VE472 Lecture 13

Jing Liu

UM-SJTU Joint Institute

Summer

- Many methods can be understood using various matrix decompositions, thus approximating the product of two matrices is a way to deal with large data.

$$\mathbf{AB} \approx \mathbf{GF}$$

- Intuitively, it means $\mathbf{AB}$ is close to $\mathbf{GF}$, but we need to define it precisely.
- Recall the notion of norm allows to talk about $\mathbf{x} \in \mathbb{R}^n$ is close to $\mathbf{y} \in \mathbb{R}^n$

$$\mathbf{x} \approx \mathbf{y} \quad \text{if} \quad \|\mathbf{x} - \mathbf{y}\| < \varepsilon$$

  for some small positive $\varepsilon \in \mathbb{R}$, and the norm of $\mathbf{x} - \mathbf{y}$ give how close they are.
- Therefore, we need to discuss matrix norms as well as vector norms!
- We have used largely the ℓ_2-norm for any $\mathbf{u} \in \mathbb{R}^n$ in this course,

$$\|\mathbf{u}\| = \sqrt{\sum_{i=1}^{n} u_i^2}$$

vector norm

- For a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, one useful matrix norm is the Frobenious norm

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij}^2}$$

- Another important matrix norm is given by

operator norm $$\|\mathbf{A}\|_O = \max_{\hat{\mathbf{x}}} \|\mathbf{A}\hat{\mathbf{x}}\| \qquad \text{where} \quad \|\hat{\mathbf{x}}\| = 1$$ unit vector

vector norm

which is called the operator norm induced by the vector norm $\|\cdot\|$ on $\mathbb{R}^n$.

- It can be shown the two matrix norms are related to each other as

$$\|\mathbf{A}\|_O \geq \|\mathbf{A}\|_F \geq \sqrt{n}\|\mathbf{A}\|_O$$

- So the problem we want to solve is whether we can find a $\mathbf{G}$ and $\mathbf{F}$ such that

$$\|\mathbf{AB} - \mathbf{GF}\|_F \qquad \text{or} \qquad \|\mathbf{AB} - \mathbf{GF}\|_O$$

is small, and computationally cheaper to find and compute $\mathbf{GF}$ in general.

Algorithm 1: Naive three-loop algorithm for matrix multiplication

Input : Matrix $\mathbf{A}$ of $m \times p$, and matrix $\mathbf{B}$ of $p \times n$.

Output: The product $\mathbf{M} = \mathbf{AB}$

```
Function NaiveMatrixMultiplication(A, B):
    for i ← 1 to m do          rows of first matrix
        for j ← 1 to n do      columns of second matrix
            [M]_ij ← 0 ;
            for ℓ ← 1 to p do
                [M]_ij ← [M]_ij + [A]_iℓ [B]_ℓj ;
            end for
        end for
    end for
    return M ;
end
```

- Notice the naive approach essentially computes the matrix multiplication as

$$m \times n \times p$$

pairs of products that need to be summed together, or as

$$m \times n$$

pairs of inner/dot products involving row and column vectors in $\mathbb{R}^p$.

- The idea is to take a sample of all computations needed to reduce the cost!
- Some of those computations are more important than other others so should be selected with a higher probability, we could also gain the insight from

$$A = \sum_{i=1}^{n} x_i = \mu \times n \approx \hat{x} \times L$$

how many samples taken

where μ is the population mean while $\hat{x}$ is the sample mean.

- We could sample according to any probability distribution, including simple random sampling, and obtain an unbiased estimator of the mean/sum, but...

data大的时候可以让bias大一点->更小的var

- In order to better implement this idea of using a non-uniform distribution to minimise the variance, we need to see the matrix multiplication as

see notes

p x n

$$\mathbf{AB} = \sum_{k=1}^{p} \mathbf{a}_k B_k$$

matrix multiplication by outer product

where $\mathbf{a}_k$ denotes the kth column of $\mathbf{A}$, while B_k denotes the kth row of $\mathbf{B}$.

- This essentially decomposes the product into p outer products, each needs

$$m \times n$$

scalar multiplications, the idea is to find a sample of those p outer products.

$$\mathbf{AB} = \sum_{k=1}^{p} \mathbf{a}_k B_k \approx \frac{1}{L} \sum_{\ell=1}^{L} \frac{1}{q_{k_\ell}} \mathbf{a}_{k_\ell} B_{k_\ell} = \sum_{\ell=1}^{L} \mathbf{g}_{k_\ell} F_{k_\ell} = \mathbf{GF}$$

sample size

probability of a particular block being calculated

Q: What are the mean and variance of the ijth element of $[\mathbf{GF}]$?

Algorithm 2: Basic sampling algorithm for matrix multiplication

Input : Matrix $\mathbf{A}$ of $m \times p$, matrix $\mathbf{B}$ of $p \times n$, positive integer L, and list of probabilities $\{q_i\}_{i=1}^{p}$.

Output : Matrices $\mathbf{G}$ and $\mathbf{F}$ such that $\mathbf{GF} \approx \mathbf{AB}$

```
Function BasicSamplingMatrixMultiplication(A, B):
    for ℓ ← 1 to L do
        Sample k_ℓ ∈ {1, ..., p} with probability Pr(k_ℓ = i) = q_i
        /* The above step is done independently and identically
           with replacement from one ℓ value to another.        */
        for i ← 1 to m do
            [G]_{iℓ} ← [A]_{iℓ} / sqrt(L q_{k_ℓ}) ;
        end for
        for j ← 1 to n do
            [F]_{ℓj} ← [B]_{ℓj} / sqrt(L q_{k_ℓ}) ;
        end for
    end for
    return M ;
end
```

Theorem 0.1

Suppose matrices $\mathbf{A}$ and $\mathbf{B}$, and the matrices $\mathbf{G}$ and $\mathbf{F}$ generated by Algo 2 , then

$$\mathbb{E}\left[\|\mathbf{AB}-\mathbf{GF}\|_F^2\right]=\sum_{k=1}^{p}\frac{\|\mathbf{a}_k\|^2\|B_k\|^2}{Lq_k}-\frac{1}{L}\|\mathbf{AB}\|_F^2$$

expected error

Furthermore, if

$$q_k=\frac{\|\mathbf{a}_k\|\|B_k\|}{A} \qquad \textit{where} \quad A=\sum_{k=1}^{p}\|\mathbf{a}_k\|\|B_k\|$$

then the expected Frobenious norm of the error is minimised, and the value is

$$\mathbb{E}\left[\|\mathbf{AB}-\mathbf{GF}\|_F^2\right]=\frac{1}{L}\left(\sum_{k=1}^{p}\|\mathbf{a}_k\|\|B_k\|\right)^2-\frac{1}{L}\|\mathbf{AB}\|_F^2$$

- Notice the expected error is the sum of all variances $\mathrm{Var}\left[[\mathbf{GF}]_{ij}\right]$.

error of using GF instead of AB is sum of GF's variance