Research Article

A medoid-based weighting scheme for nearest-neighbor decision rule toward effective text categorization

Avideep Mukherjee[1] · **Tanmay Basu**[2]

Received: 1 October 2019 / Accepted: 10 April 2020 / Published online: 4 May 2020

Abstract
The *k*-nearest-neighbor (*k*NN) decision rule is a simple and robust classifier for text categorization. The performance of *k*NN decision rule depends heavily upon the value of the neighborhood parameter *k*. The method categorize a test document even if the difference between the number of members of two competing categories is one. Hence, choice of *k* is crucial as different values of *k* can change the result of text categorization. Moreover, text categorization is a challenging task as the text data are generally sparse and high dimensional. Note that, assigning a document to a predefined category for an arbitrary value of *k* may not be accurate when there is no bound on the margin of majority voting. A method is thus proposed in spirit of the nearest-neighbor decision rule using a medoid-based weighting scheme to deal with these issues. The method puts more weightage on the training documents that are not only lie close to the test document but also lie close to the medoid of its corresponding category in decision making, unlike the standard nearest-neighbor algorithms that stress on the documents that are just close to the test document. The aim of the proposed classifier is to enrich the quality of decision making. The empirical results show that the proposed method performs better than different standard nearest-neighbor decision rules and support vector machine classifier using various well-known text collections in terms of macro- and micro-averaged *f*-measure.

Keywords *k*NN classifier · Text categorization · Data mining · Machine learning

1 Introduction

The task of *k*NN decision rule is to assign a test document to a particular category using a set of training documents. The method first finds the *k*-nearest neighbors of the test document from the training set by using a similarity measure. Therefore, the category of the test document is determined by taking a majority vote among these *k*-nearest neighbors [1, 2]. Thus the performance of *k*NN decision rule is heavily influenced by the neighborhood parameter *k* [3]. Different values of *k* can change the result of text categorization and hence choice of *k* is crucial for effective result. Moreover, text categorization is a challenging task as the text data are generally sparse and high dimensional. Hence, assigning a document to a predefined category for an arbitrary value of *k* may not be accurate when there is no bound on the margin of majority voting. The cross-validation technique is generally used to estimate an optimal value of *k* [4], but choosing an optimal *k* which provides satisfactory results for all test documents is still a difficult job. Moreover, a slight change in the value of *k* also leads to different results. For example, consider a two-class classification problem. Let there be 8 documents in the training set and d_t be a test document. Let *A* and *B* be the two

The work had done when the authors were affiliated to Ramakrishna Mission Vivekananda Educational and Research Institute, Belur Math, West Bengal, India.

✉ Tanmay Basu, welcometanmay@gmail.com; Avideep Mukherjee, mukherjeeavideep@gmail.com | [1]Indian Institute of Technology Kanpur, Kanpur, India. [2]Institute of Cancer and Genomic Sciences, University of Birmingham, Birmingham, UK.

categories. According to *k*NN algorithm, the training documents are arranged according to non-decreasing order of similarity with d_t. Let the labels of the categories of the ordered training documents are given as *{A, A, B, B, A, B, B, A}*. It can be seen that for $k = 5$, d_t is categorized to *A*, for $k = 6$ there is a tie and for $k = 7$, d_t belong to *B*. It is clear from this example that simple majority voting rule may not be useful for text categorization. In principle, when there is more or less same representation from the competing categories among the nearest neighbors, it is preferable to keep the test document unclassified rather than making a wrong judgment [5].

A tweak on the *k*NN (T*k*NN) decision rule have been proposed by Basu et al. to overcome these issues [5]. The method puts a bound on the majority voting of *k*NN by using a predefined threshold to enhance the confidence of the majority voting process. It starts with an arbitrary *k* and increases the value of *k* until it can categorize a test document. A document is thus categorized, if the difference between the number of documents of two competing categories is greater than a given threshold. The method does not require the knowledge of neighborhood parameter *k* to execute *k*NN. However, this method does not check the similarity of the documents when increasing the span of neighborhood, which is crucial. In principle, the similarity between the test document and the training documents should be checked to expand the neighborhood as the term-document matrices are generally sparse and high dimensional. The other widely used variant of *k*NN decision rule is distance-weighted *k*NN decision rule [6]. The method gives different weights to different *k* nearest neighbors based on their distances with the test document, where the closer neighbors get higher weights. Likewise, *k*NN decision rule this method too put no bound on the margin of majority voting for decision making. A method is thus desirable to overcome these limitations of the *k*NN decision rules and its variants for effective text categorization.

