DeepMind's AlphaTensor:

Discovering Faster Matrix Multiplication Algorithms with Reinforcement Learning

J. Setpal

November 15, 2022

Outline

1. Task Background
2. Algorithms as Tensor Decomposition
3. Neural Architecture Design

Outline

What is Matrix Multiplication?

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \cdot \begin{bmatrix} 2 & 2 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 4 & 4 \\ 10 & 10 \end{bmatrix} \tag{1}$$

What is Matrix Multiplication?

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \cdot \begin{bmatrix} 2 & 2 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 4 & 4 \\ 10 & 10 \end{bmatrix} \tag{1}$$

<u>Problem:</u> It's too complex! $O(n^3)$.

What is Matrix Multiplication?

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \cdot \begin{bmatrix} 2 & 2 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 4 & 4 \\ 10 & 10 \end{bmatrix} \tag{1}$$

Problem: It's too complex! $O(n^3)$.

Optimization Ideas:

- Human Search
- Continuous Optimization
- Combinatorial Search

What is Matrix Multiplication?

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \cdot \begin{bmatrix} 2 & 2 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 4 & 4 \\ 10 & 10 \end{bmatrix} \tag{1}$$

Problem: It's too complex! $O(n^3)$.

Optimization Ideas:

- Human Search
- Continuous Optimization
- Combinatorial Search

Problem: Suboptimal!

What are Tensors?

Matrices, but *generalized*.

So, let's automate the Search Process!

There's a couple of considerations:

- Representing tensors as an operation.

So, let's automate the Search Process!

There's a couple of considerations:

- Representing tensors as an operation.
- MASSIVE Search Space: 10^{12} potential actions.

So, let's automate the Search Process!

There's a couple of considerations:

- Representing tensors as an operation.
- MASSIVE Search Space: 10^{12} potential actions.
- Impossible to obtain every state tree.

So, let's automate the Search Process!

There's a couple of considerations:

- Representing tensors as an operation.
- MASSIVE Search Space: 10^{12} potential actions.
- Impossible to obtain every state tree.
- Hardwase basis for optimization.

So, let's automate the Search Process!

There's a couple of considerations:

- Representing tensors as an operation.
- MASSIVE Search Space: 10^{12} potential actions.
- Impossible to obtain every state tree.
- Hardwase basis for optimization.

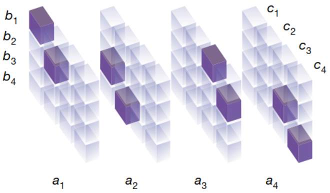
Outline

1. Task Background
2. Algorithms as Tensor Decomposition
3. Neural Architecture Design

Let's Restructure the Problem!

a

$$\begin{pmatrix} c_1 & c_2 \\ c_3 & c_4 \end{pmatrix} = \begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix} \cdot \begin{pmatrix} b_1 & b_2 \\ b_3 & b_4 \end{pmatrix}$$



b

$$m_1 = (a_1 + a_4)(b_1 + b_4)$$
$$m_2 = (a_3 + a_4)\, b_1$$
$$m_3 = a_1\, (b_2 - b_4)$$
$$m_4 = a_4\, (b_3 - b_1)$$
$$m_5 = (a_1 + a_2)\, b_4$$
$$m_6 = (a_3 - a_1)(b_1 + b_2)$$
$$m_7 = (a_2 - a_4)(b_3 + b_4)$$
$$c_1 = m_1 + m_4 - m_5 + m_7$$
$$c_2 = m_3 + m_5$$
$$c_3 = m_2 + m_4$$
$$c_4 = m_1 - m_2 + m_3 + m_6$$

c

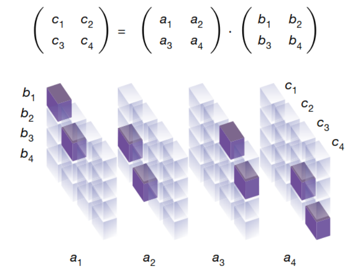
$$\mathbf{U} = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & -1 \end{pmatrix}$$

$$\mathbf{V} = \begin{pmatrix} 1 & 1 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & -1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

$$\mathbf{W} = \begin{pmatrix} 1 & 0 & 0 & 1 & -1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & -1 & 1 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Let's Restructure the Problem!

