Fuzzy document classification using ontology based approach for term weighting

Aijazahamed Qazi
Assistant Professor, Department of Computer Science and Engineering
S.D.M College of Engineering and Technology
Dharwad-580002, Karnataka, India
aqsdmcet@gmail.com
https://orcid.org/0000-0002-0013-2877

Abstract*: *With the surge in web corpus, document classification is a vital issue in information retrieval. Term weighting increases the accuracy of classification for documents represented in the vector space model. This paper proposes an ontoTf-idf term weighting method based on the assessment of semantic similarity between the group label and the term. In this paper, a comparative analysis of the performance of the traditional Term Frequency-Inverse Document Frequency (Tf-idf) method and ontoTf-idf method is carried on the WebKB and Reuters-21578 benchmark datasets. The efficiency of ontoTf-idf method is validated with kNN (k nearest neighbor) and Fuzzy kNN classifier on the WebKB and Reuters-21578 datasets. The experimental results obtained with the proposed ontoTf-idf method outperform the Tf-idf method. In the proposed work, distance metrics like Euclidean distance, Cosine similarity, Manhattan distance, and Jaccard co-efficient are applied with Fuzzy kNN classifier on the WebKB and Reuters-21578 dataset.

(Article history: Received: 15th October 2021 and accepted 3rd December 2021)

I. INTRODUCTION

With the evolution of the web, searching relevant information from a large corpus is a challenging issue. Information retrieval plays a significant role to search for user's necessities within large collections. Term Frequency-Inverse Document Frequency does not consider semantic information related to the term. In document classification, a collection of documents D where $d_i \in D$ and a group of categories C where $c_j \in C$ are considered. The aim is to estimate the function $f:D\times C\rightarrow\{1,0\}$ and provide a Boolean value {1, 0} to each $< d_i, c_j >$. A value 1 is assigned to d_i, if d_i is relevant to c_j, and a value 0 if d_i is not relevant to c_j. Document classification is categorized into statistical and context-based. Statistical approaches are based on the word occurrence and context-based approaches assess the semantic significance of a given category. Words are hierarchically related by ontological networks, namely, WordNet and DBpedia which include synonyms, hyponyms, etc. The contribution of this research paper is to semantically categorize the documents using the term weighting method based on ontologies.

The paper is organized as follows; Section 2 reviews the literature. In Section 3, the preliminaries are discussed. Section 4 elaborates the proposed term weighting methodology. The experimental results are deliberated in Section 5. Section 6 concludes the paper.

II. LITERATURE REVIEW

The weighting of a term empowers information retrieval systems by increasing their effectiveness. Ibtihel et al. [1] proposed a semantic framework for improving the accuracy of text classification based on feature expansion from knowledge bases like WordNet. Kim et al. [2] presented a multi-co-training framework for document classification. The proposed approach represented documents using Tf-idf, Latent Dirichlet allocation, and Doc2Vec. Patel & Sharma [3] proposed a framework for the classification of the newsfeed. The process involved URL crawling, parsing of news content, weighting of keywords, and identifying a relevant category. Cheng et al. [4] presented an improved Tf-idf method that used relative entropy as the calculation factor to optimize the performance. Semantic similarity improves the accuracy of information retrieval systems [5]. Sebastiani [6] proposed an automated framework for categorization of text by term weighting. Luo et al. [7] proposed a weighting technique by considering the semantics of category labels and indexing of terms. Sabbah et al. [8] presented a hybrid term weighting method for web content classification using SVM. The proposed hybrid technique combined features obtained by Tf-idf, Glasgow, and Entropy into a single feature set for classification.

Information content (IC) based methods measure the similarity amongst the concepts by calculating the distribution of concepts in a given corpus. Hu et al. [9] discussed the analysis of the distance measures using kNN classifier for medical datasets. Abu Alfeilat et al. [10] investigated the performance using the kNN classifier with various distance measures, on clean and noisy data sets. Ryu et al. [11] discussed the Fuzzy kNN classifier with a case-based selection technique and indicated improvement in the performance. Biswas et al. [12] presented a parameter-independent Fuzzy kNN method to measure the feature weight. Liangxiao Jiang et al. [13] discussed the importance of class-specific attribute weighting for state-of-the-art naive Bayes text classifiers and achieved noteworthy advances.

III. PRELIMANARIES

A. Benchmark Datasets

The WebKB dataset represents the collection of hypertext-related information. It contains the web pages collected from computer science department of Washington, Cornell, Texas and Wisconsin universities by the CMU text learning group. Four classes i.e., student, faculty, course, and project are considered and experimentation is carried on 2,798 training and 1,259 test documents.

