LITERATURE REVIEW: Performance Analysis of K-Nearest Neighbor Algorithms

Heli Alpeshkumar Patel
School of Computer Science
Carleton University
Ottawa, Canada K1S 5B6
helialpeshkumarpatel@cmail.carleton.ca

October 6, 2022

1 Introduction

The amount of various datasets has significantly increased in recent years. There is a great need for innovative techniques that can transform these enormous quantities of data into useful information automatically. The intriguing, important, and comprehensible patterns that data mining reveals in huge datasets. Important application domains include business intelligence, customer relationship management, e-commerce, the World Wide Web, scientific simulation, and many more.

Higher intelligence is needed to deal with the present expansion of Big Data because standard computer performance cannot analyze this enormous volume of data. High-performance computing (HPC) must be taken into account while designing and developing systems since it can speed up calculation and produce precise results at a lower cost. There are several different types of high-performance computing, including computer clusters, grid computing, cloud computing, graphic processing units (GPU), MIC, and FPGA. One technique for achieving great performance on the GPU level is distributed memory programming with several processors. Parallelizing machine learning algorithms is one technique to use high performance computing to manage Big Data[6].

Classification is one of the core techniques in data mining, involving the training of a model on a dataset containing class labels and using the output to infer the class of unlabeled objects. The K-Nearest Neighbors (KNN) technique is one of the most popular machine learning algorithms since it makes categorization simple and effective[12][23]. KNN is a technique that classifies an item by using the training set's nearest point[12]. Since the KNN method is easy to use, quick to implement, and has a low error rate, it can effectively manage noisy data sets and produce accurate results. These are the reasons why it is utilized in so many diverse areas[6][15].

High-dimensional data sets can be used with the K-Nearest Neighbor technique since its computation is straightforward. However, when the test set, train set, and data dimension are higher than anticipated, the computational complexity will be quite high and the operating time will be extremely long [11]. To address the problem parallel processing can be utilized and the time complexity can be significantly reduced with the use of parallel processing.

A standardized method of communicating across many computers executing concurrent programs via distributed memory is the message passing interface (MPI)[5]. When a message is sent to an object, parallel process, subroutine, function, or thread, it is usually referred to as ”passing a message,” and that message is then utilised to start a different process.

The goal of this study is to analyze the performance of Serialized KNN and Parallelized KNN. I am planning to use Message Passing Interface(MPI) to parallelize KNN algorithm. I will use openly available dataset for both implementations to discover the performance differences in terms of speedup and efficiency. Based on the obtained performance results I would like to draw some conclusions and to see which one is better.

2 Literature Review

2.1 Overview of K Nearest Neighbour

Based on supervised learning, one of the most basic machine learning techniques, is K-Nearest Neighbors. K-NN is a non-parametric method that makes no assumptions about the underlying data. It is also known as a lazy learner algorithm since it saves the training dataset rather than learning from it immediately. Instead, when classifying data, it performs an action using the dataset. The KNN algorithm merely stores the data during the training phase, and when it receives new data, it categorises it into a category that is quite similar to the new data [4].

The K Nearest Neighbor method, which is used for regression and classification, is categorized as supervised learning algorithm[3]. It is most frequently used as a classification technique since it relies on the idea that related points can be discovered close to one another [9][2].

KNN has been used to tackle a variety of problems, including the following, in many different contexts. The KNN algorithm, which is a part of machine learning, was used to fill in the gaps in N Saranya’s proposed solution for diagnosing chronic kidney disease in the year 2021 [19]. Sumandeep et al. (2018) [13] used a KNN classifier to work on sentiment analysis. Sentiment analysis is a method created to examine the positive, negative, and neutral elements of any text unit. In recent years, numerous algorithms have been created for the sentiment analysis of twitter data. In their research work, they describe a novel method for the sentiment analysis of tweet data that is based on a prior study about sentiment analysis. The suggested strategy combines techniques for feature extraction and categorization. The N-gram algorithm is used to extract features, and a KNN classifier is used to divide the input data into positive, negative, and neutral categories. Face recognition has long been a hot topic, particularly now that Covid-19 is so prevalent and less physical touch is required in settings where personnel identification is crucial. In 2021, researchers [10] assessed the effectiveness of K-Nearest Neighbors (KNN) for facial recognition in several scenarios. The experimental findings show that K-Nearest Neighbors (KNN) performed better than other methods. It is important to note that the accuracy of the KNN classifier for face recognition is 100% for frontal faces that are exposed and 74.7% for those that are covered.

