

Computation for Foundation Models

Tensor Parallelism

authors

November 19, 2025

Outline

Introduction

More Content

Slide Title

This is your first content slide.

- ▶ Point 1
- ▶ Point 2
- ▶ Point 3

What is a foudation model

A foundational model is a machine/ deep learning model that can be applied to a wide range of use cases. Examples include GPT, stable diffusion, and DALL-E.

Simplified Foundation Model

To begin with, we will only look at what happens to the input of a foundation model, and not the process of it backtracking and learning. This version of a "model" has a lot stripped away from it for the purpose of explaining, but the methods shown here not only can be used for a full model, but are actively being used in existing models.

$$y = f_n(f_{n-1}(\dots f_1(X)))$$

$$f_i = \sigma(W_i X + b_i)$$

Our input X is a 2d matrix, the W 's are linear transformations so they can be represented as matrices, and σ is a nonlinear function, and y is the output.

Tensor Parallelism

Tensor parallelism is a way of computing our large from the previous slide across multiple devices, or gpus, using linear algebra tricks to split up the computations.

Tensor Parallelism (1D)

For our 1D case, X being a matrix doesn't actually matter and we can simplify how we think about it to just considering the processing of each column of X , which we will label x . So for 1D, our model simplifies to $f_i = \sigma(W_i x + b_i)$.

$$WX = W [x_1 \quad x_2 \quad \cdots \quad x_r] = [Wx_1 \quad Wx_2 \quad \cdots \quad Wx_r]$$

In other words, we will think of x as being a single column vector for our 1D tensor parallelism cases.

Tensor Parallelism(Row)

We can split the weight matrix W_i into groups of rows depending on how many devices we have. Each entry of the k th result of $W_i x$ is equal to the k th row of W_i times x , so we just have each device compute $C_j x$. Since each device is fully responsible for specific entries of the result, we can also give the corresponding entries of b_i , which we will label b_{L_j} to each device and apply our nonlinear function. We end up with each device computing $\sigma(C_j x + b_{L_j})$, and then gathering the various elements from each device into a single vector to be fed into the next function

$$W_i = \begin{bmatrix} C_1 \\ C_2 \\ \vdots \\ C_m \end{bmatrix}$$

Tensor Parallelism(Column)

We can split the weight matrix W_i into smaller matrices depending on the number of devices we have. Each A_j is stored on a different device, and along with it we send the entries of x that correspond to the columns of W_i that are held within each A_j as a column vector, x_{L_j} .

$$W_i = [A_1 \quad A_2 \quad \dots \quad A_3]$$

Each device is then computing $A_j x_{L_j}$ which we can think of as

$$A_j x_{L_j} = \begin{bmatrix} a_{11}(j) & a_{12}(j) & \dots & a_{1k}(j) \\ a_{21}(j) & a_{22}(j) & \dots & a_{2k}(j) \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}(j) & a_{m2}(j) & \dots & a_{mk}(j) \end{bmatrix} \begin{bmatrix} x_p \\ x_p + 1 \\ \vdots \\ x_p + k - 1 \end{bmatrix}$$

Tensor Parallelism(Column) pt 2

2D Tensor Parallelism

Suppose we have $WX + B$ and q^2 devices. We can split the matrices W and X into q^2 blocks each and get

$$W = \begin{bmatrix} W_{11} & W_{12} & \cdots & W_{1q} \\ W_{21} & W_{22} & \cdots & W_{2q} \\ \vdots & \vdots & \ddots & \vdots \\ W_{q1} & W_{q2} & \cdots & W_{qq} \end{bmatrix}, \quad X = \begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1q} \\ X_{21} & X_{22} & \cdots & X_{2q} \\ \vdots & \vdots & \ddots & \vdots \\ X_{q1} & X_{q2} & \cdots & X_{qq} \end{bmatrix}.$$

$$Y = WX = \begin{bmatrix} \sum_{k=1}^q W_{1k}X_{k1} & \sum_{k=1}^q W_{1k}X_{k2} & \cdots & \sum_{k=1}^q W_{1k}X_{kq} \\ \sum_{k=1}^q W_{2k}X_{k1} & \sum_{k=1}^q W_{2k}X_{k2} & \cdots & \sum_{k=1}^q W_{2k}X_{kq} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{k=1}^q W_{qk}X_{k1} & \sum_{k=1}^q W_{qk}X_{k2} & \cdots & \sum_{k=1}^q W_{qk}X_{kq} \end{bmatrix} + B.$$

2D Tensor Parallelism pt 2 (SUMMA algorithm)

Performance Statistics

TODO: big O notation comparison

Relevancy and Impact

As AI is becoming increasingly more relevant, and increasingly more complex, large HPC's are being built with huge budgets to be used for these models, but then comes the challenge of actually having ways to utilize these large machines with the models.

Real world uses

Tensor parallelism is a very important technique being used in almost all foundation model training.

Thank You

Questions?