The Reuters-21578 (R8) dataset is a collection of text documents widely used for document classification. The documents were grouped by Reuters Ltd and ModApte train and test split is used in many research works. R8 dataset is formed by documents of 8 classes i.e., acq, earn, crude, gain, interest, money-fx, ship, and trade, having the maximum count of samples. The experimentation is carried on 4,172 training and 1,549 test documents.

Table 1. WebKB Dataset with four categories

Class	Total # docs
project	504
course	930
faculty	1124
student	1641

Table 2. Reuters-21578 Dataset with eight categories

Class	Total # docs
acq	2292
crude	374
earn	3932
grain	51
interest	271
money-fx	293
ship	144
trade	326

B. Classifiers

In …… k Nearest Neighbor (kNN) is a machine learning algorithm that calculates the proximity between the two points. The train and test samples are represented as feature vectors. The Euclidean distance amongst two vectors, $P = (p_1, p_2, \dots, p_i)$ *and* $Q = (q_1, q_2, \dots, q_i)$ is,

$$dist(P,Q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \qquad (5)$$

Fuzzy k Nearest Neighbor (Fuzzy kNN) clusters the vectors $V = \{v_1, v_2, \dots, v_n\}$ into p [1 < p < n] fuzzy subsets. The matrix of fuzzy membership is given by M, where M_{ij} is the fuzzy membership degree of v_j in class i where, i = 1,2,...,p and j = 1, 2,...,n.

The matrix M has two limitations as follows:

$$\sum_{i=1}^{\nu} M_{ij} = 1 \;\; M_{ij} \in [0,1] \qquad (6)$$

$$0 < \sum_{j=1}^{m} M_{ij} < n \,, M_{ij} \in [0,1] \qquad (7)$$

Equation (6) checks that all the objects across the classes have a degree of membership and the summation of all the degrees of membership equals one. Equation (7) specifies the fuzzy membership degree in the interval of [0, 1]. Fuzzy kNN provides membership degree to each of the vector based on the distance from its k nearest neighbors as follows,

$$M_i(V) = \sum_{j=1}^{k} \frac{\frac{M_{ij}}{\left\|v - v_j\right\|^{\left(\frac{2}{m-1}\right)}}}{\sum_{j=1}^{k} \frac{1}{\left\|v - v_j\right\|^{\left(\frac{2}{m-1}\right)}}} \qquad (8)$$

where, k refers to the total nearest neighbors and m is a constant parameter. Equation (8) computes the membership degree of an object to the i^{th} class.

C. Distance Measures

Here Distance measures assess the proximity between the two documents in the vector space. Euclidean Distance represents the distance between two points in a Euclidean space. Let A and B be represented by vectors $A = (a_1, a_2, \dots, a_m)$, $B = (b_1, b_2, \dots, b_m)$ and m be the dimensionality of the feature space.

$$d(A,B) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + \cdots + (a_m - b_m)^2} \qquad (9)$$

Cosine similarity refers to the degree of resemblance between the two documents with value between 0 and 1. The two documents are similar if the value obtained is equal to 1 and non-similar if it is 0.

$$Simi_{cos}(Docu_p, Docu_q) = \frac{\sum_{i=1}^{m}(Docu_{pi} * Docu_{qi})}{\sqrt{\sum_{i=1}^{m}(Docu_{pi})} * \sqrt{\sum_{i=1}^{m}(Docu_{qi})}} \qquad (10)$$

where, $Docu_{pi}$ is the i^{th} term in the document vector p, $Docu_{qi}$ is the i^{th} term in the document vector q and m is the total term count.

Jaccard Coefficient compares the resemblance between a query vector and a document vector.

$$Simi_{jaccard}(Docu_p, Docu_q) = \frac{\sum_{i=1}^{m}(Docu_{pi} * Docu_{qi})}{\sum_{i=1}^{m}(Docu_{pi}) + \sum_{i=1}^{m}(Docu_{qi}) - \sum_{i=1}^{m}(Docu_{pi} * Docu_{qi})} \qquad (11)$$

Manhattan Distance refers to the sum of the lengths of the projections of the line segment between the objects onto the coordinate axes.

