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
Structured linear maps with kaleidoscope matrices

- Structured linear maps (low-rank, sparse, DFT, conv…): ubiquitous in ML

- **Challenge:** they are hand picked
  → 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...

+ 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 $\leftrightarrow$ 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 $\rightarrow$ Fast algorithm

$$Ax = \cdot \cdot \cdot$$

Complexity: $O(\text{total nnz of factorization})$
Fast algorithm → Sparse factorization

Any $A$ that has an algorithm for $A \times \text{vector}$ with $S$ arithmetic operations (e.g. add/multiply)

Factorization with total nnz $O(S)$

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

Still difficult to learn
Need inductive bias
Divide-and-Conquer $\rightarrow$ Butterfly matrices

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

\[
\begin{pmatrix}
B_N & B_{N/2} & 0 & \cdots & B_2 & \cdots & 0 \\
0 & B_{N/2} & \cdots & \cdots & \cdots & \cdots & \cdots \\
\vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots \\
0 & 0 & \cdots & \cdots & B_2 & \cdots & 0
\end{pmatrix}
\]

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

$$B^{(1)} B^{(2)^T} B^{(3)} B^{(4)^T} B^{(5)} B^{(6)^T} \ldots$$

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

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

Main theory result [informal]:

Any matrix with a fast multiplication algorithm (i.e. small arithmetic circuit) \(\iff\) 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
Efficiency

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

Practically efficient in memory and speed
Universal Representation for Structure

1. Background: How to parameterize fast linear maps?
   Fast algorithm $\leftrightarrow$ 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

Depthwise separable convolution

Fixed Channel Shuffle

Grouped 1x1 Conv

Input

Depthwise Conv

Learnable Kaleidoscope matrix

Grouped 1x1 Conv

Input

ShuffleNet

ShuffleNet w/ Kaleidoscope
Replacing hand-crafted CNN channel shuffle

Image classification results on ImageNet (1.3M images)

<table>
<thead>
<tr>
<th>Model</th>
<th># Params</th>
<th>ImageNet top-1 accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5x width ShuffleNet</td>
<td>1.0M</td>
<td>57.1%</td>
</tr>
<tr>
<td>0.5x width ShuffleNet w/ Kaleidoscope</td>
<td>1.1M</td>
<td>59.5%</td>
</tr>
<tr>
<td>ShuffleNet</td>
<td>2.5M</td>
<td>65.3%</td>
</tr>
<tr>
<td>ShuffleNet w/ Kaleidoscope</td>
<td>2.8M</td>
<td>66.5%</td>
</tr>
</tbody>
</table>

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 is competitive with MFSC

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

<table>
<thead>
<tr>
<th>Model</th>
<th># Params</th>
<th>Phoneme Error Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>MFSC + LSTM</td>
<td>14.3M</td>
<td>14.2%</td>
</tr>
<tr>
<td>Kaleidoscope + LSTM</td>
<td>15.5M</td>
<td>14.6%</td>
</tr>
</tbody>
</table>

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