International Journal of Computer Sciences and Engineering
Vol.**11**, Issue **6**, pp.**30-36**, June **2023**
ISSN: 2347-2693 (Online)
Available online at: www.ijcseonline.org

ISSN: 2347-2693 (E)

Research Paper

Hybrid Classification Algorithm for Improved Big Data Processing

Azibator Banigo[1*], Nuka Dumle Nwiabu[2], Vincent Ike Anireh[3]

[1,2,3]Department of Computer Science, Rivers State University, Port Harcourt, Nigeria

**Corresponding Author: azibator.banigo@ust.edu.ng*

Received: 20/Apr/2023; **Accepted**: 22/May/2023; **Published**: 30/Jun/2023. **DOI**: https://doi.org/10.26438/ijcse/v11i6.3036

Abstract: This paper puts forward a novel approach in big data processing and it is aimed at cutting computational time and enhancing classification accuracy. The research focuses on the relevance and significance of hybrid algorithms, specifically combining the Ball Tree and Weighted k Nearest Neighbors (k-NN) algorithms. The objective of this study is to address the limitations of traditional algorithms by reducing computational time while improving classification accuracy. The methodology employed in this research is the constructive research method, which allows for the development and evaluation of new algorithms. This methodology was chosen as it facilitates the creation of innovative approaches to tackle the challenges of big data processing. Experimental results demonstrate that the proposed hybrid algorithm yields promising outcomes. When classifying the MNIST dataset, the algorithm achieved an accuracy rate of 97%, misclassifying only 256 out of 10,000 images. The harmonic mean between precision and recall was found to be 0.999716, indicating a high level of performance. Notably, the computational time required for classification was significantly shorter compared to traditional classification techniques. Overall, the hybrid algorithm combining the Ball Tree and Weighted k-NN proved to be an effective solution for big data processing. By reducing computational time and enhancing accuracy, it presents a valuable contribution to the field. This research opens avenues for further exploration and application of hybrid algorithms in various domains where efficient and accurate big data processing is crucial.

Keywords: Big Data, Ball Tree Algorithm, Classification, Weighted K-Nearest Neighbours (WKNN), Hybrid Algorithm, K-Nearest Neighbours, MNIST Dataset.

1. Introduction

The production of data on a daily basis is rapidly increasing, and it is no longer just in batches but also in continuous streams. The need to extract valuable insights from large datasets has increased significantly in various domains like healthcare, business analytics, bioinformatics, and more. Data warehousing plays a crucial role in enabling this capability by integrating data from different transactional, legacy, and visible systems and applications. By incorporating data from diverse sources, data warehousing facilitates the extraction of meaningful information and supports analysis in these domains [1]. Big Data encompasses data collections that can be analyzed for insights, regardless of their structure, whether structured, semi-structured, or unstructured. Big data focuses on aggregating the data rather than considering individual records [2]. Big data processing refers to a set of techniques and programming models that enable access to vast amounts of data, with the goal of extracting useful information that can support and inform decision-making [3]. This process is characterized by a repetitive cycle of discovery, using either automated or manual methods, and involves collaboration between humans and computers. Classification involves organizing data into distinct categories, tags, or labels to maximize its efficiency and effectiveness. To gain insights from big data, it is necessary to properly define the tags and labels with clear and specific terms. Classification is a supervised learning approach and as such analyses a given dataset to construct a model that can partition the data into the desired classes. Various classification methods, such as Logistic Regression, Support Vector Machines (SVM), Naive Bayes, random forest, and k-nearest neighbours, among others, prove beneficial for analysing large volumes of data swiftly. Hybrid algorithms are utilized to merge multiple distinct algorithms. This approach is rooted in the recognition that the individual strengths of different algorithms can collectively contribute to improved performance when handling and analysing extensive datasets. By combining these algorithms, the aim is to overcome the limitations associated with individual algorithms and leverage their unique strengths to achieve superior outcomes. Hybrid algorithms demonstrate superior capability in effectively handling complex and large-scale datasets compared to traditional algorithms. The significance of hybrid algorithms lies in their ability to integrate various techniques such as clustering, classification, and regression analysis, enabling them to extract meaningful insights from diverse types of datasets and ensuring a synergy of practiced based solution and theoretical developments are drawn [4]. The amount of computational time needed to process large data grows together with its size and volume. Consequently, specialized hardware, software, and algorithms are essential to manage the computational load associated with big data processing. Traditional algorithms employed for big data processing are often inadequate, unmodified, or not optimized to effectively handle the immense volume and intricacy of the data sets. As a result, the development of specialized algorithms specifically designed to enhance big data processing becomes crucial. These algorithms are tailored to address the unique challenges posed by big data, ensuring more efficient and effective processing. Computational time is an important issue in big data processing and a good approach is to utilize specialized algorithms which can learn from the data and adapt favourably to changing conditions.

