Modified K-nearest Neighbor Algorithm with Variant K Values

Kalyani C. Waghmare[1], Balwant A. Sonkamble[2]
Department of Computer Engineering
Pune Institute of Computer Technology, Pune, India

Abstract—**In Machine Learning K-nearest Neighbor is a renowned supervised learning method. The traditional KNN has the unlike requirement of specifying 'K' value in advance for all test samples. The earlier solutions of predicting 'K' values are mainly focused on finding optimal-k-values for all samples. The time complexity to obtain the optimal-k-values in the previous method is too high. In this paper, a Modified K-Nearest Neighbor algorithm with Variant K is proposed. The KNN algorithm is divided in the training and testing phase to find K value for every test sample. To get the optimal K value the data is trained for various K values with Min-Heap data structure of 2*K size. K values are decided based on the percentage of training data considered from every class. The Indian Classical Music is considered as a case study to classify it in different Ragas. The Pitch Class Distribution features are input to the proposed algorithm. It is observed that the use of Min-Heap has reduced the space complexity nonetheless Accuracy and F1-score for the proposed method are increased than traditional KNN algorithm as well as Support Vector Machine, Decision Tree Classifier for Self-Generated Dataset and Comp-Music Dataset.**

Keywords—Classification; K-nearest Neighbor (KNN) classification algorithm; Indian Classical Music; Performance measures; Heap data structure

I. Introduction

The K-nearest neighbors is a simple and effective classification algorithm. The most important advantage is that the classification results can be easily interpreted. Despite all these advantages, it has shortcomings like high computational cost, large memory requirement, and equal-weighted features and in last deciding appropriate value of the input parameter K [1]. There are many variants of the KNN algorithm proposed to overcome these shortcomings. In [2, 3] the author proposed a weighted KNN. In [2] first learns weights for different attributes and according to the weights assigned, each attribute would affect the process of classification that much only. In [3] inverse of Euclidean distance is considered as the weight for load forecasting. In [4] various distance functions are implemented with KNN on a medical dataset with different types of attributes. In [5] authors used various pitch distributions as feature set for KNN with different distance functions in Raga Identification.

In [6] authors pointed out that traditional KNN has limitations to solve few problems like imbalance, noisy, sparse dataset. The authors proposed Hybrid KNN (HBKNN) to sort out these problems.

In KNN variations the researchers combined KNN with K-means clustering algorithm to reduce the computation complexity. In [7] authors applied this approach to improve accuracy in air quality assessment. This approach worked well for Big data as well in [8].

The basic assumption of the standard KNN is fixed K value for all data points to classify. However, many datasets have uneven distributions of data points, or even experts also not able to predict optimal K value. So many researchers proposed various methods for predicting k value. In [9] authors proposed a local mean representation-based k-nearest neighbor classifier (LMRKNN) method. In this method the representation-based distances calculated by the categorical k-local mean vectors instead of the simple majority vote for making the classification decision. The LMRKNN is outperformed on many real datasets downloaded from the University of California, Irvine (UCI), and Knowledge Extraction based on Evolutionary Learning (KEEL) repositories than traditional KNN. In [10] authors proposed an algorithm called Adaptive K-nearest neighbor (AdaKNN) algorithm which uses the density and distribution of the neighborhood of a test point and learns a suitable K for it with the help of artificial neural networks. This strategy for rightly classifying the test point is employed by Wettschereck and Dietterich in [11] in which, the value of K is determined for different portions of input space by applying cross-validation in its local neighborhood. The Ada-KNN2 is proposed as an extension to the Ada-KNN algorithm in which the neural network is replaced with a heuristic learning method based on local density indicator of a test point and information about its neighboring training points.

The large value of K would increase the computational cost and time in case of large data sets. To solve this problem, in [12] the variant value of K is proposed so that the early break of the algorithm can be possible, which ultimately saves computational time.

In [13] Adaptive KNN algorithm is developed by choosing optimal k for each item by maximizing its expected accuracy computed on similar points. The evaluation is done on three different datasets of Geo-Spatial Data.

