** structured linear maps (nearly tight # param. & run time).

3. Kaleidoscope matrices: replace **hand-crafted** structure

    Broad applications in vision, speech.

Kaleidoscope: Learnable structured matrices

Deep composition of butterfly matrices: $B^{(1)} {B^{(2)}}^T B^{(3)} {B^{(4)}}^T B^{(5)} {B^{(6)}}^T \dots$



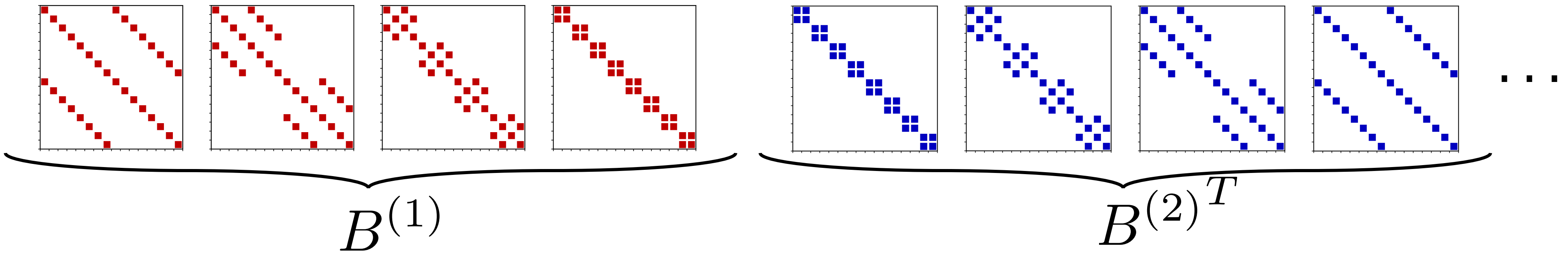
$B^{(1)}$ $\qquad$ ${B^{(2)}}^T$

Butterfly matrix: **Fixed** sparsity

Learnable with gradient descent on nonzero entries of butterfly matrices.

Kaleidoscope hierarchy: Tunable knob

Deep composition of butterfly matrices: $B^{(1)} {B^{(2)}}^T B^{(3)} {B^{(4)}}^T B^{(5)} {B^{(6)}}^T \ldots$



From very compressed $(BB^T)^{O(1)}$ to general matrices $(BB^T)^{O(N)}$

Kaleidoscope hierarchy: Expressiveness

Matrix	Min params / FLOPs	Butterfly params / FLOPs
DFT, DCT, Hadamard, Conv	$\Theta(N \log N)$	$O(N \log N)$
Permutation	$\Theta(N \log N)$	$O(N \log N)$
s-Sparse	$\Theta(s)$	$O(s \log N)$
Rank r	$\Theta(rN)$	$O(rN \log N)$
Arithmetic circuit (s total gates, depth d)	$\Theta(s)$	$O(ds \log s)$

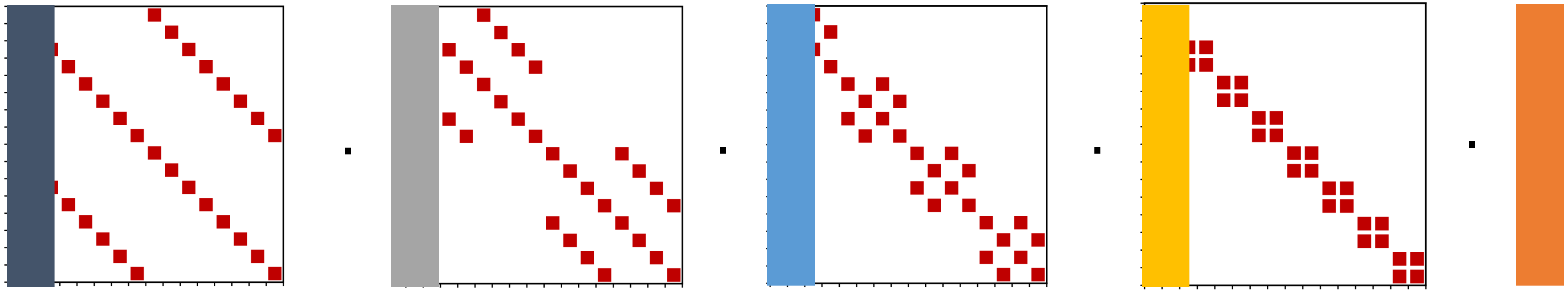
Main theory result [informal]:

Any matrix with a fast multiplication algorithm (i.e. small arithmetic circuit) $\Longleftrightarrow$ Kaleidoscope matrix representation with few parameters

Captures **all** fast linear maps with almost tight parameter count / FLOPs (up to log factor)

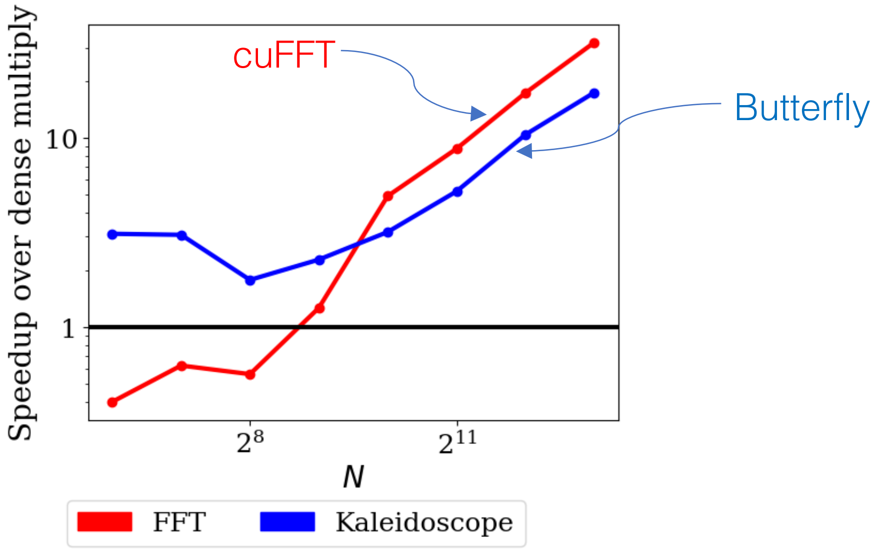
Efficiency

- Each butterfly: 2N log N parameters, O(N log N) multiplication algorithm

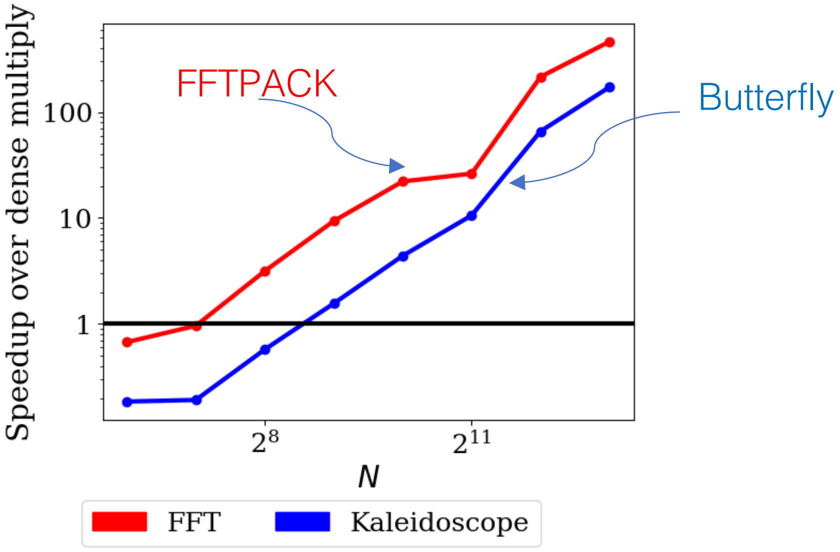
Bx = 

Efficiency

- Each butterfly: 2N log N parameters, O(N log N) multiplication algorithm