Another common issue of big data processing is often times the accuracy of prediction. Traditional classification algorithm may not be fully optimized for big data processing and may encounter lots of difficulty in processing the sheer volume and complexity of big data. This can lead to inaccurate predictions and consequently reduce the effectiveness of the classification model. To address this issue, researchers have been developing various techniques for big data classification, these techniques can improve the accuracy of the predictions, they include deep learning, active learning among others. Also traditional algorithms can be modified in a way and manner that improves the accuracy of predictions. A powerful approach for quick and accurate closest neighbour classification is the ball tree algorithm. It is effective for a variety of applications and data types since it can handle both continuous and categorical data as well as manage or include different distance matrices. The ability of the ball tree algorithm to efficiently divide the search space makes it particularly useful in machine learning. For particular data distributions and smaller dimensions, this partitioning enables effective neighbourhood requests with an O(log N) time complexity[5].

K Nearest Neighbour (KNN) algorithm involves identifying the subset of points in a dataset that are the most closest or similar to a given query point. This closeness is determined by a distance function, denoted as d(p, q), which calculates the distance between each point in the dataset (P) and the query point (q) [6].

In [7], weighted K Nearest Neighbour rule outperformed the standard majority K Nearest Neighbour rule in terms of error rate. This finding emphasized the significance of incorporating weights into the nearest neighbour searches, leading to improved accuracy in classification and prediction tasks.

The rest of the paper is organized as follows, Section 2 contains the related works, Section 3 contains the research methodology and Hybrid Algorithm Section 4 contains the results and discussion, and finally section 5 concludes research with future direction.

2. Related Work

2.1 Computational Time:

In order to improve the computational time in big data Processing, a novel algorithm called Enhanced Fuzzy based Linkage Clustering Algorithm (EFCA) proposed in [8]. MATLAB was utilized as the programming platform to execute this algorithm. Throughout the entire process, it adopts an approach focused on clustering in high-dimensional spaces. The results demonstrated a notable reduction in computational time of approximately 16.4% compared to alternative methods, making it highly efficient for handling datasets with high dimensions.

In order to improve the computational time of big data processing, the k –means clustering algorithm was modified by [9]. The centroid was improved with a selection technique called the Radical Basis Function (RBF) kernel and a distance measuring function. The time consumption was about 500 milliseconds with data instances of over 3000 as compared to 2000 from the existing system.

In [10] a Hierarchical Spatial-Temporal State Machine (HSTSM) data modeling approach, which incorporates a number of soft computing techniques to handle the difficulties of analyzing various types of big data and offer a powerful big data analytics tool for numerous application domains. A hypothetical cyber-physical architecture that may profit from this approach is also provided. In light of the digital revolution and society's expanding interconnection, this paper addresses the relevance of big data analytics and computational intelligence. The need of using computational intelligence techniques to efficiently handle and comprehend the enormous amounts of data produced by connected devices is highlighted in this research. It serves as a summary of various approaches, which can help in handling and processing massive volumes of data effectively, thus reducing computing time. The study also examines certain application fields where big data is widely used, including social network sentiment analysis, intelligent transportation, and healthcare. Big data, cyber-physical systems, and computational intelligence are explored in the context of cutting-edge research and creative applications in these fields.

2.2 Accuracy:

By calculating the Bhattacharya coefficient and identifying the input space, the Bhattacharya distance is used to assess how similar the data are. J.R. Quinlan developed the decision tree in 1980, and in the study conducted by [11], a decision tree algorithm named C4.5 was utilized for processing big data. The size, time, and cost factors are only a few of the difficulties in big data processing discovered in the paper. The study focuses on the goal of improving decision tree model accuracy while using training data. The authors suggest integrating the Bhattacharya distance and the C4.5 method and find that it performs more accurately across a variety of test conditions than the original C4.5 approach. The capacity of the decision tree to assess the numerical weight of connections between nodes is credited with this development.