In [14] the author employed a correlation Matrix, to reconstruct test data and assign different K values to the different test data points. The proposed algorithm achieved high accuracy and efficiency in applications of classification, regression, and missing data assertion.

The prediction of K value with the cross-validation method is usually time-consuming. In [15] authors introduced the training phase in the KNN classification algorithm and proposed a k*Tree method to learn different optimal k values for different test samples. The proposed K*Tree method reduced the running cost of the test phase. The efficient working of the proposed method is observed using 20 different real datasets.

In ICM the lots of work done in Raga recognition using KNN algorithm [5, 16, 17, 18, 19]. The researchers focused either on Features or compared using the different classifiers. The classifiers are used in their traditional form. In Data Mining as the application changes, the keen thinking about the parameters used in classifiers is required. The impact of these parameters on the performance also need to be observed.

The paper is organized as follows: Section II briefs about the proposed Modified Variant K Nearest Neighbor (MVKNN) algorithm. Section III gives details of experimental results and the analysis. Section IV Conclusion.

II. Proposed Algorithm

The Ragas is the central notion of Indian Classical Music. Usually, researchers find Pitch Class Distribution (PCD) features and apply classifiers. In literature authors used traditional classifiers. The traditional KNN works as follows.

Traditional KNN Algorithm

Procedure: - To find a class label for test input using KNN

Input: - D the set of Test samples, T the set of training samples,

Output: - P the class labels of test samples

Steps

```
1. SET 'K' Value
2. Read training samples
3. Read test samples
4. P= { }
5. For each d in D
 5.1 For each t in T
 5.1.1    Dis = distance(d,t)
 Endfor
 Endfor
 5.2 Sort Dis in ascending order
5.3 Select first 'K' entries
5.4 Find class labels of first 'K' entries
5.5 Allocate class label of maximum in first 'K' entries
```

In traditional KNN the K value is expected to provide in advanced which is very impractical. In this section, a Modified K Nearest Neighbor algorithm using variant K value for each test sample is proposed.

Modified Variant K Nearest Neighbor (MVKNN) algorithm using Min_Heap for Raga Identification

Procedure: - To find class label for test input
Input: - D the set of Test samples, T the set of training samples,
Output: - P the class labels of test samples

Steps

```
1. Read training samples
2. Read test samples
3. P= { }
4. For each c in C do // C the count of samples belong to each class in training set (T)
 4.1 C{c} = count (t) where class_label(t)==c
 Endfor
5. For each c in C do
 5.1 K{c} = round(C{c} *100/length(T) )
 5.2 M_K = max(K)
 Endfor
6. For each t in T do
 6.1 For each t1=t+1 in T do
 6.1.1 Dis = distance(d,t)
 6.1.2 Add Dis in min_heap[t][t1]of size M_K{c}
 6.1.3 Add 't1' in neighbor_heap[t] of size M_K{c}
 6.1.4 Add Dis in min_heap[t1][t]of size M_K{c}
 6.1.5 Add 't' in neighbor_heap[t1] of size M_K{c}
 Endfor
 Endfor
7. For each k in K do
 7.1 Class_neighbor{1.k} =findClass(neighbor_heap)
 7.2 P{d} = max(count(Class_neighbor))
 7.3 TP[k,c]= countif(class_label(t)==class_label(P))
 Endfor
8. For each c in C do
 8.1 K_test[c] = max(TP[c,k])
 Endfor
9. For each d in D do
 9.1 For each t in T do
 9.1.1 T_label = class_label(t)
 9.1.2 Dis = distance(d,t)
 9.1.3 Add Dis in min_heap of size K_test{T_label}
 9.1.4 Add 't' in neighbour_heap of size K{T_label}
 Endfor
 9.2 Class_neighbors{1..K} = findClass(neighbour_heap{1..K})
 9.3 P{d} = max(count(Class_neighbour))
 Endfor
Note:
\\ findClass(n) returns class label of samples in mean-heap
\\ max() returns class label appearing in 'K' nearest neighbor
\\ count() returns number of training samples class label is equal to predicted class label.
```

The traditional KNN does not have a training phase. It calculates the distance between every sample in test data with every sample in training data. The most nearest 'K' neighbors are identified for every sample based on distance. The class having maximum count belong to 'K' nearest Neighbor is assign to test sample.