$$d(A,B) = \sum_{i=1}^{m} |a_i - b_i|^2 \qquad (12)$$

D. Performance Measures

Precision represents the count of correctly classified positive samples divided by the total count of samples i.e. Precision (p) $= \frac{TP}{TP+FP}$. Recall represents the count of correctly classified positive samples divided by the total count of positive samples in the data i.e. Recall (r) $= \frac{TP}{TP+FN}$. F1 measure represents the harmonic mean of precision and recall i.e. F_1 measure $(F_1) = 2\frac{p*r}{p+r}$. Micro_averaging is obtained by the summation over all individual results, $Micro - F_1 = 2\frac{Precision_\mu * Recall_\mu}{Precision_\mu + Recall_\mu}$. Macro_averaging evaluates the precision and recall of each category, $Macro - F_1 = 2\frac{Precision_M * Recall_M}{Precision_M + Recall_M}$.

IV. Proposed Term Weighting Method: ontoTf-idf

Tf-idf reflects the importance of a term in a document. The Tf-idf is computed by,

$$Tf - idf_{i,j} = \frac{o_{i,j}}{\sum_k o_{k,j}} * log\frac{|DOCU|}{|\{doc_j : o_i \in doc_j\}|} \quad (1)$$

where, $o_{i,j}$ is the count of term occurrence in a document doc_j and $\sum_k o_{k,j}$ is the total occurrence of all the terms o_k in the document doc_j with $1 < k < 0$. $|DOCU|$ represents the document corpus and $|\{doc_j : o_k \in doc_j\}|$ is the set of documents with the term o_k and $o_{k,j} \neq 0$.

To measure the proposed term weighting method (ontoTf-idf), initially, each document doc_i of the collection $DOCU$ is subjected to removal of stop words and tokenization. The term frequency $term\_freq_i$ is computed for each term te_i in the token set TOK. Let $CATG$ represent the number of classes of the documents and $CLASS_L = \{cl_1, cl_2, \ldots, cl_{CATG}\}$ denotes the set of class labels of the documents. Let SYN_{cl_c} where $c = 1, 2, \ldots, CATG$, be the first synonym of cl^{th}_{CATG} category obtained from WordNet. Construct the set of synonym labels

$$SYN_{CLASS_L} = \{SYN_{cl_1}, SYN_{cl_2}, \ldots, SYN_{cl_c}\}$$

and Lin's similarity between the term te_i and synonym SYN_{cl_c} is calculated.

Calculate,

$$\beta eta_i \leftarrow max_{SYN_{cl_c} \in SYN_{CLASS_L}} \left(LIN_{SIM}\left(te_i, SYN_{cl_c}\right)\right) \quad (2)$$

where,

$LIN_{SIM}(m,n) \leftarrow 2\log(LCS(m,n))/(\log(P(m)) + \log(P(n)))$

with m, n representing the terms and LCS(m, n) represents the least common subsumer of m, n. The maximum βeta_i among the $CATG$ classes is found, which provides the semantic similarity between the terms.

Modified term frequency $mod\_term\_freq_i$ for each term te_i is calculated by considering Equation (1 and 2),

$$mod_{term_{freq_i}} = term_{freq_i} + \beta eta_i \quad (3)$$

By considering Equation (3), for each document $doc_j \in DOCU$ and for each term $te_i \in TOK$ of doc_j,

Calculate, $W_{ij} \leftarrow mod_{term_{freq_i}} * log\left(\frac{|DOCU|}{doc_{freq_i}}\right)$ (4)

where, |D| represents the total number of documents Equation (4) and doc_freq_i is the number of documents containing term te_i.

V. Experimental Analysis

A. Performance on the WebKB dataset

Removal of stop-words and stemming are the techniques applied to the documents before they are classified. Experimental analysis of an information retrieval model in the literature suggests that it is necessary to decrease the size of the feature space to improve the retrieval performance.