A Jen-Ton Framework was developed in [12] to effectively analyze the sentiment of social networking data. The framework integrates three distinct techniques: Imputation of Missing Sentiment (IMS), Aspect-based Sentiment Analysis using Fuzzy Logic (ASFuL), and Aspect-based Sentiment Analysis using Clustering with Genetic Algorithm (ASTA-CGA). The study focuses on improving aspect-based sentiment analysis's accuracy in the context of big data. In order to analyze big data effectively, it is essential to have a reliable architecture and methodologies that utilize both supervised and unsupervised algorithms. The goal is to get better sentiment analysis findings. The research seeks to overcome the issues related to sentiment analysis in large-scale data sets by integrating various methodologies.

2.3 Weighted K-Nearest Neighbours:

A study conducted by [13] explored the characteristics and regulations governing electricity in Australia's National Electricity Market. They employed the weighted K-Nearest Neighbor technique to investigate all the various aspects. The study proposes a novel approach for short-term load forecasting that utilizes the Euclidean distance as a weight, aiming to achieve enhanced accuracy in the predictions.

Breast cancer is a prevalent and aggressive type of cancer among women. Due to its severity and mortality rate, researchers have been actively working on Computer-Aided Detection (CAD) methods for classifying lethal forms of cancer. In a recent study by [14], a hybrid approach was proposed that combines Weighted K-Nearest Neighbors with metaheuristic techniques. This combined algorithm effectively explores the search space to identify optimal solutions, aiming to improve survival rates and enable early diagnosis. The study incorporates three common metaheuristic algorithms namely: Particle Swarm optimization (PSO), Dragon-Fly Optimization (DFOA), and Crow-Search Optimization algorithm (CSOA). The results demonstrate that the weighted K-Nearest Neighbor algorithm exhibits enhanced accuracy in predicting breast cancer.

According to the findings presented in [15] the weighted K-Nearest Neighbor (KNN) classification algorithm outperformed the standard K-Nearest Neighbor method, Random Forest Algorithm, and Support Vector Machine on 14 different datasets. This can be attributed to the weighted KNN's ability to improve accuracy, as well as its utilization of bootstrapping to create an ensemble of diverse models.

2.4 Ball Tree:

The field of forensic face sketch-photo recognition has gained significant interest within the law enforcement community. In a study conducted in [16], a novel technique for face sketch-photo identification was developed by combining the VGG deep feature extraction method with the ball-tree searching algorithm. The results demonstrated that this technique surpassed other existing methods in terms of recognition accuracy and performance, as evaluated using the CUFS and IIT-D datasets.

In [17], the transmission radius of an automated light trap particularly created for the management of the Brown plant hopper (BPH) was determined using the ball tree algorithm. The researchers sought to determine the ideal transmission radius for the automated light trap, which could successfully catch and regulate the BPH population, using the ball tree classification algorithm. The light trap was designed specifically to automatically detect BPHs and count the number of BPHs captured within the trap. To efficiently handle the computational tasks, a parallel approach using the CUDA NVIDIA platform was adopted. This parallel approach facilitated the creation of the Ball Tree data structure and enabled the determination of the communication radius for the autonomous light trap.

To tackle the computational challenges associated with searching for similar patches across multiple images in computer vision applications, a study was conducted. The researchers examined and assessed various nearest neighbor algorithms to enhance the efficiency of finding similar patches. Although picture patches are known to have independent distributions, Gaussian distributions are often used for evaluating nearest neighbor methods. The findings of the study indicated that the ball tree method performed exceptionally well, ranking second only to vantage point trees in terms of achieving the most favorable outcomes. These conclusions were drawn based on extensive experimentation conducted on datasets consisting of images. [18].

3. Experimental Method

This paper presents a hybrid algorithm that utilizes the Ball-Tree and Weighted KNN algorithm carried out with the MNIST dataset on Pycharm IDE and Jupyter Notebook. The constructive research method was utilized for this paper because of its practical relevance and its ability to bridge the gap between theoretical research and practical approach. The project highlights the path from theory to successful implementation in the real world by incorporating both theoretical developments and actual applications. [19].
The flowchart in figure one shows how the hybrid algorithm works.