In the proposed method algorithm is divided in two phases training and testing. In step 4.1 the samples per class are present in training data are calculated. The step5.1 calculates K value for each class label considering its percentage contribution in training data. In steps 6 and 6.1 the Euclidean distance is calculated between every sample in training data and stored in Min-Heap of size 2M_K. In this M_K is the maximum size of Heap.

Ones the Min Heap is ready, in step 7.1 the class labels are identified for every test sample from first entries in Min-Heap. The class label with maximum count will be assigned to the test sample in step 7.2. Step 7.3 counts the correctly classified samples for every class and stored in the TP array. Where TP gives True Positive values for each class. The steps 7.1 to 7.3 are executed for every distinct value of K which was calculated in step 5.1. The value of K will vary from minimum to maximum value of K for classes calculated in step 5.1. After calculating TP for all different ‘K’ values. The optimal 'K' value for every class is calculated by finding maximum trup positive count of every class. This completes the training phase. In the best case for all classes, the same 'K' may come. In the worst-case, every class will get different optimal 'K'.

In testing phase distance between every test sample and training sample is calculated and the Min-Heap is constructed for maximum optimal 'K' value which has got from the training phase. The nearest neighbors are identified from the first K entries in Min-Heap. The class label of maximum count of neighbors is assigned to the test sample.

The computational complexity of KNN is one of the limitations of KNN. In traditional KNN training phase is not available. The Time complexity of traditional KNN is $O(T * D) + O(D * T \log_2 T) + O(D * K)$. The complexity for calculating distance between every testing sample with training samples is $O(T * D)$. After calculating the distance between samples the sorting algorithm with average-case complexity N $\log_2$N is required to sort the distance array. So to sort D tuples the sorting complexity will be $O(D * T \log_2 T)$. To get ‘K’ nearest neighbor from sorted data will be O(K) which will be finally O(D*K) for D test samples. Even if instead of sorting the Heap data structure is used to get 'K' nearest neighbor, complexity will reduced to O(D*T$\log_2$T) + O(D*K$\log_2$T).

In MVKNN training and testing phases are introduced. The complexity of training phase is (O(T*(T+1)/2) + O(T*T$\log_2$K) + O(K*K$\log_2$K))}. In the worst-case, number of distinct K values, will become equal to the distinct value of percentage of records belonging to the number of classes present in Dataset, and in the best case, only the same K value is for all classes. The complexity to calculate the distance between every training sample with other training sample is O(T*(T-1)/2). To find the K nearest neighbor first it will create ‘T’ number of Min-Heap of 2K size. So the complexity to create the T number of Min-Heap with T elements of size 2K will be O(T*T$\log_2$K). To get K nearest neighbor Delete_min operation will be performed K times so its complexity will be O(K$\log_2$K).

The testing phase complexity will be O(D*T) + O(D*T$\log_2$K) + O(K$\log_2$K). The O(D*T) is complexity for calculating distance between every test sample with training sample. The O(D*T$\log_2$K) is complexity for creating Min-Heap of K size for T distance values. The Heap will be generated for every test sample.

The computation complexity of traditional KNN is higher than the computation complexity of the testing phase. If the complexity of both training and testing phase in MVKNN is considered then it is higher than traditional KNN but as we know the training of classifier is done only ones and are not required to perform whenever testing is executed. So based on this assumption the computational complexity of MVKNN testing phase is lower than traditional KNN.

The computational calculations can be understand more clearly by taking a small example.

Let us consider total samples 1000. Take a 70:30 ratio for training and testing. So T= 700 and D= 300, the number of classes present in the dataset are 8.

The total computations in traditional KNN will be.

Distance calculations = 210000.

Finding K nearest neighbor = 20, 32,949.