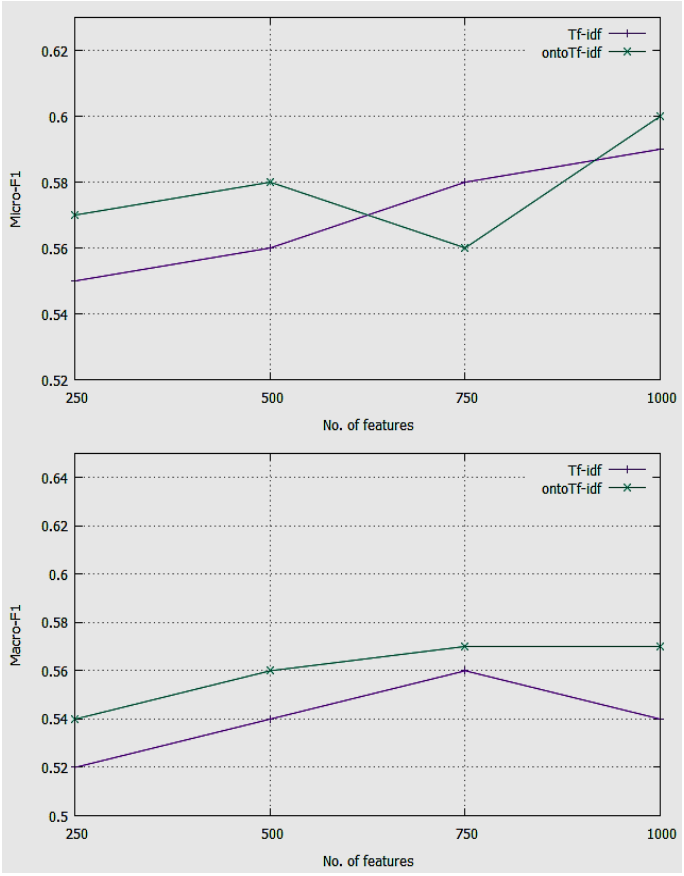

Figure 1. Micro-F_1 and Macro-F_1 values for the varying count of features with kNN (k=5) and WebKB dataset.

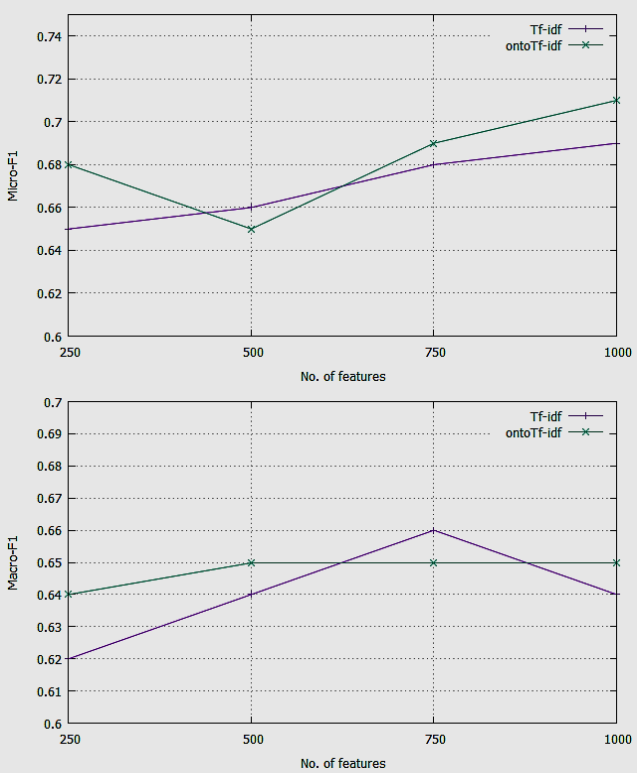

Figure 2. Micro-F_1 and Macro-F_1 values for the varying count of features with kNN (k=10) and WebKB dataset.