A nearest-neighbor decision rule is proposed here in spirit of the weighted *k*NN and T*k*NN decision rules. The proposed decision rule forms the neighborhood of a test document by considering the documents from the training set that are closely related to both medoid of a category and the test document. The medoid of a category is a representative document whose average dissimilarity to all the other documents in that category is minimal [7, 8]. Note that medoids are always restricted to be the members of a data set. The method first finds the medoid of each category in the data set and subsequently it identifies the training documents that are closely related to the medoid of individual categories and the test document. These training documents constitute the neighborhood of the test document. The weight of a training document in that neighborhood is computed by considering the distance of that document from the medoid and also from the test document. Thereafter the first few neighbors are considered and the weights of these documents belonging to the individual categories are aggregated. The test document is then assigned to a particular category that has the maximum aggregated weight and this weight is greater than the weight of its competing categories by a given threshold. The method continues until this condition is not satisfied or the method has checked all the documents in the neighborhood. The objective of the proposed decision rule is to enrich the quality of the decision making. In worst case, it may happen that the proposed decision rule has examined all the neighbors of the test document, but could not take a decision. The test document will remain unclassified in such cases. Note that, in practice it is better not to take a decision when we are not sure about it. The proposed technique is developed in this spirit. The performance of the proposed method is compared with different standard nearest-neighbor decision rules and support vector machine classifier using standard text collections. The empirical results show that the proposed method outperforms the state of the arts in terms of macro- and micro-averaged *f*-measure.

The paper is organized as follows. The related works to this study are described in Sect. 2. Section 3 explains the vector space model for representation of text data. The proposed method is described in Sect. 4. Section 5 presents the experimental evaluation. Finally, we conclude with the scopes of future works in Sect. 6.

2 Related works

Text categorization is the problem of assigning predefined categories to the new documents. It assigns a new document to a particular category when the document is similar with more number of documents of that category than any other category [9]. A number of methods have been developed for effective text categorization [9]. Support vector machine (SVM) was introduced to solve two class classification problems using the structural risk minimization principle [10]. In its simplest linear form, SVM finds a hyperplane that separates the documents of two different categories with maximum margin [11]. Joachim reported an efficient implementation of SVM and its application in text categorization on Reuters-21578 corpus [12]. The *k*NN decision rule is a simple and effective similarity-based classifier and it has performed well for text categorization [13, 14]. Cover and Hart [1] introduced the *k*NN decision rule, where a test sample is assigned to a particular category, which has the maximum number of representative

training samples among the k nearest neighbors of the test sample.

The other widely used variant of kNN decision rule is distance weighted kNN decision rule [6]. The method assigns different weights to different k nearest neighbors based on their distances with the test document, where the closer neighbors get higher weights. Let $\mathbf{d_1}, \mathbf{d_2}, \ldots, \mathbf{d_k}$ be the k nearest neighbors of a test document, say, $\mathbf{d_t}$. Let the corresponding distances of these neighbors from $\mathbf{d_t}$ is denoted by $\rho(\mathbf{d_j}, \mathbf{d_t})$, $\forall j = 1, 2, \ldots k$, where ρ is a distance function. The weight w_j associated with the jth nearest neighbor $\mathbf{d_j}$ is defined as

$$w_j = \begin{cases} \frac{\rho(\mathbf{d}_k,\mathbf{d}_t)-\rho(\mathbf{d}_j,\mathbf{d}_t)}{\rho(\mathbf{d}_k,\mathbf{d}_t)-\rho(\mathbf{d}_1,\mathbf{d}_t)} & \text{if} \quad \rho(\mathbf{d}_k,\mathbf{d}_t) \neq \rho(\mathbf{d}_1,\mathbf{d}_t) \\ 1 & \text{if} \quad \rho(\mathbf{d}_k,\mathbf{d}_t) = \rho(\mathbf{d}_1,\mathbf{d}_t) \end{cases} \tag{1}$$

The test document $\mathbf{d_t}$ is assigned to the category for which the sum of the weights of the representative documents of the category among these k nearest neighbors is maximum [6]. The major limitation of this method is that it also suffers from the influence of neighborhood parameter k. Different values of k may lead to different assignments of categories to the test document.

Gowda et al. have developed the condensed nearest-neighbor (CNN) technique [15], which eliminates similar or redundant data sets that do not add extra information. Although it reduces the memory requirements and recognition rate while improving query time, it still poses the problem of computational cost. The reduced nearest-neighbor (RNN) algorithm [16] does an extra job over CNN by removing the samples that are independent of the training set. Rank-based kNN (RNN) decision rule is quite effective in case of data with huge variations between features [17]. Bagui et al. [17] have proposed a generalization of the RNN rule by assigning ranks to the training data for each category. However, these methods have never used for text categorization.

Guan et al. have proposed a modification on kNN decision rule for text categorization, which considers mostly the documents that lie on the boundary region of individual categories in decision making and ignores the other documents [18]. The efficiency and effectiveness of the method is demonstrated using the standard Reuters corpus [18]. Tan has proposed a method called neighbor-weighted K-nearest neighbor (NWKNN) for unbalanced text categorization problems [19]. Instead of balancing the training data, NWKNN assigns high weight to the neighbors belong to the categories containing a few documents and provides small weight to the neighbors belong to the categories containing large number of documents [19].

