# PARALLEL COMPUTING

Aditya Rajyaguru, Brandon Daryl Wanji
*6582282, 6151351*
Brock University
St. Catharines, ON, Canada

MACHINE LEARNING, K-NEAREST NEIGHBORS

***Abstract*—This document analyzes the performance of the K-nearest neighbors algorithm's implementation in C++, pinpoints its bottlenecks and discusses ways to improve its performance.**

## I. INTRODUCTION

The K-nearest neighbors algorithm is one of the simplest supervised machine learning algorithm used in regression and classification.
This document demonstrates how different such an algorithm is in terms of execution time by implementing it sequentially and paralleled.

## II. IMPLEMENTATION

### A. Sequential Implementation

A sequential implementation of the K-nearest neighbors implies there is a need to limit as much as possible the use of any specialized libraries that might help with parallel execution optimizations. [1]
This algorithm is as follows : like
K-nearest neighbors

1: Load the training and testing data-sets
2: Choose the value of K
3: **for all** points int test data **do**
4: - Find Euclidean distance to all training data points
5: - Store the Euclidean distances in a list and sort it
6: - Choose the first K points from Euclidean list
7: - Assign a class to the test point based on the majority of classes present in the chosen points =0

Please find attached the algorithm fully implemented in **C++**.

### B. Assumptions

The following assumptions were made ;

- The accuracy of the algorithm is not prioritized.
- The different classes for each data-sets are balanced. That is, they have equal number of points per class.
- Finding the best value for **K** is not prioritized.

## III. DESIGN

The Design phase for this project was a crucial phase as it had to be simple, easy to understand and modular. This phase brought into perspective the different needs for this project.
The different needs were devised into 5 sections, which are ;

### A. Data-sets

In order to test algorithm in different scenarios, 03 different data-sets were used. **The Prostate Cancer data-set** [2], **Abalone data-set** [3] and **Breast Cancer data-set** [4].
The data-sets are used to demonstrate how long the algorithm could take with the number of dimensions increasing.

### B. Point Class

The point class abstracts the dimensions of an index and it classification. This class is responsible for calculating the Euclidean distance between the coordinates within the point and another **Point q**'s coordinates, printing the coordinates and having the getters and setters for both the coordinates and classification.
The sorting algorithm used here is **Quick sort**.
Of all other sorting algorithms, quick sort was used because it has potential for parallelism and will help demonstrate a difference in execution times.

### C. Euclidean Class

The Euclidean class abstracts the distance from a point to be classified, Point P to another Point q with a pointer, pointing to the Point q. Having this class helps to easily identify the point from which the distance was calculated.

### D. KNN class

This class handles the parsing of all 3 data-sets, the K-nearest neighbors algorithm's main execution flow. It handles the sorting of the Euclidean distances after their calculations, switch the data-set depending on what a user's input.
In addition to that, this class devides the data-set into training and testing data-sets. Furthermore, this class is responsible for the calculation execution time's calculation over some variables.

### E. RUN

A make file was created to ease the execution of the program.
The program's execution is as follows :

- On the terminal, run the make file with the command **make**
- User will be prompted to enter a number from 0 to 03 inclusive to choose the data set to use.

## IV. ANALYSIS ON PERFORMANCE

With the implementation done, an analysis over the 03 data sets is done. The table below shows the average execution time of the algorithm over 05 runs for each data-set. The data-sets are in increasing order of points from left to right.

| Prostate Cancer | Breast Cancer | Abalone |
|---|---|---|
| 0.6 | 25 | 669 |

TABLE I
AVERAGE EXECUTION TIMES OVER 5 RUNS FOR EACH DATA-SET

As seen above, we can see a considerate increase in time as we go from the prostate cancer data-set to the Abalone data-set. This indicates that of all these data-set, the data-set with the highest parallelization potential is the Abalone data-set as it contains way more points than the 02 prior.

## V. BOTTLENECKS

## VI. CONCLUSION

## REFERENCES


[1] https://discuss.analyticsvidhya.com/t/practice-dataset-for-knn-algorithm/3104 Manish. *Prostate Cancer Data-set*. 2018.
[2] https://discuss.analyticsvidhya.com/t/practice-dataset-for-knn-algorithm/3104 Manish. *Prostate Cancer Data-set*. 2018.
[3] https://discuss.analyticsvidhya.com/t/practice-dataset-for-knn-algorithm/3104 Manish. *Prostate Cancer Data-set*. 2018.
[4] https://discuss.analyticsvidhya.com/t/practice-dataset-for-knn-algorithm/3104 Manish. *Prostate Cancer Data-set*. 2018.