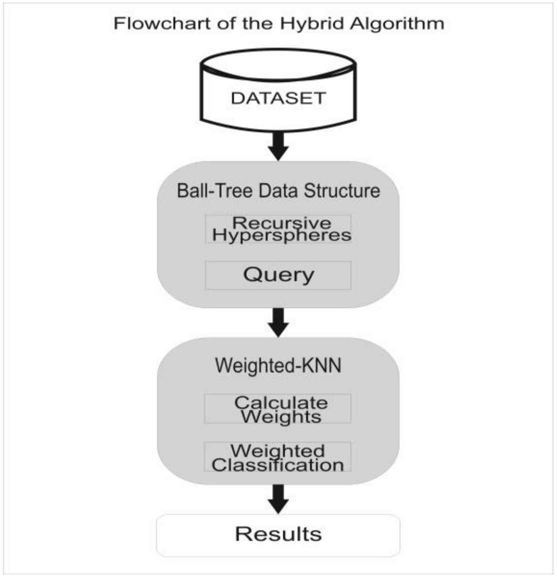

Figure 1: Flow Chart of the Hybrid Algorithm

Ball-tree
The ball tree is data structure used for efficient nearest neighbour search. It is employed because it allows for faster searching by narrowing down the search space resulting in faster computational time. A crucial component of the algorithm is the recursive hyper spheres, which divides the search space into spheres via recursion. A root node serves as the starting point for the hyper sphere, which then splits into smaller offspring hyper spheres until a predetermined stopping condition is satisfied. As a result, the search area is reduced.

By utilizing the Ball Tree algorithm's structure, the querying procedure aids in directing the classification search to the dataset's most pertinent features. This can help to reduce the computational time in classification as compared to other exhaustive techniques.

Weighted K-Nearest Neighbours
In Nearest Neighbours search, the K value is crucial, and once it is known, the algorithm is designed to give each point a weight based on its proximity it is to the query point. When evaluating the closest neighbors, the weights will also be calculated as the inverse of distance using the Euclidean distance as the distance metric. The hybrid algorithm can perform classification by incorporating the nearest neighbour weights in the classification process.

Hybrid Algorithm
```
class BallTreeNode:
    def __init__(self, points=None):
        self.points = points
        self.pivot = None
        self.radius = None
        self.child1 = None
        self.child2 = None
def construct_ball_tree(points, leafSize):
    # Base case: If only one point remains, create a leaf
node and return
    if len(points) == 1:
        return BallTreeNode(points[0])
    # Create the parent node
    parent = BallTreeNode(points)
    parent.pivot = centroid(parent.points)  # Calculate the
centroid of the points
    parent.radius = max(distance(parent.pivot, point) for point
in parent.points)  # Calculate the maximum distance to a
point
    # Create child nodes
    child1, child2 = BallTreeNode(), BallTreeNode()
    parent.child1, parent.child2 = child1, child2
    # Set the pivots for the child nodes
    child1.pivot = point_farthest_from_parent(parent)
    child2.pivot = point_second_farthest_from_parent(parent)
    # Recursive construction of child nodes if the number of
points exceeds the leaf size
    if len(child1.points) > leafSize:
        child1 = construct_ball_tree(child1.points, leafSize)
    if len(child2.points) > leafSize:
        child2 = construct_ball_tree(child2.points, leafSize)
    parent.child1, parent.child2 = child1, child2
    return parent

def weighted_knn_classification(query_point, ball_tree, K):
    # Find the nearest neighbors of the query point in the
ball tree
    nearest_neighbors =
ball_tree.find_nearest_neighbors(query_point, K)
    # Calculate weights for each neighbor based on their
distances to the query point and the distance to the 1st nearest
neighbor
    weights = [1] + [(distance(query_point, neighbor.point) -
distance(query_point, nearest_neighbors[0].point)) /
                (distance(query_point, nearest_neighbors[K -
1].point) - distance(query_point, nearest_neighbors[0].point))
                for neighbor in nearest_neighbors[1:]]
    # Collect the class labels of the nearest neighbors
    class_labels = [neighbor.point.label for neighbor in
nearest_neighbors]
    # Calculate the weighted votes for each class label
    weighted_votes = {label: sum(weight for weight, neighbor
in zip(weights, nearest_neighbors) if neighbor.point.label ==
label)
for label in set(class_labels)}
    # Determine the predicted class label based on the
highest weighted votes
    predicted_label = max(weighted_votes,
key=weighted_votes.get)
    return predicted_label
```