2.1.1 Classification

A class label is chosen for classification issues based on a majority vote, meaning that the label that is most commonly expressed around a particular data point is adopted[2]. Let's think about the case displayed in Figure 1. There are two classes, A and B, and nearby data points will be used to categorize orange coloured point to identify whether that orange point belongs to A or B by calculating their distance from the k closest data points.

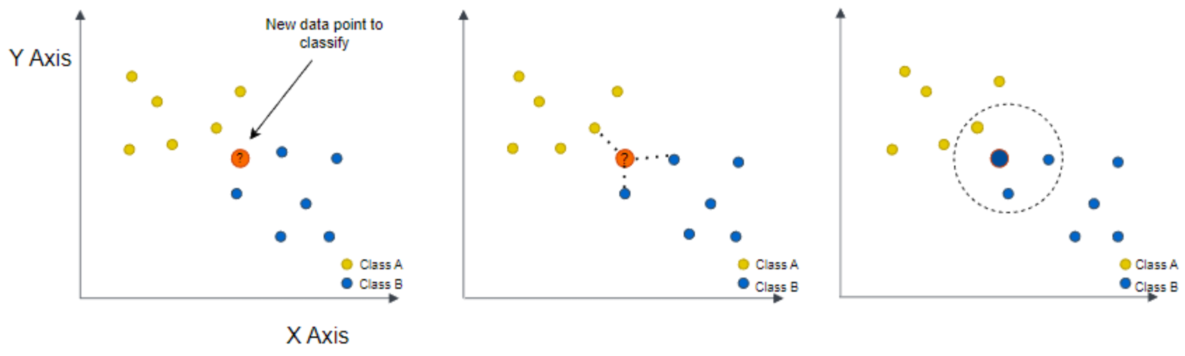

Figure 1: KNN diagram

2.1.2 Regression

Similar to classification issues, regression problems utilize the same principle; however, in this instance, a classification prediction is made by averaging the classifications of the first k nearest neighbors[2].

2.2 Serialized implementation of KNN

Based on the following method, the operation of the K-NN can be explained[8][1]:

Step 1: It is to choose the neighbours' K-number.

Step 2: Determine the K-number of neighbours' Euclidean distances from each other.

Step 3: Using the Euclidean distance estimate, select the K closest neighbours.

step 4: Count the amount of data points in each category among these k neighbours.

Step 5: Place the new data points in the category where the number of neighbours is highest.

Step 6: Our model is complete.

2.3 Laziness of K-Nearest Neighbour and previous research proposing solutions

KNN classifier is a model-free lazily learning algorithm. In contrast to current model-based classification algorithms, which build a model with a given training dataset and predict any test samples using the generated model, KNN classifier requires to store all the training instances in memory in order to locate all K nearest neighbours for a test sample. [20][24].

A straightforward yet effective machine learning classification method is the K-Nearest Neighbors (KNN) algorithm. However, it has some flaws, including a large memory requirement, a slow processing speed, class overlap, and a challenging K value setting process.To address the aforementioned problems in a single framework, xin zhang et al.[24] proposed an Improved K-Nearest Neighbor rule integrating Prototype Selection and Local Feature Weighting (IKNN PSLFW). Moreover, there are other studies are done as well for the improvement in efficiency of KNN such as, authors proposed a novel kNN algorithm with data-driven k parameter computation called S-KNN which is better in comparision of state-of-the-art KNN [21]. Then,other study presented a kTree technique that incorporates a training stage into the kNN classification to learn several optimal k values for various test/new samples [22].