Training (GPU): within 2x of cuFFT



Inference (CPU): within 3-5x of FFTPACK

Practically efficient in memory and speed

Universal Representation for Structure

1. Background: How to parameterize **fast linear maps?**

    Fast algorithm ↔ Sparse matrix factorization

    Butterfly matrices

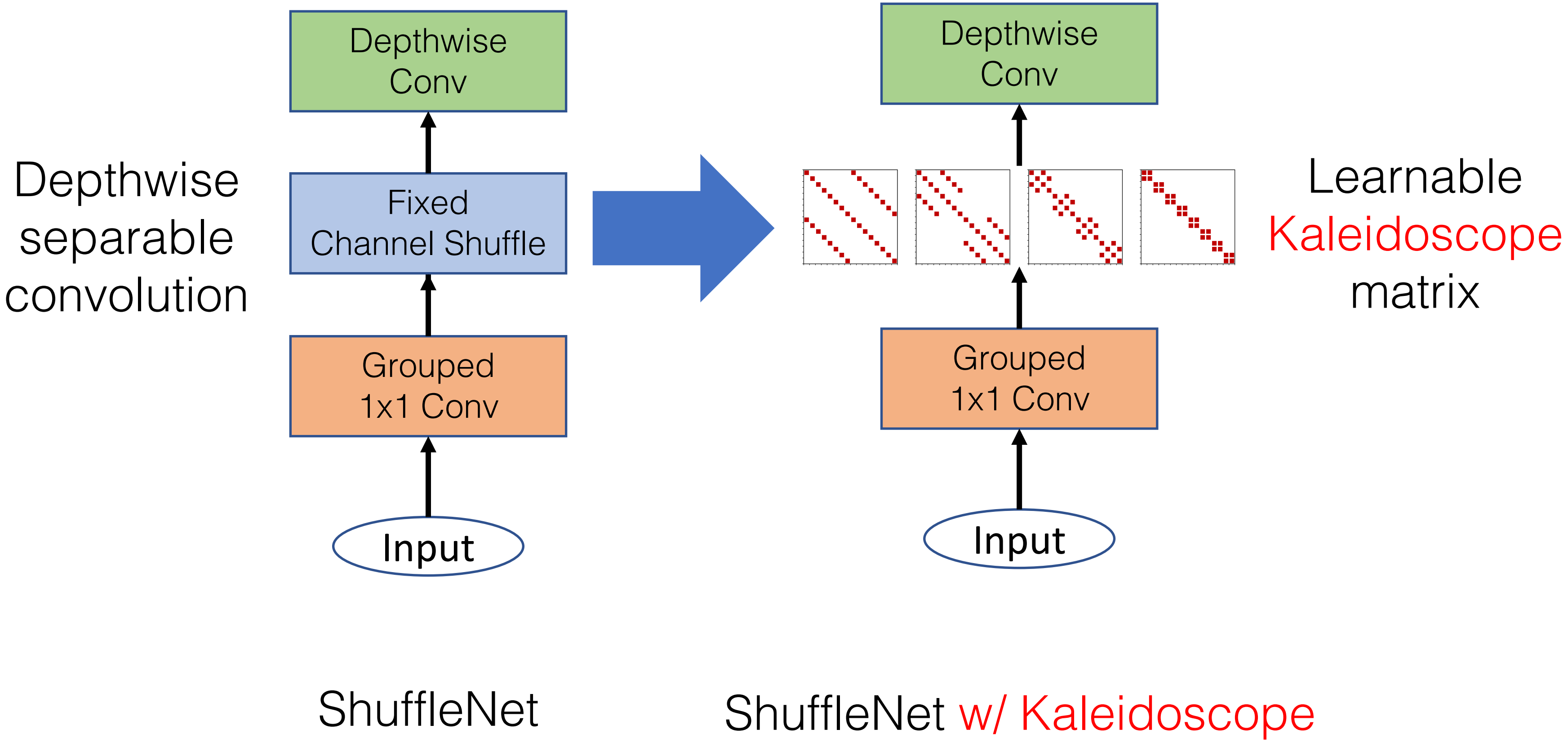
2. Kaleidoscope matrices: **learnable** end-to-end, **efficient** and **expressive**

    Capture **all** structured linear maps (nearly tight # param. & run time).