Basu et al. have proposed the TkNN decision rule by putting a bound on the majority voting process of the kNN decision rule as discussed earlier [5]. TkNN rule restricts the majority voting of kNN by a predefined positive integer threshold, say β, to assign a test document to a category. The method starts with β number of neighbors, i.e., $k = \beta$. Subsequently, it checks whether the difference between the number of members of the best and the second-best competing categories is β. If so, then the test document is categorized to the best competing category by this rule. Otherwise, the value of the neighborhood parameter k is increased by one. Thus the process continues till a decision is made or it reaches the last document of the set of neighbors. The set of neighbors is literally the training set ordered as per the distance with the test document. If the test document is not categorized till the process checks all the documents of the set of neighbors, then it remains unclassified. However, this method does not consider the distance between the neighbors and the test document when performing the majority voting for decision making. A training document that is far away from the test document can take part in decision making by this rule, which is not desirable.

3 Representation of text data

The length of different documents in a corpus are different. Note that here length means the number of terms in a document. It is very difficult to find the similarity between two document vectors of different dimensions (length). Therefore, it is necessary to maintain the uniform length of all the documents in the corpus. Several models have been introduced in the information retrieval literature to represent the document data sets in the same frame [20, 21].

The *vector space model* enables efficient analysis of huge document collections in spite of its simple idea [21]. It was originally introduced for indexing and information retrieval, but is now used in several text categorization and clustering techniques as well as in most of the currently available document retrieval systems [22].

Let us assume that the number of documents in the corpus is n and the number of terms is m. Let us also assume that the ith term is represented by t_i and the number of times the term t_i occurs in the jth document is denoted by tf_{ij}, $i = 1, 2, \ldots, m; j = 1, 2, \ldots, n$. Document frequency df_i is the number of documents in which t_i occurs. Inverse document frequency $idf_i = \log(\frac{n}{df_i})$, determines how frequently a term occurs in the corpus. The weight of t_i in the jth document, denoted by w_{ij}, is determined by combining the term frequency with the inverse document frequency [22] as follows:

$$w_{ij} = tf_{ij} \times idf_i = tf_{ij} \times \log(\tfrac{n}{df_i}), \quad \begin{matrix} \forall i = 1, 2, \ldots, m \text{ and} \\ \forall j = 1, 2, \ldots, n \end{matrix} \tag{2}$$

The documents can be efficiently represented using the vector space model in most of the text categorization and clustering algorithms [22]. In this model each document d_j is considered to be a vector $\mathbf{d_j}$, where the ith component of the vector is w_{ij}, i.e., $\mathbf{d_j} = (w_{1j}, w_{2j}, \ldots, w_{mj})$.

The similarity between two documents is achieved through some distance function. Given two document vectors $\mathbf{d_i}$ and $\mathbf{d_j}$, it is required to find the similarity (or dissimilarity) between them. Various similarity measures are available in the literature, but the commonly used measure is cosine similarity between two document vectors [20], which is given by

$$\cos(\mathbf{d_i}, \mathbf{d_j}) = \frac{\mathbf{d_i}.\mathbf{d_j}}{|\mathbf{d_i}|\,|\mathbf{d_j}|} = \frac{\sum_{k=1}^{m}(w_{ki} \times w_{kj})}{\sqrt{\sum_{k=1}^{m} w_{ki}^2 \times \sum_{k=1}^{m} w_{kj}^2}}, \quad \forall i, j \tag{3}$$

Note that the weight of each term in a document is nonnegative. As a result the cosine similarity is nonnegative and bounded between 0 and 1, both inclusive. $\cos(\mathbf{d_i}, \mathbf{d_j}) = 1$ means the documents are exactly similar and the similarity decreases as the value goes to 0. An important property of the cosine similarity is its independence of document length. Thus cosine similarity has become popular as a similarity measure in the vector space model [23]. The vector space model is used here to represent a document vector.

4 A medoid-based nearest-neighbor decision rule for text categorization

In this work, a medoid-based weighting scheme is proposed to overcome the influence of the boundary documents on nearest-neighbor decision rule. A *medoid* is a document of a particular category whose average similarity to all the other documents in the category is maximal [8, 24]. Let $D = \{\mathbf{d_1}, \mathbf{d_2}, \ldots, \mathbf{d_n}\}$ be the set of n document vectors corresponding to n documents in the training corpus. Here $\mathbf{d_i} \in \mathbb{R}^m, \forall i = 1, 2, \ldots, n$ are generated from the raw texts following the tf-idf weighting scheme of vector space model [20]. Let us consider there are r categories in the training corpus, say, $C_1, C_2, \ldots, C_r$. The medoid of the documents of a particular category, say, C_j is defined as

$$\hat{\mathbf{d}}_\mathbf{j} = \underset{\mathbf{d} \in C_j}{\operatorname{argmax}} \sum_{\forall d_i \in C_j} \Psi(\mathbf{d}, \mathbf{d_i}), \quad \forall j = 1, 2, \ldots, r \tag{4}$$

Note that Ψ is a normalized similarity measure i.e., $\Psi \in [0, 1]$, where 1 indicates the highest similarity and the similarity decreases when the value decreases to 0. In the experimental analysis of this article, Ψ is treated as cosine similarity.