Total computations = 22, 42, 949.

The total computations in the training phase of MVKNN for the worst case will be.

Distance calculations = 2, 45, 350.

Finding K nearest neighbor = 25, 14, 780.

Total computations = 27, 60, 130.

The total computations in the testing phase of MVKNN for the worst case will be.

Distance calculations = 210000.

Finding K nearest neighbor = 10, 75, 379.

Total computations = 12, 85, 379.

This case study shows that computation for the training phase in the worst-case nearly one and half times of computations in traditional KNN. The testing computations are almost half of the computations in traditional KNN. So this work may conclude that MVKNN is computationally efficient than traditional KNN provided training should be performed occasionally.

The space complexity is also reduced. In traditional KNN O(D*T) memory will be required to store the distance in sorted array or Heap form. Wherein MVKNN space complexity for the training phase is O(T*$\log_2$K) and testing phase O(T*$\log_2$K).

III. Experimental Results

The proposed algorithm is presented as an extension of the traditional KNN algorithm. The performance of both algorithms is compared with our data set and CompMusic dataset.

In our dataset, 1450 samples of 8 different Ragas are present sung by different singers. The samples are stored in

.wav format with sampling frequency 44100Hz and 16bps. The frame size is considered as 20ms with 25% overlapping.

CompMusic dataset [16, 17] includes full-length audio recordings with the Raga label. It is a collection of several artists' vocal as well as instrumental performances. The clips were extracted from the live performances and CD recordings of 13 artists. Total 129 tunes for 08 ragas are considered. The dataset is downloaded as per instructions given in [20]. The duration of each tune averages 5-6 minutes. The tunes are converted to mono-channel, 44100 Hz sampling rate, 16 bit PCM.

The Pitch values are calculated as mentioned in [21]. The Pitch values are divided into 36 bins and constructed Pitch Class Distribution of every sample. Fig. 1 show the PCD for one sample of Raag Asavari. The PCD of the sample shows the frequency count of every bin. This sample is sung in the second octave so the Notes are present between bin numbers 13 to 25.

The PCD is calculated for all the samples and created a feature vector to give input to traditional KNN and MVKNN algorithm.

The experimentation for traditional KNN is done for varying K values from 1 to sqrt(T). The elbow method is applied and observed that after K=11, the accuracy is nearly constant up to K=20. Similarly with Decision Tree and SVM classifiers are also implemented with same datasets. Accuracy and F1 score is calculate as per following equations 1 and 2 respectively [22]. The results are documented in Table I.

$$Accuracy = \frac{(TP+TN)}{(TP+TN+FP+FN)} \tag{1}$$

$$F1 - Score = \frac{2*TP}{(2*TP+FP+FN)} \tag{2}$$

The PCD input is given to the MVKNN algorithm. For one instance the dataset is split into 30% testing and 70% training using train_test_split in Python. The training is performed for k=10, 11, 12, 13, 14 distinct 'K' values using Min-Heap. The confusion matrix containing True Positive, True Negative, False Positive and False Negative values is calculated for given dataset. The True Positive values are observed in every class for each 'K'. The 'K' having maximum True Positives is taken as an optimal K value for that class during the testing phase. In Table II the optimal K values are shown for every class for one instance.

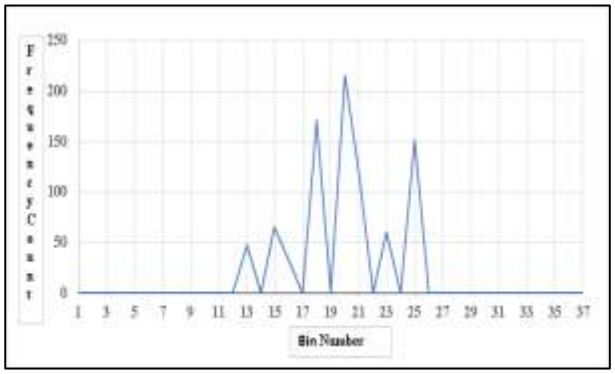

Fig. 1. PCD for One Sample.