a



b

$$
\begin{aligned}
m_1 &= (a_1 + a_4)(b_1 + b_4) \\
m_2 &= (a_3 + a_4)\, b_1 \\
m_3 &= a_1\, (b_2 - b_4) \\
m_4 &= a_4\, (b_3 - b_1) \\
m_5 &= (a_1 + a_2)\, b_4 \\
m_6 &= (a_3 - a_1)(b_1 + b_2) \\
m_7 &= (a_2 - a_4)(b_3 + b_4) \\
c_1 &= m_1 + m_4 - m_5 + m_7 \\
c_2 &= m_3 + m_5 \\
c_3 &= m_2 + m_4 \\
c_4 &= m_1 - m_2 + m_3 + m_6
\end{aligned}
$$

c

$$
\mathbf{U} = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & -1 \end{pmatrix}
$$

$$
\mathbf{V} = \begin{pmatrix} 1 & 1 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & -1 & 0 & 1 & 0 & 1 \end{pmatrix}
$$

$$
\mathbf{W} = \begin{pmatrix} 1 & 0 & 0 & 1 & -1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & -1 & 1 & 0 & 0 & 1 & 0 \end{pmatrix}
$$

This can then be represented using:

$$
\mathcal{T}_n = \sum_{r=1}^{R} u^{(r)} \otimes v^{(r)} \otimes w^{(r)}
$$

Outline

1. Task Background
2. Algorithms as Tensor Decomposition
3. Neural Architecture Design

Attention is All It Needs!

Leveraging the Current tensor and the history, embedding, policy and value head.

Synthetic Demonstration

Decomposing a 3D tensor is NP-hard (it's the challenge).

Synthetic Demonstration

Decomposing a 3D tensor is NP-hard (it's the challenge).

However, the reverse is trivial!

Synthetic Demonstration

Decomposing a 3D tensor is NP-hard (it's the challenge).

However, the reverse is trivial!

Randomly selected tensor pairs are used to compose a 3D-tensor. This generates synthetic data on which the AlphaTensor agent trains.

Change of Basis

Change the basis is similar to a linear operation.

Change of Basis

Change the basis is similar to a linear operation.

It's the same transofmation, but the algorithm interprets it as a new input.

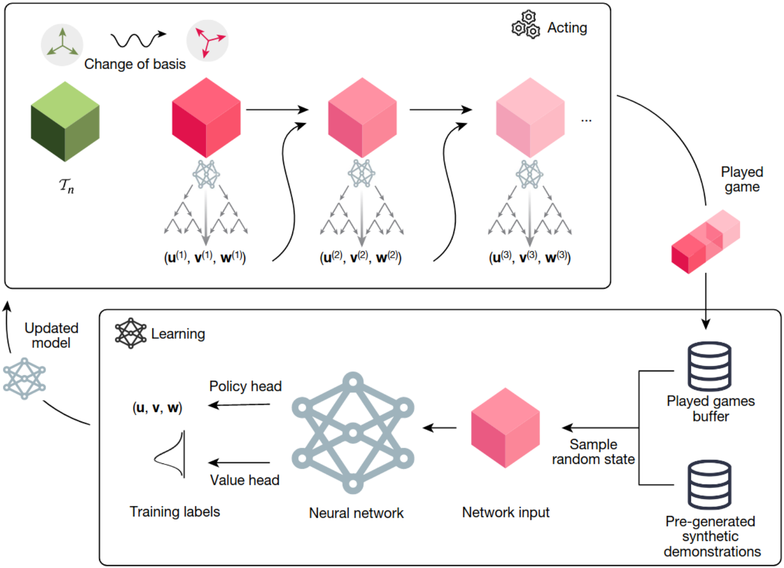
Change of Basis

Change the basis is similar to a linear operation.

It's the same transofmation, but the algorithm interprets it as a new input.

Easy way to augment existing data!

Putting it Together



Thank you!

Have an awesome rest of your day!

Slides: `https://cs.purdue.edu/homes/jsetpal/alphatensor.pdf`