4. Results and Discussion

This paper proposed a Hybrid classification algorithm for big data processing. The MNIST dataset was given special consideration throughout the testing of the algorithms since it is a helpful starting point for creating and testing big data processing algorithm. Before scaling up to much bigger and more complicated datasets, it enables simple comprehension of the behavior of algorithms by providing a standardized and accessible dataset for early validation and benchmarking. Furthermore its well-defined labels and features for each image allows for comprehensive evaluation of classification algorithms. Precision, accuracy, recall and other performance metrics can be evaluated to ascertain the performance of the algorithm and compare it to other approaches. The runtime of the hybrid algorithm was measured against the traditional KNN algorithm and the accuracy was determined after analysis using a confusion matrix for classification on the MNIST dataset. All of this was carried out with an Intel Core I5 series on a 2.20 GHz processor and 4GB memory, and runs on a windows operating system the dataset was downloaded on the local machine and are available online at kaggle.com and machine learning repository.

Figure 2 shows the computational time of the Hybrid Algorithm on the MNIST dataset, Wine, Iris and Wisconsin Breast Cancer Dataset.

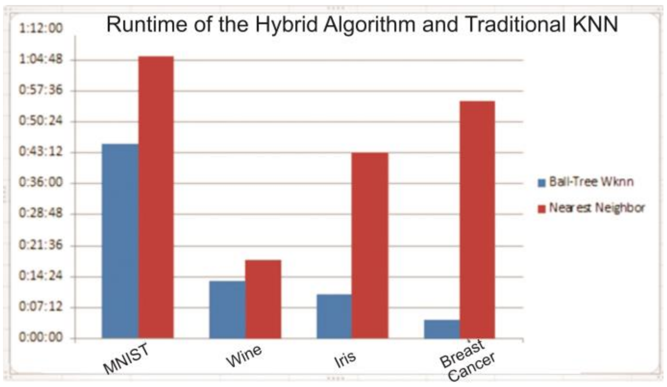

Figure 2: Time taken to Classify Various Datasets using the Nearest Neighbor and Ball-Tree WKNN algorithm.

An algorithm's computational time is the length of time it takes for a computer to execute it completely. It is one of the simplest ways, among many others, to measure an algorithm's effectiveness and performance over time. The hybrid algorithm was utilized on a variety of dataset to show its robustness and for proper comparative analysis, figure 2 shows that the hybrid algorithm outperforms the traditional KNN algorithm on various datasets.

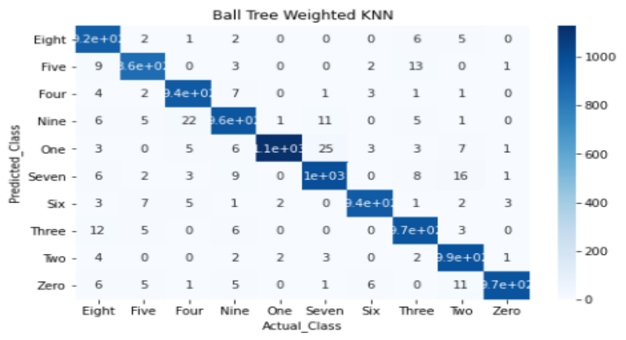

Figure 3: Confusion Matrix of the Hybrid Algorithm

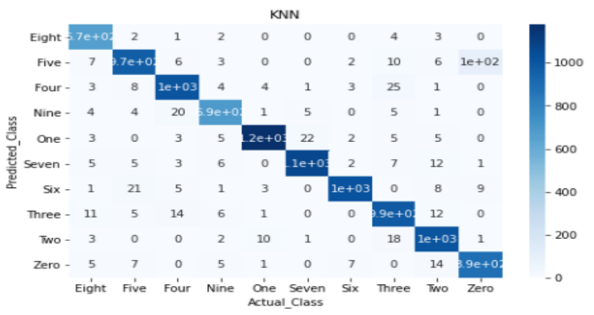

Figure 4: Confusion Matrix of Traditional KNN

A predictive analytic tool is a confusion matrix. It is a table that contrasts values from the real world with predictions from the model, and as such, it is a useful tool for generating metrics to evaluate how a machine learning classifier performed on a dataset. The confusion matrices for both the Hybrid Algorithm and the Traditional Algorithm are shown in Figures 3 and 4. Information on true positives, true negatives, false positives, and false negatives is provided by the confusion matrix. These numbers are crucial for assessing how well the algorithms work. The confusion matrix may also be used to construct performance metrics, such as accuracy, precision, recall, and F1 score, which adds to our understanding of how well the algorithms perform in classification tasks. The confusion matrix for a single class zero is derived as follows:

1. True Positive (TP) refers to data points that have been accurately classified as belonging to the positive class. In other words, TP stands for the number of samples that a classification model correctly classifies as positive occurrences when those samples really fall within the positive category. It shows how well the model can recognize instances of the positive class by representing the correctly categorized positives in the confusion matrix.
2. True Negative, (TN) refers to the number of samples that are accurately identified as negative instances by a classification model when they indeed do not belong to the class of interest. It represents the correctly classified negatives in the confusion matrix, providing information about the accuracy of the model in identifying non-class instances correctly.
3. False Positives (FP) this refers to Data points that have been mistakenly classified as the positive class. FP is the quantity of samples that a classification model incorrectly classifies as positive instances even when they do not truly belong to the positive class. It displays the positives that were incorrectly categorized in the confusion matrix, highlighting the model's propensity to identify cases that did not fit any category as positives.
4. False Negatives (FN) Data points that have been wrongly classified negative class are known as False Negatives. In other words, FN stands for the number of samples that a classification model incorrectly classifies as negative instances even if they actually belong to the positive class. It depicts the occurrences that the confusion matrix incorrectly categorized as negatives, demonstrating the model's inability to accurately detect positive situations.

Table 1: Confusion Matrix for Zero

	Classified Negative	Classified Positive
Positive	971(TP)	35(FP)
Negative	7(FN)	8978(TN)

The values from Table 1 can be used to calculate several performance matrices like the accuracy, recall, f1 score and precision.

Table 2: Performance Matrix for Class Zero

	Accuracy	Precision	Recall	F1 Score
Zero	0.9958	1.01766	0.945641	0.94546

The values in Table 2 were derived using the following:
1. Accuracy: The number of data points out of all the data points in the test set that was properly classified.
2. Precision: Precision is simply the number of samples projected to be positive evaluates the samples that were classified as positive.
3. Recall: This is simply the number of samples that are correctly identified as being in the positive class relative to the total number of samples that really fall into this category.
4. F1 Score: This is the harmonic mean between the precision and recall.

The rest of all the classes and their performance metric are laid out in table 3 and 4

Table 3: Matrices of the Hybrid Algorithm's Performance on the MNIST Dataset

	Accuracy	Precision	Recall	F1 Score
Eight	0.9931	1.01766	0.945641	0.98033
Five	0.9944	1.033493	0.96861	1
Four	0.9944	1.020541	0.962283	0.990556
Nine	0.9908	1.055921	0.959163	1.005219
One	0.9942	1.049257	0.995591	1.021719
Seven	0.9914	1.047219	0.960539	1.002008
Six	0.9962	1.026115	0.985371	1.00533
Three	0.9935	1.027572	0.96131	0.993337
Two	0.994	1.014403	0.955426	0.984032
Zero	0.9958	1.037393	0.992843	1.014629
Average	0.99378	1.032957	0.968678	0.999716

$$\text{Accuracy} = \frac{\mathbf{TP+TN}}{\mathbf{TP+FP+TN+FN}} \quad (1)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \quad (2)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \quad (3)$$

$$\text{F1 Score} = \frac{2*\text{Precision}*\text{Recall}}{\text{Precision+Recall}} \quad (4)$$

Table 4: Performance Matrices of Traditional KNN on the MNIST Dataset

	Accuracy	Precision	Recall	F1 Score
Eight	0.9946	1.018209	0.941093	0.978134
Five	0.9813	1.162064	0.949019	1.044792
Four	0.9899	1.050882	0.951127	0.998519
Nine	0.9926	1.061162	0.953296	1.004341
One	0.9935	1.039682	0.983319	1.010715
Seven	0.993	1.039651	0.973731	1.005612
Six	0.9936	1.049230	0.984600	1.015888
Three	0.9877	1.052294	0.930188	0.987481
Two	0.9903	1.035934	0.942110	0.986797
Zero	0.9849	1.045828	0.888223	0.960604
Average	0.99014	1.055494	0.949671	0.999288

Table 3 and Table 4 gives a detailed evaluation of the both the traditional and the Hybrid classification algorithm showing that both algorithms are solid in performance when the k value is set to 15.