According to me, it is obvious that while each of these methods has enhanced KNN functionality, they have not sufficiently solved the problem. There are research that show parallel processing significantly improved KNN effectiveness in various domains such as, using the popular K-Nearest Neighbor technique on multi-core CPUs, P.P. Halkarnikar et al. [11] presented a case study for categorising a big database, such as the electoral data for the Kolhapur constituency, into age-based categories. The three age groups for election candidates were YOUNG, MIDDLE, and OLD. The processing time produced a significant enhancement. In a later study in 2019 [18], the parallel gwk-NN classifier that the researchers had developed throughout the model training and image classification stages was further developed using calls to MPI and GDAL in the C++ development environment on an eight-core CPU. Early experiments by the authors show that the proposed parallel gwk-NN classifier can improve the performance of high-resolution remotely sensed images with different types of land cover.

2.4 Message Passing Interface(MPI)

In parallel computing, nodes are groups of computers or even individual processing cores on a single computer. A fraction of the entire computing issue is generally worked on by each node in a parallel configuration. To synchronise each parallel node's activities, communicate data across nodes, and exert command and control over the entire parallel cluster is the next difficult task. For these responsibilities, the message passing interface specifies a standard set of functions [5]. Benefits of the message passing interface include standardization, portability, speed, and functionality.

2.5 Related work

KNN algorithm is worth researching to improve since it is extremely practical, helpful, and widely utilised in the business. Various studies have been done in an effort to enhance the KNN algorithm and expand the problems that it may be used to solve with big data.

CUKNN, a parallel KNN implementation built on the CUDA multi-thread paradigm. To increase GPU efficiency, a variety of CUDA optimization approaches were used. CUKNN performs far better and can accelerate computations by up to 15.2X. When the training dataset's dimension and record count are changed, it also demonstrated strong scalability[14].

The primary contribution of Jan Masek et al. in 2015 study is a technique that uses OpenCL to speed the k-Nearest Neighbor machine learning algorithm. The technique may run concurrently on many GPUs. The updated version of the technique developed by the authors produces excellent results for k neighbours that are less than 10. In comparison to utilising a single core CPU (Intel Core i7-3770, 4.1GHz), they discovered that using very inexpensive hardware (2x NVIDIA GeForce GTX 690) made it feasible to calculate 4 million items (each with 10 characteristics) in 3 minutes as opposed to almost 31 hours. Up to 750x of acceleration was obtained[16].

M. M. A. Patwary et al. introduced PANDA, a distributed kd-tree-based KNN implementation that parallelizes the building of kd-trees and querying on enormous datasets of up to 189B particles drawn from several scientific fields. PANDA performs more than an order of magnitude quicker on a single node than the most recent KNN implementations. Additionally, they demonstrated that PANDA scales effectively for all phases of computation up to 50,000 cores in both strong and weak scaling senses. As a consequence, PANDA can create kd-trees for 189 billion particles in about 48 seconds and respond to 19 billion KNN queries in around 12 seconds utilizing 50,000 cores[17].

The parallel KNN method was used for 3D reconstruction matching. For the research's higher quality photos, a GPU device running the Nvidia CUDA SDK was utilized. In order to reduce access time, the distance calculating result is kept in shared memory and used in the pipeline for real-time feature tracking. The outcome demonstrated that parallel KNN is 10 times quicker than sequential KNN with superior effectiveness and efficiency[7].

As can be observed, numerous studies have been conducted to enhance KNN for diverse domains.

References

[1] K-nearest neighbor. https://medium.com/swlh/k-nearest-neighbor-ca2593d7a3c4. (Accessed: October 2022).

[2] Simple understanding and implementation of knn algorithm! https://www.ibm.com/topics/knn. (Accessed: October 2022).

[3] Simple understanding and implementation of knn algorithm! https://www.analyticsvidhya.com/blog/2021/04/simple-understanding-and-implementation-of-knn-algorithm/, 2021. (Accessed: October 2022).

[4] K-nearest neighbor(knn) algorithm for machine learning. https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning, 2022. (Accessed: October 2022).

[5] message passing interface. https://www.techtarget.com/searchenterprisedesktop/definition/message-passing-interface-MPI, 2022. (Accessed: October 2022).

[6] Maha A. Alanezi and Abdulla AlQaddoumi. Applying parallel processing to improve the computation speed of k-nearest neighbor algorithm. In *2020 International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy (ICDABI)*, pages 1–6, 2020.