4.1 Medoid-based weighting scheme

Let $\mathbf{d_t}$ be the test document, whose category is to be identified. The proposed method considers a training document as effective neighbor of $\mathbf{d_t}$, whose similarity with $\mathbf{d_t}$ is greater than a predefined threshold and the similarity between $\mathbf{d_t}$ and medoid of a particular category. It forms the set of effective neighbors (EN) of $\mathbf{d_t}$ as follows:

$$\begin{aligned}\text{EN} = \{d \in D :\ & \Psi(\mathbf{d_t}, \mathbf{d}) \geq \theta \text{ and } \Psi(\mathbf{d_t}, \mathbf{d}) \geq \Psi(\mathbf{d_t}, \hat{\mathbf{d}}_\mathbf{j}), \\ & j = 1, \ldots, r\}\end{aligned} \tag{5}$$

This indicates that the effective neighbors of a test document are those training documents which lie between the test document and medoid of individual categories and have sufficient content similarity with d_t. Here $\theta > 0$ is a threshold on document similarity and it ensures that the documents in EN have sufficient content similarity with d_t. The weight of a particular document, say, $d \in$ EN in terms of d_t is defined as

$$W(\mathbf{d}, \mathbf{d_t}) = \Psi(\mathbf{d}, \mathbf{d_t}) \times \Psi(\mathbf{d}, \hat{\mathbf{d}}_\mathbf{j}), \tag{6}$$

Here $d \in C_j, \forall\, j = 1, 2, \ldots, r$ and $\hat{d}_j$ is the medoid of C_j. It may be noted that $W(\mathbf{d}, \mathbf{d_t}) \in [0, 1]$.

- The highest value of $W(\mathbf{d}, \mathbf{d_t})$ is 1, which indicates that d is close to both $\hat{d}_j$ and d_t.
- The value of $W(\mathbf{d}, \mathbf{d_t}) = 0$ when $\Psi(\mathbf{d}, \mathbf{d_t}) = 0$.
- When d is close to $\hat{d}_j$ but, far from d_t i.e., $\Psi(\mathbf{d}, \hat{\mathbf{d}}_\mathbf{j})$ is high, but $\Psi(\mathbf{d}, \mathbf{d_t})$ is low then $W(\mathbf{d}, \mathbf{d_t})$ will be low.

Note that $\Psi(\mathbf{d}, \mathbf{d_t})$ in Eq. 6 indicates the similarity between the test document and a training document, whereas $\Psi(\mathbf{d}, \hat{\mathbf{d}}_\mathbf{j})$ denotes the similarity between the same training document and the medoid of the category of this training document. The product of these two similarity values will be high only when their individual values are very high. Thus this weighting scheme ensures that the training documents which are not only close to the test document but also close to the medoid of the corresponding categories are given higher preference than the other documents in EN to take part in the majority voting of the proposed decision rule to categorize the test document.

Algorithm 1 Proposed Nearest Neighbor Decision Rule Using Medoid Based Weighting Scheme

Input: a) $D = \{d_1, d_2, ..., d_n\}$ be n documents of training set.
b) A set of r predefined categories, $C = \{C_1, C_2, ..., C_r\}$
c) Let d_t be a particular test document, η be the initialization parameter on neighborhood of d_t.
d) Ψ and γ respectively be the similarity measure and threshold on weights of the categories.

Steps of the Algorithm:

Find the median of each category, say $\hat{\mathbf{d}}_\mathbf{j}$, $\forall j = 1, 2, ...r$ following equation 4
$EN \leftarrow \emptyset$
for $i \leftarrow 1$ to n **do**
  for $j \leftarrow 1$ to r **do**
    if $\Psi(\mathbf{d_t}, \mathbf{d}) \geq \theta$ and $\Psi(\mathbf{d_t}, \mathbf{d}_i) > \Psi(\mathbf{d_t}, \hat{\mathbf{d}}_\mathbf{j})$ **then**
      $EN \leftarrow EN \bigcup \mathbf{d}_i$
    end if
  end for
end for
Rearrange EN in non-decreasing order of similarity with respect to $\mathbf{d_t}$ and rearrange D and C accordingly
$L \leftarrow \eta$
$S_L \leftarrow$ First L documents from EN
for all $d \in S_L$ **do**
  for $j \leftarrow 1$ to r **do**
    if $\mathbf{d} \in C_j$ **then**
      Compute $W(\mathbf{d}, \mathbf{d_t}))$ following equation 6
      $W(C_j) = W(C_j) + W(\mathbf{d}, \mathbf{d_t}))$
    end if
  end for
end for
$Category(d_t) \leftarrow \emptyset$
while $L \leq EN$ **do**
  $W(C_{max1}) \leftarrow \max\{W(C_1), W(C_2), ..., W(C_m)\}$
  $W(C_{max2}) \leftarrow \max\{\{W(C1), W(C_2), ..., W(C_m)\} - \{W(C_{max1})\}\}$
  if $\frac{W(C_{max1})}{|C_{max1}|} - \frac{W(C_{max2})}{|C_{max2}|} \geq \gamma$ **then**
    $Category(d_t) = C_{max1}$
    Return $Category(d_t)$
  else
    $L = L + 1$
    $S_L = S_L \bigcup \mathbf{d}_L$
    for $j \leftarrow 1$ to r **do**
      if $\mathbf{d_L} \in C_j$ **then**
        Compute $W(\mathbf{d_L}, \mathbf{d_t}))$ following equation 6
        $W(C_j) = W(C_j) + W(\mathbf{d_L}, \mathbf{d_t}))$
      end if
    end for
  end if
end while
Return $Category(d_t)$