TABLE I. RESULTS OF DECISION TREE, SVM CLASSIFIER

	Self-Generated Data	**CompMusic Data**
Decision Tree Accuracy	94.02%	86.33%
SVM Accuracy	84.74%	82.72%
Decision Tree F1-Score	79.30%	49.13 %
SVM F1-Score	38.99%	38.70%

TABLE II. OPTIMAL K VALUE FOR EACH CLASS

Class No.	1	2	3	4	5	6	7	8
K value	14	12	14	13	13	13	11	12

Table III shows a comparison of Accuracy and F1-score of traditional KNN and MVKNN for self-Generated data and CompMusic data. It is observed that Accuracy and F1-Score are improved for both datasets.

TABLE III. RESULTS OF KNN AND MVKNN

	Self-Generated Data	**CompMusic Data**
KNN Accuracy	89.46%	86.02%
MVKNN Accuracy	95.82%	89.45%
KNN F1-Score	57.90%	44.11 %
MVKNN F1-Score	83.28%	57.81%

The experimentation is done several times by taking an equal number of samples belonging to each class as well as by making imbalanced classes. It is observed that the variation in 'K' values always improved results than the same value of 'K'.

IV. CONCLUSION

In this paper, the survey of modified KNN algorithms is done. The KNN algorithm for variant K values for every test sample is proposed. The training phase is introduced to identify the optimal K value. The use of the Min-Heap data structure of 'K' size has reduced the space complexity. The algorithm was implemented using Indian Classical Music for classifying it based on the Raga. The PCD features of two different datasets are considered as an input vector. The Accuracy and F1-score measures are considered for comparing performance. The improvement in Accuracy and F1-score is observed using the proposed MVKNN algorithm in comparison with traditional KNN, Decision Tree and SVM. In Indian Classical Music, the repeating patterns play a very important role for Raga identification. In the future, the plan to apply the proposed algorithm on high dimensional feature vector of repeating patterns in a signal to improve the results of Raga identification.

REFERENCES

[1] Alka Lamba and Dharmender Kumar, "Survey on KNN and its Variants," in International Journal of Advanced Research in Computer and Communication Engineering Vol. 5, Issue 5, pp.430- 435, May 2016.

[2] Eui-Hong (Sam) Han, George Karypis and Vipin Kumar, "Text categorization using weight adjusted k-nearest neighbour classification," in Text categorization using weight adjusted k-nearest neighbour classification, Springer Berlin Heidelberg, 2001, pp. 53–65.

[3] Guo-Feng Fan, Yan-Hui Guo, Jia-Mei Zheng, and Wei-Chiang Hong, "Application of the Weighted K-Nearest Neighbour Algorithm for

Short-Term Load Forecasting," Energies 2019, 12, 916; doi:10.3390/en12050916, pp.1-19.

[4] Li-Yu Hu, Min-Wei Huang, Shih-Wen Ke, Chih-Fong Tsai, "The distance function effect on k-nearest neighbour Classification for medical datasets," Springer Plus, Vol. 5, issue 1, Dec.2017, pp.1-9.

[5] Parag Chordia and Senturk Sertan, "Joint recognition of Raag and Tonic in North Indian Music," in IEEE Computer Music Journal, Vol. 37, No-3, Sept. 2013, pp.82-98.

[6] Zhiwen Yu, Hantao Chen, Jiming Liu, Jane You, Hareton Leung, and Guoqiang Han, "Hybrid k-Nearest Neighbour Classifier," in IEEE Transactions on Cybernetics, Vol. 46, No. 6, June2016, pp.1263-1275.

[7] YANG Rui-jun, DING Dan-feng, YAN Feng, "Application of Improved KNN Algorithm in Air Quality Assessment," in HPCCT 2019, June 22–24, 2019, Guangzhou, China, pp.108-112.

[8] Hamid Saadatfar, Samiyeh Khosravi, Javad Hassannataj Joloudari, Amir Mosavi and Shahaboddin Shamshirband, "A New K-Nearest Neighbors Classifier for Big Data Based on Efficient Data Pruning," in Mathematics 2020, 8, 286.