[7] Ming-Wei Cao, Lin Li, Wen-Jun Xie, Wei Jia, Zhi-Han Lv, Li-Ping Zheng, and Xiao-Ping Liu. Parallel k nearest neighbor matching for 3d reconstruction. *IEEE Access*, 7:55248–55260, 2019.

[8] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

[9] Pádraig Cunningham and Sarah Jane Delany. K-nearest neighbour classifiers - a tutorial. *ACM Comput. Surv.*, 54(6), jul 2021.

[10] Xinyu Guo. A knn classifier for face recognition. In *2021 International Conference on Communications, Information System and Computer Engineering (CISCE)*, pages 292–297, 2021.

[11] P. P. Halkarnikar, Ananda P. Chougale, H. P. Khandagale, and P. P. Kulkarni. Parallel k-nearest neighbor implementation on multicore processors. In *2012 International Conference on Radar, Communication and Computing (ICRCC)*, pages 221–223, 2012.

[12] S.B. Imandoust and Mohammad Bolandraftar. Application of k-nearest neighbor (knn) approach for predicting economic events theoretical background. *Int J Eng Res Appl*, 3:605–610, 01 2013.

[13] Sumandeep Kaur, Geeta Sikka, and Lalit Kumar Awasthi. Sentiment analysis approach based on n-gram and knn classifier. In *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)*, pages 1–4, 2018.

[14] Shenshen Liang, Cheng Wang, Ying Liu, and Liheng Jian. Cuknn: A parallel implementation of k-nearest neighbor on cuda-enabled gpu. In *2009 IEEE Youth Conference on Information, Computing and Telecommunication*, pages 415–418, 2009.

[15] Chun Jie Ma and Zheng Sheng Ding. Improvement of k-nearest neighbor algorithm based on double filtering. In *2020 5th International Conference on Mechanical, Control and Computer Engineering (ICMCCE)*, pages 1567–1570, 2020.

[16] Jan Masek, Radim Burget, Jan Karasek, Vaclav Uher, and Malay Kishore Dutta. Multi-gpu implementation of k-nearest neighbor algorithm. In *2015 38th International Conference on Telecommunications and Signal Processing (TSP)*, pages 764–767, 2015.

[17] Md. Mostofa Ali Patwary, Nadathur Rajagopalan Satish, Narayanan Sundaram, Jialin Liu, Peter Sadowski, Evan Racah, Suren Byna, Craig Tull, Wahid Bhimji, Prabhat, and Pradeep Dubey. Panda: Extreme scale parallel k-nearest neighbor on distributed architectures. In *2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 494–503, 2016.

[18] Yingxia Pu, Xinyi Zhao, Guangqing Chi, Shuhe Zhao, Jiechen Wang, Zhibin Jin, and Junjun Yin. Design and implementation of a parallel geographically weighted k-nearest neighbor classifier. *Computers Geosciences*, 127:111–122, 2019.

[19] N Saranya, M Sakthi Samyuktha, Sharon Isaac, and B Subhanki. Diagnosing chronic kidney disease using knn algorithm. In *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)*, volume 1, pages 2038–2041, 2021.

[20] Shichao Zhang. Cost-sensitive knn classification. *Neurocomputing*, 391:234–242, 2020.

[21] Shichao Zhang, Debo Cheng, Zhenyun Deng, Ming Zong, and Xuelian Deng. A novel knn algorithm with data-driven k parameter computation. *Pattern Recognition Letters*, 109:44–54, 2018. Special Issue on Pattern Discovery from Multi-Source Data (PDMSD).

[22] Shichao Zhang, Xuelong Li, Ming Zong, Xiaofeng Zhu, and Ruili Wang. Efficient knn classification with different numbers of nearest neighbors. *IEEE Transactions on Neural Networks and Learning Systems*, 29(5):1774–1785, 2018.

[23] Xin Zhang, Hongshan Xiao, Ruize Gao, Hongwu Zhang, and Yu Wang. K-nearest neighbors rule combining prototype selection and local feature weighting for classification. *Know.-Based Syst.*, 243(C), may 2022.

[24] Xin Zhang, Hongshan Xiao, Ruize Gao, Hongwu Zhang, and Yu Wang. K-nearest neighbors rule combining prototype selection and local feature weighting for classification. *Knowledge-Based Systems*, 243:108451, 2022.