4.2 Proposed text categorization technique

In the first stage, the proposed method finds the medoids of the individual categories. Therefore it creates the effective neighborhood, EN of the test document $\mathbf{d_t}$. EN is then rearranged in non-increasing order of similarity values between $\mathbf{d_t}$ and individual members of EN. The method considers the first L documents of EN and stores them in S_L to categorize $\mathbf{d_t}$. The initial values of L is predefined and it is denoted as η in Algorithm 1. Subsequently, $W(\mathbf{d_i}, \mathbf{d_t}))$ is computed for each document $\mathbf{d_i} \in S_L$. The weight of a category, C_j, $j = 1, 2, \ldots, r$ is computed by aggregating the weights of the individual documents of C_j as follows.

$$W(C_j) = \sum_{\forall \mathbf{d_i} \in C_j, S_L} W(\mathbf{d_i}, \mathbf{d_t})), \quad \forall j = 1, 2, \ldots, r \tag{7}$$

The weights of the maximum and the second maximum category are obtained from the set of category weights $\{W(C_j) : j = 1, \ldots, r\}$. Let they be called $W(C_{\text{max1}})$ and $W(C_{\text{max2}})$ respectively. These weights are then divided by the total number of documents of the respective categories, i.e., $|C_{\text{max1}}|$ and $|C_{\text{max2}}|$ respectively to get normalized scores. The proposed decision rule assigns the test document to the best category, when the normalized weights of the best category and its competing category is differed by a predefined threshold, say, γ, i.e., if $\frac{W(C_{\text{max1}})}{|C_{\text{max1}}|} - \frac{W(C_{\text{max2}})}{|C_{\text{max2}}|} > \gamma$. If this criterion is not satisfied then the value of L is increased by 1 and the weight of the next document in EN is computed. The method is repeated until the aforesaid condition is satisfied or the method has checked all the members of EN. In worst case, d_t is kept unclassified, if the method cannot categorize it after exploring all the documents in EN. The steps of the proposed method is presented in Algorithm 1.

Note that $\eta = 1$ implies one nearest-neighbor decision rule and thus the minimum value of η is 2. The value of η can be at most |EN|. Note that γ is ensuring sufficient difference between the weights of majority category and its competing categories and thus it is enriching the confidence of the decision making. The value of γ is at least 0. As the category weights are normalized between 0 and 1, the maximum value of γ cannot be greater than 1. Thus the value of γ lies between 0 and 1, both inclusive.

5 Experimental evaluation

5.1 Description of data

The proposed method and the state of the arts are evaluated using seven text corpora. All the corpora are developed by Karypis and Han [25] and these are mostly collected from TREC.[1] These corpora consists of documents as less as 204 to at most 4069, and has number of terms ranging from 3758 to 18,483. The number of categories of these corpora vary from 5 to 25. The overview of the corpora are presented in Table 1.

[1] https://trec.nist.gov/.

Table 1 Overview of the corpora

Dataset	#Documents	#Terms	#Categories
re1	1657	3758	25
reviews	4069	18,483	5
tr45	690	8261	10
tr41	878	7454	10
tr11	414	6429	9
tr23	204	5832	6
tr12	303	5804	8

5.2 Evaluation techniques

The performance of the proposed method and the state-of-the-art classifiers are evaluated using the standard precision, recall and *f*-measure [13]. The precision and recall for two class classification problem can be computed as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{8}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{9}$$

Here TP stands for *true positive* and it counts the number of documents correctly predicted to the positive category. FP stands for *false positive* and it counts the number of documents that actually belong to the negative category, but predicted as positive (i.e., *falsely predicted as positive*). FN stands for *false negative* and it counts the number of documents that actually belong to the positive category, but predicted as negative. The *f*-measure combines recall and precision with an equal weight in the following form:

$$F\text{-Measure} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} \tag{10}$$

The closer the values of precision and recall, the higher is the *f*-measure [26]. *F*-measure becomes 1 when the values of precision and recall are 1 and it becomes 0 when precision is 0, or recall is 0, or both are 0. Thus *f*-measure lies between 0 and 1. A high *f*-measure value is desirable for good classification [26].

