INTRODUCTION TO MACHINE LEARNING COMPSCI 4ML3

Lecture 6

Hassan Ashtiani

$\phi : \mathbb{R}^{d_1} \to \mathbb{R}^{d_2}$

COMPUTATIONAL COMPLEXITY OF NAÏVE RLS

Φ $n \times d_2$

$\Phi^T \Phi : d_2 \times d_2$

- Training: Calculate $w^{\mathrm{RLS}} = (\phi^T \phi + \lambda I)^{-1} \phi^T Y$
- Bottleneck: Matrix inversion
  - How many operations?

    cubic in d_2

- Prediction: $\hat{y} =< \phi(x), w^{RLS} >$ $\quad \sum_i \phi(x)_i \, w_i^{RLS} \rightarrow \boxed{d_2 \text{ operations}}$
  - How many operations?

- Regularization allows us to go into high-dimensional space without overfitting, but it does not solve the computational problem

COMPUTATIONAL COMPLEXITY

- Matrix multiplication (n-by-n matrices)
  - Native method: $O(N^3)$
  - Strassen's Algorithm: $O(N^{2.8074})$
  - Coppersmith–Winograd-like algorithms [current best $O(N^{2.3728639})$]
- Matrix inversion
  - Gaussian Elimination: $O(N^3)$
  - Possible to reduce it to multiplication

THE COMPUTATIONAL PROBLEM

- CAN WE SOLVE THE REGULARIZED LEAST SQUARES IN R^{d_2} **WITHOUT EXPLICITLY MAPPING THE DATA** INTO R^{d_2}?
  - $W^* = \min_{W \in R^{d_2}} \|\Phi W - Y\|_2^2 + \lambda \|W\|_2^2$
- SOMETHING LIKE MULTIPLICATION USING FFT
- IF SO, WE COULD EVEN MAP THE DATA TO AN INFINITE DIMENSIONAL SPACE!!

FFT AND MULTIPLICATION

2 3 5 6 7 8 9 1 1 2 3 - - - - 2 2 n digits $O(n^2)$ operations

4 5 7 - - - - - - - - . 1 1 n digits

1 0 1 0 0 1 0 1 1 1 0 1 1 1 0 1 1 1

x 1 0 1 1 1 0 1 1 1 0 0 1 0 0 1 0 0 0 0

FFT

nlogn time

1 0 0 0 1 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0

1 1 1 1 1 0 1 1 1 1 1 0 1 0 0 0 1 1 1

FFT

1 1 0 0 0 0 0 0 0 0 0

nlogn time

$x \times y = f^{-1}\left(f(x) \odot f(y) \right)$

$f(x \times y) = f(x) \odot f(y)$

THE COMPUTATIONAL PROBLEM

- CAN WE SOLVE THE REGULARIZED LEAST SQUARES IN R^{d_2} **WITHOUT EXPLICITLY MAPPING THE DATA** INTO R^{d_2}?
  - $\min_{W} \|\Phi W - Y\|_2^2 + \lambda \|W\|_2^2$
- SOMETHING LIKE MULTIPLICATION USING FFT
- IF SO, WE COULD EVEN MAP THE DATA TO AN INFINITE DIMENSIONAL SPACE!!

$d_2 = \infty$

THE KERNEL TRICK

$x^j, x^i \in \mathbb{R}^{d_1}$

$\phi(x^i) \in \mathbb{R}^{d_2}$

- Compute the high-dimensional inner product efficiently
  - $K(x^i, x^j) = < \phi(x^i), \phi(x^j) > = \sum_{k=1}^{d_2} \ldots$
- Use this as a building-block for performing other operations
- Rewrite the least squares solution so that it only uses the inner product of the feature maps?!
- $(\Phi^T\Phi + \lambda I)^{-1}\Phi^T Y$

THE KERNEL FUNCTION

- Kernel function for a mapping ϕ:

$\phi: R^{d_1} \rightarrow R^{d_2}$

- Example:

$d_2 = (d_1)^2$

  - $\phi(x)^T = [1, \sqrt{2}x_1, \sqrt{2}x_2, \dots \sqrt{2}x_d,$
    $(x_1)^2, x_1x_2\ \ x_1x_3, \dots, x_1x_d,\ \ x_2x_1, \dots \dots, x_dx_d]$
  - So polynomial basis functions with $M = 2$
  - Computing $K(u, v) =< \phi(u), \phi(v) >$
    - Complexity of naïve calculation? $d_2 = (d_1)^2$ operations
    - Better approach?

$d = d_1$

$$K(u,v) = \langle \phi(u), \phi(v) \rangle = \langle (1, \sqrt{2}u_1, \dots, \sqrt{2}u_d, u_1^2, u_1u_2, \dots, u_d^2), (1, \sqrt{2}v_1, \dots, \sqrt{2}v_d, \dots) \rangle$$

$$= 1 + 2\sum_{i=1}^{d} u_i v_i + \sum_{i=1}^{d}\sum_{j=1}^{d} u_i u_j v_i v_j \qquad \boxed{O(d^2) \text{ terms}}$$

$$= \left(1 + \sum u_i v_i\right)^2 = \left(1 + \langle u, v \rangle\right)^2$$

$O(d)$ operations is enoug

- $k(x^i, x^j) = \left(1 + (x^i)^T x^j\right)^2 = (1 + < x^i, x^j >)^2$
- Number of operations?

DEGREE M POLYNOMIALS

- For higher degree polynomials, we can use

$$k(x^i, x^j) = \left(1 + (x^i)^T x^j\right)^M$$

  - How many operations?

Naive: $\binom{M+d}{d} = \binom{M+d}{M}$ terms

improved:

$\alpha = 1 + \langle x^i, x^j \rangle$ $\boxed{d \text{ operations}}$

α^M → M mulliplication

→ $\log M$ (how?)

$\boxed{d + \log M}$

α
α^2
α^4
α^8
α^{16}
α^{32}

The Kernel Trick

- Compute the high-dimensional inner product efficiently
  - $K(x^i, x^j) =< \phi(x^i), \phi(x^j) >$
- Use this as a building-block for performing other operations
- Rewrite the least squares solution so that it only uses the inner product of the feature maps?!
- $(\Phi^T\Phi + \lambda I)^{-1}\Phi^T Y$

ROADMAP

- ASSUME d_2 IS VERY LARGE, EVEN $d_2 >> n$
- INSTEAD OF FINDING W, TRY TO INTRODUCE NEW PARAMETER a WHOSE SIZE IS n RATHER THAN d_2
- NOW WE HAVE n PARAMETERS
- FIND OPTIMAL a AS A FUNCTION OF K