[9] Jianping Gou, Wenmo Qiu, Zhang Yi, Yong Xu, Qirong Mao, and Yongzhao Zhan, " A Local Mean Representation-based K-Nearest Neighbour Classifier," ACM Transaction Intelligent System and Technology, Vol. 10, No. 3, May 2019,pp. 1-29.

[10] Sankha Subhra Mullick, Shounak Datta, and Swagatam Das, "Adaptive Learning-Based k-Nearest Neighbour Classifiers With Resilience to Class Imbalance," in IEEE Transaction on Neural Networks and Learning systems, Vol. 29, No. 11, Nov.2018, pp.5713-5725.

[11] Dietrich Wettschereck and Thomas G. Dietterich, "Locally adaptive nearest neighbour algorithms," Adv. Neural Inf. Process. Systems (NIPS), vol. 6, 1994, San Mateo, pp. 184–184.

[12] S. Ougiaroglou, A. Nanopoulos, A. N. Papadopoulos, Y. Manolopoulos, and T. Welzer-Druzovec, "Adaptive k-Nearest-Neighbour Classification Using a Dynamic Number of Nearest Neighbours," in Advances in Databases and Information Systems, Y. Ioannidis, B. Novikov, and B. Rachev, Eds. Springer Berlin Heidelberg, 2007, pp. 66–82.

[13] Mark Kibanov, Martin Becker, Juergen Mueller, Martin Atzmueller, Andreas Hotho, Gerd Stumme, "Adaptive kNN using Expected Accuracy for Classification of Geo-Spatial Data," in Proceedings of Symposium on Applied Computing (SAC), 2017, pp.1-9.

[14] Shichao Zhang, Xuelong Li, Ming Zong, Xiaofeng Zhu, and Debo Cheng, "Learning k for KNN Classification," in ACM Transactions on Intelligent Systems and Technology, Vol. 8, No. 3, jan. 2017, pp.1-19.

[15] Shichao Zhang, Xuelong Li, Ming Zong, Xiaofeng Zhu, and Ruili Wang, "Efficient KNN Classification With Different Numbers of Nearest Neighbours," in IEEE Transactions on Neural Networks and Learning Systems, vol. 29, no. 5, May 2018, pp. 1774-1785.

[16] Sankalp Gulati, J. Serra, V. Ishwar, S. Senturk, Xavier Serra, " Phrased based Raga Recognition using vector space modelling," in IEEE International Conference on Acoustics, Speech, and Signal Processing, Shanghai, China 20th -25th Mar. 2016, pp.66-70.

[17] Sankalp Gulati, J. Serra, K. Ganguli, S. Senturk, Xavier Serra, "Time-Delayed Melody Surfaces for Raga Recognition," in Proceedings of 17th International Society for Music Information Retrieval Conference, New York, USA, 7th -11th Aug.2016, pp.751-757.

[18] Parag Chordia and Alex Rae, "Raga recognition using Pitch Class and Pitch Class Dyad Distribution," in 8th International Society of Music Information Retrieval Conference, Vienna, Austria, 2007, pp. 431-436.

[19] Gopala Koduri, Sankalp Gulati, Preeti Rao, " A survey of Raaga Recognition techniques and improvements to the state-of-the-art," in Conference of sound and Music, Padova, Italy, 6th -9th July 2011, pp.1-4.

[20] https://compmusic.upf.edu/node/300.

[21] Kalyani C. Waghmare, Balwant A. Sonkamble, "Timbre with Note Based Features for Improving Performance of Music Classification," in International Journal of Advanced Science and Technology, Vol. 29, No. 3, (2020), pp. 10328 – 10338.

[22] Jiawei Han, Micheline Kamber and Jian Pei, "Data Mining: Concepts and Techniques 3rd ed.," in the Morgan Kaufmann Series in Data Management Systems, Morgan Kaufmann Publishers, July 2011, ch-8, sec-8.5, pp.-364-370.