There are two conventional methods to generalize these evaluation functions for multi class classification problem, namely *macro-averaging* and *micro-averaging* [27]. The macro-averaged measure finds the precision and recall score for each class, and then these scores for all the categories are aggregated [13]. The micro-averaged measure individually aggregates the true positives, false positives and false negatives over all the categories and then finds the precision and recall [13]. We have used both macro-averaged and micro-averaged *f*-measure to evaluate the performance of the classifiers.

5.3 Experimental setup

The performance of the proposed method is compared with SVM [10], *k*NN [1], weighted *k*NN [6] and T*k*NN [5, 28] classifiers. It may be noted that SVM has been widely used for text categorization in the last few years [29] and so that the performance of SVM is reported in this work for comparison. The concept of the proposed method has been introduced in spirit of nearest-neighbor decision rule, and therefore the performance of the proposed method is compared with *k*NN, weighted *k*NN and T*k*NN classifiers. The corpora used here have no specific training and test sets. Therefore we have randomly split the data sets into two parts—80% is considered as training set and the rest as test set. The random split is done in such a way that ensures the representative documents of each category in both training and test set. The training set is used to train the classifiers and the test set is used to evaluate the performance of individual classifiers.

The proposed algorithm has two major parameters: The first one is η, which is used to initialize the neighborhood of the test document and the other one is γ, which is used as the bound on the weights of the competing categories. It may be noted that $\eta \in [2, 3, \ldots, |\text{EN}|]$, where EN is the set of effective neighbors of the test document. In the experiments $\eta = 3$ is used. The value of γ is experimentally fixed by using grid search-based tenfold cross-validation technique on the training set by using $\gamma = 0.1, 0.2, 0.3, 0.4, 0.5$. The value of θ is fixed as 0.3.

The parameters of the state-of-the-art classifiers, e.g., *k*NN, SVM etc. are tuned using grid search-based tenfold cross-validation technique on the training set. In case of *k*NN and weighted *k*NN classifiers, the value of *k* is chosen by varying it from 2 to 20. The state-of-the-art classifiers are implemented using Scikit-learn[2] [30], a machine learning tool in Python.

5.4 Analysis of results

The performance of the proposed method and state-of-the-art classifiers on different text corpora are shown in Tables 2 and 3 respectively using micro-averaged and macro-averaged *f*-measure. The raw text data are transformed into feature vectors using the vector space model as described in Sect. 3. The value of the parameter *k* that has been selected by the tenfold cross-validation technique on training set to perform *k*NN and weighted *k*NN algorithms on the test documents are shown in Tables 2

[2] http://www.scikit-learn.org.

Table 2 Micro-averaged *F*-measure of the proposed classifier and state-of-the-art classifiers on various text corpora

Dataset	SVM	*k*	*wk*NN	*k*	*k*NN	*L* (avg)	*tk*NN	*L* (avg)	γ	Proposed
re1	0.77	12	0.83	15	0.82	5	0.80	3	0.01	**0.84**
reviews	0.80	14	0.80	15	0.79	3	0.80	10	0.025	**0.83**
tr45	0.93	3	0.91	6	0.87	5	0.91	6	0.025	**0.93**
tr41	**0.96**	4	0.95	8	0.91	5	0.92	4	0.10	0.95
tr11	0.90	2	0.87	2	0.87	4	0.84	3	0.05	**0.92**
tr23	0.87	3	0.90	6	0.85	3	0.85	5	0.10	**0.91**
tr12	**0.85**	2	0.82	2	0.84	4	0.81	3	0.025	**0.85**

Bold value indicates the best performance (i.e., the highest value) in each row

Table 3 Macro-averaged *F*-measure of the proposed classifier and state-of-the-art classifiers on various text corpora

Dataset	SVM	*k*	*wk*NN	*k*	*k*NN	*L*(avg)	*tk*NN	*L*(avg)	γ	Proposed
re1	0.61	12	0.72	15	0.71	5	0.70	3	0.01	**0.73**
reviews	0.52	14	0.52	15	0.52	3	0.43	10	0.025	**0.53**
tr45	0.86	4	0.87	3	0.82	5	0.86	2	0.025	**0.90**
tr41	**0.95**	3	0.84	7	0.82	5	0.93	4	0.10	0.94
tr11	0.78	3	0.75	5	0.74	4	0.69	5	0.05	**0.81**
tr23	0.88	4	0.91	6	0.83	3	0.82	7	0.10	**0.92**
tr12	0.85	5	0.71	2	0.63	4	0.81	6	0.025	**0.86**

Bold value indicates the best performance (i.e., the highest value) in each row

and 3 beside individual *f*-measure values. The values of *L*, the average of the number of nearest neighbors of all test documents of the individual corpora for both T*k*NN and the proposed method are presented in Tables 2 and 3 beside individual *f*-measure values. The value of γ of the proposed technique for the individual corpora are also reported in these tables.