3. Kaleidoscope matrices: replace **hand-crafted** structure

    Broad applications in vision, speech.

Replacing hand-crafted CNN channel shuffle



Replacing hand-crafted CNN channel shuffle

Image classification results on ImageNet (1.3M images)

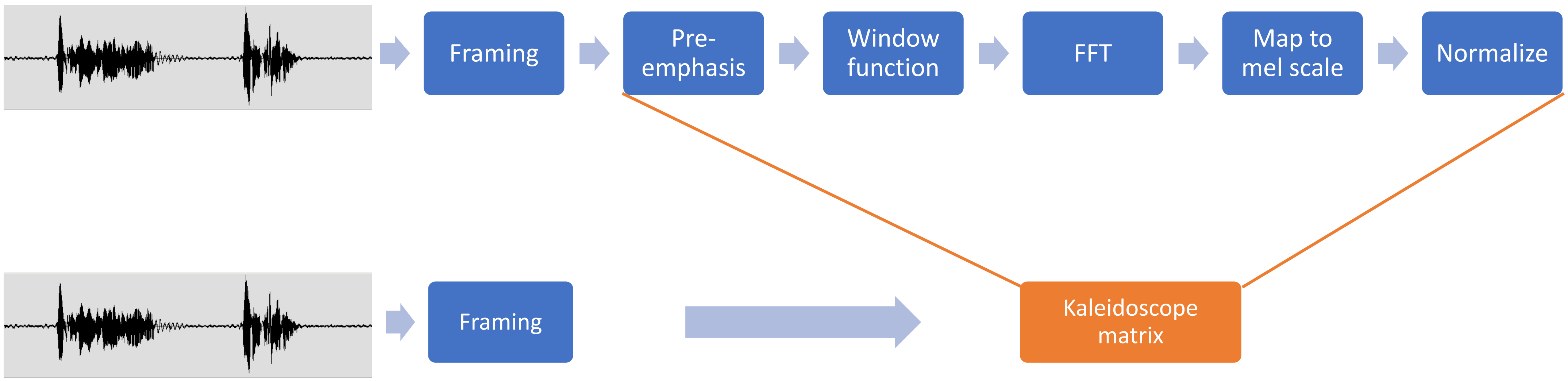
Model	# Params	ImageNet top-1 accuracy	
0.5x width ShuffleNet	1.0M	57.1%	
0.5x width ShuffleNet w/ Kaleidoscope	1.1M	59.5%	
ShuffleNet	2.5M	65.3%	1-2%
ShuffleNet w/ Kaleidoscope	2.8M	66.5%	improvement

Replacing fixed channel shuffle with learnable Kaleidoscope matrices improves accuracy

Simplified speech preprocessing pipeline

Kaleidoscope matrices to replace complicated, hand-engineered preprocessing pipelines

Standard Filter bank/MFSC features



Kaleidoscope pipeline

Kaleidoscope pipeline is competitive with MFSC

Phoneme error rate on TIMIT speech recognition dataset
(lower is better)

Model	# Params	Phoneme Error Rate	
MFSC + LSTM	14.3M	14.2%	
Kaleidoscope + LSTM	15.5M	14.6%	0.4% gap

Much simpler kaleidoscope pipeline is competitive with hand-crafted preprocessing

Universal Representation for Structure: **Summary**

1. Background: How to parameterize **structured linear maps?**

    Fast algorithm ↔ Sparse matrix factorization

    Butterfly matrices

2. Kaleidoscope matrices: learnable end-to-end, expressive, and efficient

    Capture **all** structured linear maps (nearly tight # param. & run time).

3. Kaleidoscope matrices: replace **hand-crafted** structure

    Broad applications in vision, speech.

Thank you! Questions?

- **Code** available at:
  https://github.com/HazyResearch/butterfly/

- **Blog post** (gentle introduction):
  https://dawn.cs.stanford.edu/2019/06/13/butterfly/

Tri Dao

trid@stanford.edu