From the confusion matrix in Figure 3 and 4 it we can generate how many instances were classified correctly and wrongly by adding all of the values in the diagonal and subtracting them the overall amount of the dataset set.

Table 5: Percentage Accuracy of Hybrid Algorithm

	Misclassified	Classified	Accuracy%
Hybrid	311	9689	3.2%
Traditional	493	9507	5.2%

Table 5 shows that the hybrid algorithm outperforms the traditional algorithm when judging how many instances it misclassified.

5. Conclusion and Future Scope

In this paper, the Weighted K-Nearest Neighbors (KNN) algorithm's accuracy was merged with the faster computing time made possible by the Ball Tree data structure. The authors showed that combining the Weighted KNN and Ball Tree algorithms is a successful strategy for attaining both quick and accurate classification. The researchers demonstrated enhanced classification performance while preserving computing economy by utilizing the benefits of the Ball Tree data structure, which improves closest neighbor searches, and adding the Weighted KNN method to evaluate the relevance of neighbors. The outcomes demonstrated this combination strategy's potential for effective and precise classification tasks. However, this hybrid approach requires significant memory resources because the entire dataset must be stored in memory, and the weighted component of the algorithm is sensitive to outliers. In the future, to address these limitations, the algorithms could be implemented on distributed systems. Additionally, using robust distance metrics that can handle the classification of individual data points and their distributions may be helpful in dealing outliers.

References

[1]. Kalio, Q.P, Nwiabu, N, “A Framework for Securing Data Warehouse Using Hybrid Approach,” *International Journal of Computer Science and Mathematical Theory,* Vol.**4**, No.**1**, pp.**3-4**, **2019**.

[2]. Prasad, Lakshmi Y.” Big Data Analytics Made Easy”. Vol 1, Punjab : Notion Press, **2016**

[3]. Farhad, M., Hamid, N., & Bahman, J., “Energy-Efficient Big Data Analytics”., *Advances in Computer.,* Vol.**100**, **2015**

[4]. Nwiabu, N., Adeyanju, I. “User Centred Design Approach to Situation Awareness,” International Journal of Computer Application, Vol.**49**. Issue.**27**, pp **26-30, 2012**

[5]. Kramer, Oliver. “Dimensionality Reduction with Unsupervised Nearest Neighbors”. “Intelligent Systems Reference Library” First Edition, Vol **51**, Berlin : Springer Berlin, Heidelberg, **2013**. ISBN **978-3-642-38652-7**..

[6]. Kumar , N., Zhang, L., & Nayar, S. “What Is a Good Nearest Neighbors Algorithm for Finding Similar Patches in Images?” [ed.] David Forsyth, Philip Torr and Andrew Zisserman. Marseille : “In the Proceedings of the European Conference on Computer Vision.” Vol.**5303**, pp.**364-379, 2008**, Springer, Berlin, Heidelberg, ISBN: **978-3-540-88688-4.**

[7]. Dudani, S.A. “The Distance-Weighted k-Nearest-Neighbor Rule.” IEEE Transactions on Systems, Man, and Cybernetics 4, Vols. SMC-6, pp.**325-327**. May **21, 1975**.

[8]. Kiruthika, R., & Vijayakumar, V. “An Enhanced Fuzzy Based Linkage Clustering Algorithm (EFCA) in High Dimensional Data,” International Journal of Computer Sciences and Engineering, Vol.**8**, No **2**, pp.**12-17**, February **28, 2020**, E-ISSN: **2347-2693**.

[9]. Gupta, A., & Pratik, G. “Implementation of K-Means Clustering in Big Data Environment,” International Journal of Computer Sciences and Engineering, Vol.**7**, Issue.**11**, pp.**38-44**, November **10, 2019**, E-ISSN: 2347-2693.

[10]. Iqbal, R., Doctor, F., More, B., Mahmud, S., Yousuf, U. “Big Data analytics and Computational Intelligence for Cyber–Physical Systems: Recent trends and state of the art applications,” *Future Generation Computer Systems,* Vol.**105**, pp.**766-778, 2020.** https://doi.org/10.1016/j.future.2017.10.021.