Tables 2 and 3 show that the proposed method performs better than the other classifiers for all the data sets except *tr41*. For the *tr41* data set, SVM performs better than the proposed method in terms both macro-averaged and micro-averaged *f*-measure scores. It can be seen from Tables 2 and 3 that there are 56 comparisons for the proposed method and the proposed one has performed better than the other methods in 51 cases. The statistical significance of these results is to be tested. For example, for *tr12*, the macro-averaged *f*-measure of SVM is 0.85 and for the proposed method it is 0.86, so we have to test whether this difference is statistically significant.

A paired *t* test is suitable for testing the equality of means when the variances are unknown. A suitable test statistic is described and tabled in [31] and [32], respectively. The statistic uses the null hypothesis of equal means assuming unequal variance on same sample size. The statistic *t* is measured as $t = \frac{\mu_1 - \mu_2}{\sqrt{\sigma_1^2/n_1 + \sqrt{\sigma_2^2/n_2}}}$, where μ_1, μ_2 are the means, σ_1, σ_2 are the standard deviations and n_1, n_2 are the number of observations [31]. It has been found that the results are statically significant in 39 out of 51 cases, where the proposed technique performs better than the other methods for the level of significance 0.05. The test results are statistically significant in 3 out of 5 cases for the same level of significance, when other methods have an edge over the proposed technique. Thus in 92.85% cases the performance of the proposed technique is significantly better than the other classifiers. The effectiveness of the proposed method can be observed from these results.

The robustness of different classification algorithms can be determined by using the idea of Friedman [33]. Robustness of a classifier *h* for a particular data set is defined as $E_h = E_h/E_0$, where E_h is either macro-averaged or micro-averaged *f*-measure of *h* and $E_0 = \max_h E_h$ [28]. The best classifier for a particular corpus will have $E_h = 1$, while the other competing algorithms will have $E_h \leq 1$. Lower values of E_h indicate the lack of robustness of the algorithm *h*. We have computed this ratio for all the classifiers and for all the corpora using micro-averaged and macro-averaged *f*-measure, and they are graphically shown by box-plots, respectively, in Figs. 1 and 2. It can be observed from these figures that the proposed method outperforms the competing classifiers.

6 Conclusion

A method has been introduced in this article to overcome some of the limitations of the state-of-the-art nearest-neighbor decision rules for effective text

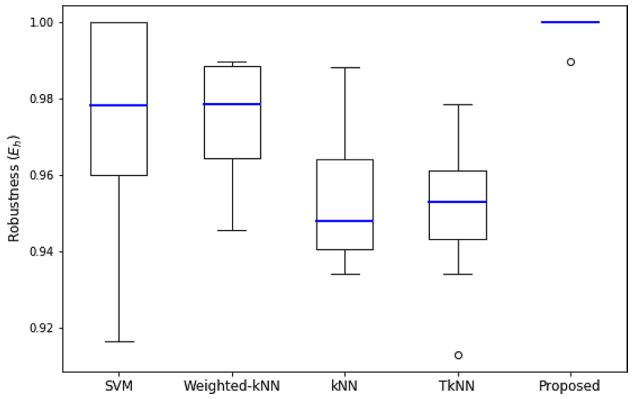

Fig. 1 Robustness of different classifiers using micro-averaged *F*-measure

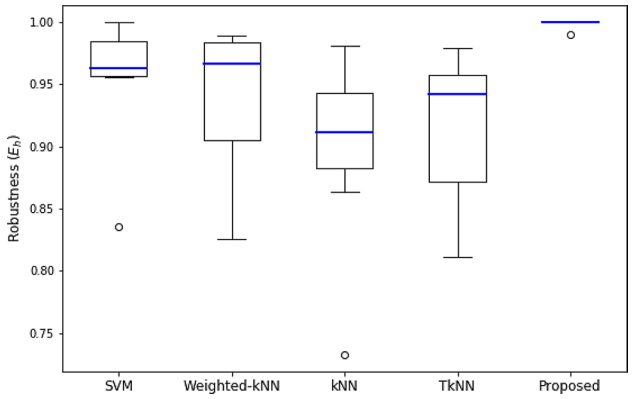

Fig. 2 Robustness of different classifiers using macro-averaged *F*-measure

categorization. The performance of the proposed method is evaluated on different standard benchmark corpora. The method uses a parameter γ to provide a bound on the difference between the weights of the competing categories. Note that for a high value of γ, many documents may remain unclassified and for a low value of γ, we may compromise with the quality of the decision making. Thus choice of γ is crucial. In the experiments, the value of γ is chosen using the cross-validation technique on the training set. The empirical analysis show that the proposed technique outperforms the state-of-the-art classifiers in most of the cases. It is also observed that no document remain unclassified by the proposed method for all corpora. This proves the effectiveness of the method. In future, the performance of the proposed method should be tested in different other applications, e.g., customer review analysis.

Acknowledgements The authors would like to thank the editors and reviewers for their valuable comments to improve the contents of the article.

