Parallel coordinate descent for the AdaBoost problem

Olivier Fercoq, olivier.fercoq@ed.ac.uk

Introduction

The AdaBoost algorithm [2] is a widely used classification algorithm. Its goal is to combine many weak hypotheses with high error rate to generate a single strong hypothesis with very low error.

We propose here a randomised parallel coordinate descent to decrease at most this error.

We will also consider the following equivalent objective function with Lipschitz gradient

\[ F(\lambda) = \log(f(A\lambda)), \]

and its associated C^1,1 AdaBoost problem

\[ \inf_{\lambda \in \mathbb{R}^m} \frac{1}{m} \sum_{j=1}^{m} \exp(A_j^T \lambda) = \inf_{\lambda \in \mathbb{R}^m} f(A\lambda) = f_A, \]

where \( f(x) = \frac{1}{m} \sum_{j=1}^{m} \exp(x_j) \).

We will also consider the following equivalent objective function with Lipschitz gradient

\[ F(\lambda) = \log(f(A\lambda)), \]

and its associated C^1,1 AdaBoost problem

\[ \inf_{\lambda \in \mathbb{R}^m} F(\lambda). \]

Classically [2], this problem is solved by greedy coordinate descent. At each iteration, one selects the classifier with the largest error and updates its weight in order to decrease at most this error.

We propose here a randomised parallel coordinate descent method to solve this optimisation problem.

The AdaBoost problem

Let \( M \in \mathbb{R}^{n \times m} \) be a matrix of features, \( y \in \mathbb{R}^n \) be a vector of labels and \( A_{ij} = y_i M_{ij} \).

The AdaBoost problem is the minimisation of the exponential loss:

\[ \inf_{\lambda \in \mathbb{R}^m} \frac{1}{n} \sum_{i=1}^{n} \exp(A_i^T \lambda) = \inf_{\lambda \in \mathbb{R}^m} f(\lambda) = f_A, \]

where \( f(x) = \frac{1}{m} \sum_{j=1}^{m} \exp(x_j) \).

We will also consider the following equivalent objective function with Lipschitz gradient

\[ F(\lambda) = \log(f(A\lambda)), \]

and its associated C^1,1 AdaBoost problem

\[ \inf_{\lambda \in \mathbb{R}^m} F(\lambda). \]

Classically [2], this problem is solved by greedy coordinate descent. At each iteration, one selects the classifier with the largest error and updates its weight in order to decrease at most this error.

We propose here a randomised parallel coordinate descent method to solve this optimisation problem.

Separable Overapproximation

Let \( \omega \) be the maximum number of element in a row of matrix \( A \), that is

\[ \omega = \max_{i \in [1, \ldots, n]} |\{ j \in [1, \ldots, m] : A_{ij} \neq 0 \}|. \]

For \( \tau \leq n \), let us denote

\[ p_i = \left( \frac{1}{\omega} \right)^{\frac{1}{\tau}} \frac{1}{|\{ j \in [1, \ldots, m] : A_{ij} \neq 0 \}|}, \]

\[ c_i = \max \left( 1, \frac{\tau - 1}{\omega} \right) \frac{1}{|\{ j \in [1, \ldots, m] : A_{ij} \neq 0 \}|}, \]

\[ \beta = \sum_{k=1}^{\omega} \min \left( 1, \frac{1}{\tau} \sum_{i=1}^{\omega} p_i c_i \right). \]

The AdaBoost algorithm

• Compute \( \beta \) and \( (L_i)_{i \leq n} \) for \( \tau \geq 0 \).

• Randomly generate \( S_i \) following sampling \( \tilde{S} \) for \( i \in [1, \ldots, n] \).

• end for

• if \( F(\lambda^{t+1}) > F(\lambda^t) \) then

• end if

Theorem 1.

The function \( F \) has a coordinate-wise Lipschitz gradient with constants

\[ L_i = \max_{1 \leq j \leq m} A_{ij}^2, \quad 1 \leq i \leq n. \]

Moreover, if \( \tilde{S} \) is a \( \tau \)-nice sampling, then

\[ \mathbb{E}[F(\lambda + \delta)] \leq F(\lambda) + \frac{1}{\tau} \left( \frac{\|\nabla F(\lambda)\|}{\delta} \right)^2 + \beta \|\delta\|_2^2. \]

Parallel Adaboost algorithm

• Firstly define \( \tilde{S} \) of the problem. The convergence speed is in \( O(1/\varepsilon) \) through the EPSRC grant EP/I017127/1 (Mathematics for Vast Digital Resources).

Convergence

Numerical results

My work was supported by the British government through the EPSRC grant EP/I017127/1 (Mathematics for Vast Digital Resources).

References