[11]. Rawal, B., & Ruchi, A. “Improving Accuracy of Classification Based on C4.5 Decision Tree Algorithm Using Big Data Analytics.” *Advances in Intelligent Systems and Computing (AISC),*. Vol.**711**, No **7**, pp.**203-210.** ISBN **978-981-10-8055-5.**

[12]. Jothi, J. M., Arockiam, L “A Framework to enhance the Accuracy of Aspect level Sentiment Analysis in Big Data, ”*Conference on Inventive Computing and Informatics,”* Vol.**17**, pp.**452-457**, **2017,**

ISBN: 978-1-5386-4031-9.

[13]. Guo-Feng, F., Yan-Hui, G., & Jia-Mei, Z. “Application of the Weighted K-Nearest Neighbor Algortihm for Short-Term Load Forecasting,” *Multidisciplinary Digital Publishing Institute*, Vol.**12**, Issue**.5,** pp. **1-19, 2019.**

[14]. Chakravarthy, S. S., Bharanidharan, N., & Rajaguru, H. “Deep Learning-Based Metaheuristic Weighted K-Nearest Neighbor Algorithm for the Severity Classification of Breast Cancer,” *Innovation and Research in BioMedical Engineering(IRBM)*, Vol.**44**, Issue.**3** pp.**50-62**. June **15, 2023** ISSN 1959-0318.

[15]. Naz, G., Muhammad, A., Aldahmani, S., & Khan, Z.A “Weighted k-Nearest Neighbours Ensemble With Added Accuracy and Diversity,” *In the Proceedings of the IEEE International Conference on Big Data 2022* , Osaka., pp.**3341-3347, 2022.**

[16]. Wan, W., & Lee, H. J. “Deep feature representation and ball-tree for face sketch,” *International Journal of System Assurance Engineering and Management,* Vol.**11**, pp.**818-823**. September **3, 2019**.

[17]. Giang, N. T., Huong, H. L., Tai, H. P., & Hiep, X. H. “A parallel algorithm for determining the communication radius of an automatic trap based on balltree structure,” In the Proceedings of the Eighth International Conference on Knowledge and Systems Engineering (KSE) Hanoi Vietnam. pp.**139-143, 2016.**

[18]. Neeraj, K., Li, Z., & Nayar, S. “What is a good Nearest Neighbors Algorithm for Finding Similar Patches in Images.” *In the Proceedings of the 10th European Conference on Computer Vision - ECCV 2008,* pp.**364-378, 2008.** 978-3-540-88689-1.

[19]. Nwiabu, N., Alison, I., & Oyeneyin, B. “Modelling Case-Based Reasoning in Situation-Aware Disaster Management,” *Communications of the IIMA,* Vol.**19**, Issue.**1,** pp.**1-19, 2021.**

AUTHORS PROFILE

Banigo Azibator earned his B.Sc in Computer Science from the Niger Delta University, Bayelsa State Nigeria in 2015. He is currently gunning for a Master’s of Science from the Department of Computer Science, Rivers State University, Port Harcourt, Nigeria since 2021. His main research focuses on Big Data Processing, Data Structures and Algorithms. He has 1 year of research experience.

Dr. N. Nwiabu earned his Bachelor of Science form the Kwame Nkrumah University of Sc Science and Technology, Kumasi, Ghana in 2002 and a Master of Science from the University of Port Harcourt, Nigeria in 2006. He also obtained his Ph.D. from Robert Gordon University, Aberdeen, Scotland in 2009. He is currently working as an Associate Professor in the Department of Computer Science, Rivers State University, Nigeria since 2012. He is a member of the NCS and CPN since 2005, he is also a member of the IEEE computer society since 2011. He has numerous publications and conference papers in reputed international journals. His main research work focuses on Pipeline Monitoring, Decision support systems, and Artificial Intelligence among others. He has over 17 years of Teaching experience and over 10 years of Research Experience.

Dr. V.I Anireh earned his Bachelor of Science from the University of Nigeria Nsukka, Nigeria in 1990. He has a Post Graduate Diploma from the Rivers State University of Science and Technology, Nigeria. He attained his Master of Science and PhD from the University of Port Harcourt in 2007 and 2015 respectively. He is currently an Associate Professor in the Department of Computer Science Faculty of Science, Rivers State University, Nigeria, and has been working with the University since 2000. He has published over 40 papers and is an expert in Artificial Intelligence and Computer Networking.