Compliance with ethical standards

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Cover T, Hart P (1967) Nearest neighbor pattern classification. IEEE Trans Inf Theory 13(1):21–27
2. Gou J, Ma H, Ou W, Zeng S, Rao Y, Yang H (2019) A generalized mean distance-based k-nearest neighbor classifier. Expert Syst Appl 115:356–372
3. Zhang S, Li X, Zong M, Zhu X, Wang R (2017) Efficient kNN classification with different numbers of nearest neighbors. IEEE Trans Neural Netw Learn Syst 29(5):1774–1785
4. Duda R, Hart P (1973) Pattern classification and scene analysis. Wiley, New York
5. Basu T, Murthy CA, Chakraborty H (2012) A tweak on k-nearest neighbor decision rule. In: Proceedings of the international conference on image processing, computer vision, and pattern recognition, Las Vegas, USA, pp 929–935
6. Dudani SA (1976) The distance-weighted k-nearest-neighbor rule. IEEE Trans Syst Man Cybern 4:325–327
7. Struyf A et al (1997) Clustering in an object-oriented environment. J Stat Softw 1(4):1–30
8. Mukherjee A, Basu, T (2018) An effective nearest neighbor classification technique using medoid based weighting scheme. In: Proceedings of international conference on data science. CSREA Press, pp 231–234
9. Sebastiani F (2002) Machine learning in automated text categorization. ACM Comput Surv (CSUR) 34(1):1–47
10. Burges CJC (1998) A tutorial on support vector machines for pattern recognition. Data Min Knowl Discov 2(2):121–167
11. Dumais S, Platt J, Heckerman D, Sahami M (1998) Inductive learning algorithms and representations for text categorization. In: Proceedings of international conference on information and knowledge management (CIKM), Bethesda, USA, pp 148–155
12. Joachims T (1998) Text categorization with support vector machines: learning with many relevant features. In: Proceedings of the European conference on machine learning, ECML'98, Berlin, Germany, pp 137–142
13. Yang Y (1999) An evaluation of statistical approaches to text categorization. Information Retr 1(1–2):69–90
14. Du S, Li J (2019) Parallel processing of improved kNN text classification algorithm based on Hadoop. In: Proceedings of IEEE international conference on information, communication and networks, pp 167–170

15. Gowda K, Krishna G (1979) The condensed nearest neighbor rule using the concept of mutual nearest neighborhood (corresp.). IEEE Trans Inf Theory 25(4):488–490
16. Gates G (1972) The reduced nearest neighbor rule. IEEE Trans Inf Theory 18(3):431–433
17. Bagui SC, Bagui S, Pal K, Pal NR (2003) Breast cancer detection using rank nearest neighbor classification rules. Pattern Recognit 36(1):25–34
18. Guan J, Zhou S (2002) Pruning training corpus to speedup text classification. In: Proceedings of the international conference on database and expert systems applications. Springer, Berlin, pp 831–840
19. Tan S (2005) Neighbor-weighted k-nearest neighbor for unbalanced text corpus. Expert Syst Appl 28(4):667–671
20. Basu T, Murthy CA (2016) A supervised term selection technique for effective text categorization. Int J Mach Learn Cybern 7(5):877–892
21. Manning CD, Raghavan P, Schutze H (2008) Introduction to information retrieval. Cambridge University Press, New York
22. Salton G, McGill MJ (1983) Introduction to modern information retrieval. McGraw Hill, New York
23. Huang A (2008) Similarity measures for text document clustering. In: Proceedings of the New Zealand computer science research student conference, Christchurch, New Zealand, pp 49–56
24. Park HS, Jun CH (2009) A simple and fast algorithm for k-medoids clustering. Expert Syst Appl 36(2):3336–3341
25. Karypis G, Han ES (2000) Fast supervised dimensionality reduction algorithm with applications to document categorization and retrieval. In: Proceedings of the ninth international conference on Information and knowledge management. ACM, pp 12–19
26. Basu T, Murthy CA (2012) A feature selection method for improved document classification. In: Proceedings of international conference on advanced data mining and applications. Springer, Berlin, pp 296–305
27. Schutze H, Manning CD, Raghavan P (2008) Introduction to information retrieval, vol 39. Cambridge University Press, Cambridge
28. Basu T, Murthy CA (2014) Towards enriching the quality of k-nearest neighbor rule for document classification. Int J Mach Learn Cybern 5(6):897–905
29. Witten IH, Frank E, Hall MA, Pal CJ (2016) Data mining: practical machine learning tools and techniques. Morgan Kaufmann, Burlington
30. Pedregosa F et al (2011) Scikit-learn: machine learning in python. J Mach Learn Res 12(Oct):2825–2830
31. Ruxton GD (2006) The unequal variance t-test is an underused alternative to student's t-test and the Mann–Whitney u test. Behav Ecology 17(4):688–690
32. Rao CR, Mitra SK, Matthai A, Ramamurthy KG (eds) (1966) Formulae and tables for statistical work. Statistical Publishing Society, Calcutta
33. Friedman J (1994) Flexible metric nearest neighbor classification. Technical report, Stanford University