NEURAL INFORMATION PROCESSING SYSTEMS

A Tighter Complexity Analysis of SparseGPT

Xiaoyu Li, Yingyu Liang, Zhenmei Shi, Zhao Song

STEVENS INSTITUTE of TECHNOLOGY THE INNOVATION UNIVERSITY · 香 港 大 學 THE UNIVERSITY OF HONG KONG · THE UNIVERSITY of WISCONSIN MADISON · Berkeley UNIVERSITY OF CALIFORNIA

Background

Frantar and Alistarh (2023) developed the algorithm SparseGPT to uses calibration data to prune the parameters of GPT-family models in one-shot. It can prune at least 50% parameters with structure patterns, while the perplexity increase is negligible. Thus, SparseGPT can reduce the running time and GPU memory usage while keeping high performance for LLMs' applications. However, they only gives a loose bound on the time complexity of the algorithm, which is $O(d^3)$ where d is the the model's hidden feature dimension.

Algorithm 1 The SparseGPT algorithm (Algorithm 1 in [FA23]).

procedure SparseGPT($p \in [0,1]$, $W \in \mathbb{R}^{d\times d}$, $X \in \mathbb{R}^{d\times d}$, $B \in \mathbb{N}_+$, $B_s \in \mathbb{N}_+$, $\lambda > 0$)
- ▷ Pruning ration $p \in [0,1]$
- ▷ Weight matrix $W \in \mathbb{R}^{d\times d}$
- ▷ Input feature matrix $X \in \mathbb{R}^{d\times d}$
- ▷ Lazy update block size $B \in \mathbb{N}_+$, $B = d^a$ for any $a \in [0,1]$
- ▷ Adaptive mask size $B_s \in \mathbb{N}_+$
- ▷ Regularization parameter $\lambda > 0$

    $M,\ E \leftarrow \mathbf{1}_{d\times d},\ \mathbf{0}_{d\times B}$ ▷ $O(d^2)$
    $\widetilde{H} \leftarrow (XX^\top + \lambda I_{d\times d})^{-1}$ ▷ $O(d^\omega)$
    for $i = 0, B, 2B, \dots, \lfloor \frac{d}{B} \rfloor B$ **do**
        for $j = i+1, \dots, i+B$ **do**
            if $j \bmod B_s = 0$ **then** ▷ $O(d)$
                $M_{*,[j,j+B_s]} \leftarrow$ MaskSelect($p, W_{*,[j,j+B_s]}, \widetilde{H}, j-1$) ▷ $O(d^2 \log d)$
            end if
            $E_{*,j-i} \leftarrow (\mathbf{1}_{d\times 1} - M_{*,j}) \circ W_{*,j} / \widetilde{H}_{j,j}$ ▷ $O(d^2)$
            $W_{*,[j,i+B]} \leftarrow W_{*,[j,i+B]} - E_{*,j-i} \widetilde{H}_{j,[j,i+B]}$ ▷ $O(d^{2+a})$
        end for
        $W_{*,[i+B,d]} \leftarrow W_{*,[i+B,d]} - E\widetilde{H}_{[i,i+B],[i+B,d]}$ ▷ $O(d^{1+\omega(1,1,a)-a})$
    end for
    $W \leftarrow W \circ M$ ▷ $O(d^2)$
end procedure

procedure MaskSelect($p \in [0,1]$, $W' \in \mathbb{R}^{d\times r}$, $\widetilde{H} \in \mathbb{R}^{d\times d}$, $s \in \mathbb{N}_+$)
- ▷ Sub-weight matrix $W' \in \mathbb{R}^{d\times r}$; Inverse of Hessian matrix $\widetilde{H} \in \mathbb{R}^{d\times d}$
- ▷ Index $s \in \mathbb{N}_+$, recording the position of W' in W

    $M' \leftarrow \mathbf{0}_{d\times r}$ ▷ $O(dr)$
    for $k = 1, \dots, r$ **do**
        $w \leftarrow W'_{*,k}$ ▷ $w \in \mathbb{R}^d$, $O(dr)$
        $w \leftarrow (w \circ w) / (\widetilde{H}_{s+k,s+k})^2$ ▷ $O(dr)$
        $J \leftarrow$ indices of top $(1-p)d$ largest entries of w ▷ $O(r \cdot d \log d)$ by sorting
        for $j \in J$ **do**
            $M'_{k,j} \leftarrow 1$ ▷ $O(dr)$
        end for
    end for
    return M'
end procedure

Fast Matrix Multiplication and Lazy Update

Definition 1. For three integers d_1, d_2, d_3, we use $\mathcal{T}_{\mathrm{mat}}(d_1, d_2, d_3)$ to denote the time of multiplying a $d_1 \times d_2$ matrix and a $d_2 \times d_3$ matrix.

Fact 2. It holds that $\mathcal{T}_{\mathrm{mat}}(d_1, d_2, d_3) = \mathcal{T}_{\mathrm{mat}}(d_1, d_3, d_2) = \mathcal{T}_{\mathrm{mat}}(d_2, d_1, d_3)$.

Definition 3 (Exponent of Matrix Multiplication). For $a, b, c > 0$, we use $d^{\omega(a,b,c)}$ to denote the time of multiplying a $d^a \times d^b$ matrix and a $d^b \times d^c$ matrix. We denote $\omega := \omega(1,1,1)$ as the exponent of matrix multiplication.

Definition 4 (Dual Exponent of Matrix Multiplication). We use α to denote the dual exponent of matrix multiplication, which is the largest value such that $\omega(1, \alpha, 1) = 2 + o(1)$.

Lemma 5 (Current Values). Currently, $\omega \approx 2.731, \alpha \approx 0.321$.

The idea of **lazy update** comes from an interesting fact of fast rectangular matrix multiplication: the time complexity of multiplying a $d \times d$ matrix by a $d \times 1$ matrix is the same as the times complexity of multiplying a $d \times d$ matrix by a $d \times d^a$ matrix for any nonnegative $a \le \alpha$, where α is the dual exponent of matrix .

Main Result

Theorem (Main Results). Let lazy update block size $B = d^a$ for $a \in [0,1]$. The running time of SparseGPT is

$$O(d^\omega + d^{2+a+o(1)} + d^{1+\omega(1,1,a)-a}).$$

Under the current values $\omega \approx 2.731, \alpha \approx 0.321$, the running boils down to

$$O(d^{2.53}).$$