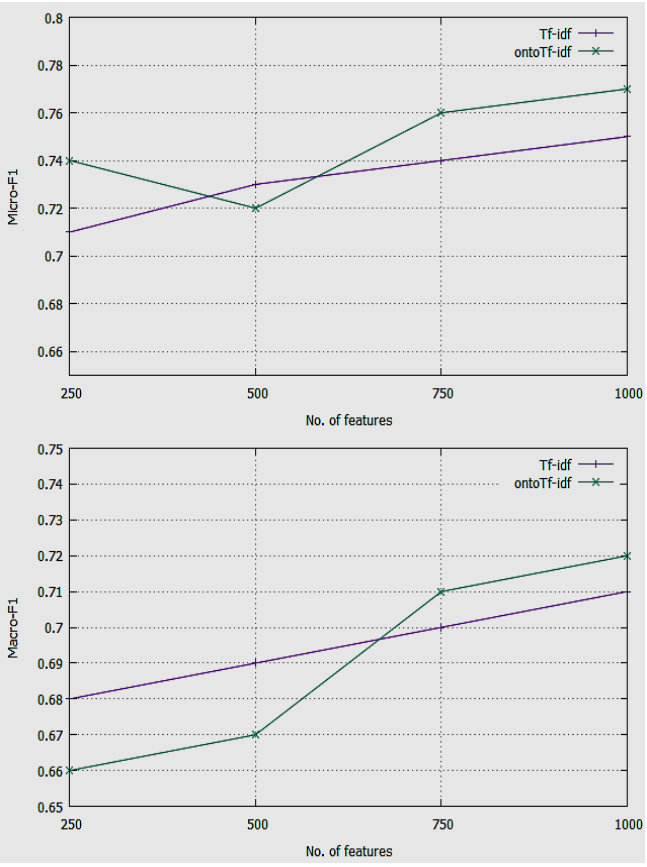

Figure 3. Micro-F_1 and Macro-F_1 values for the varying count of features with kNN (k=15) and WebKB dataset.

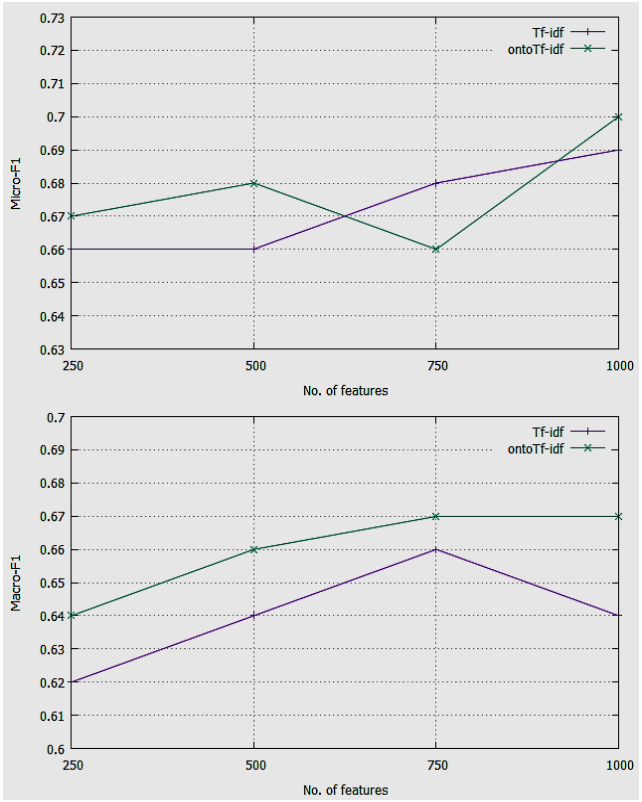

Figure 4. Micro-F_1 and Macro-F_1 values for the varying count of features with Fuzzy kNN (k=5) and WebKB dataset.

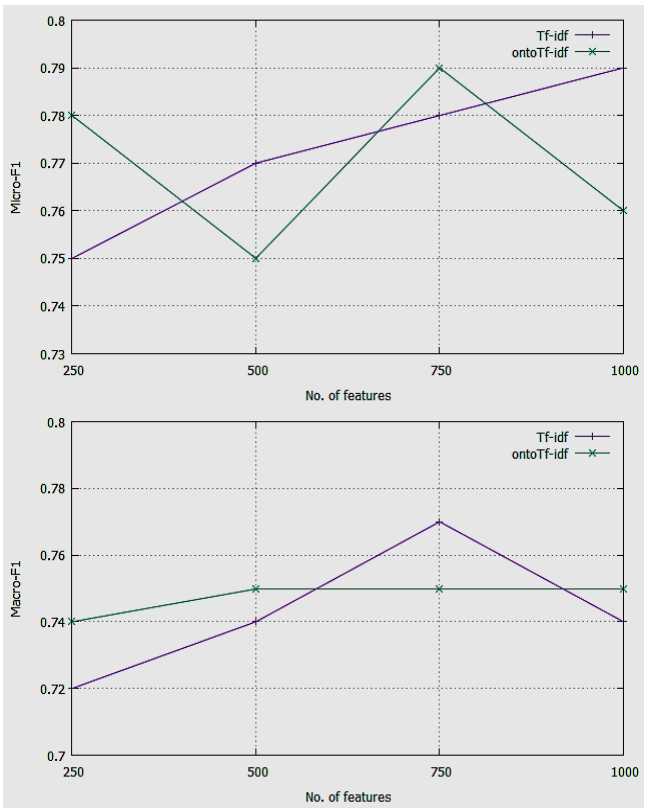

Figure 5. Micro-F_1 and Macro-F_1 values for the varying count of features with Fuzzy kNN (k=10) and WebKB dataset.

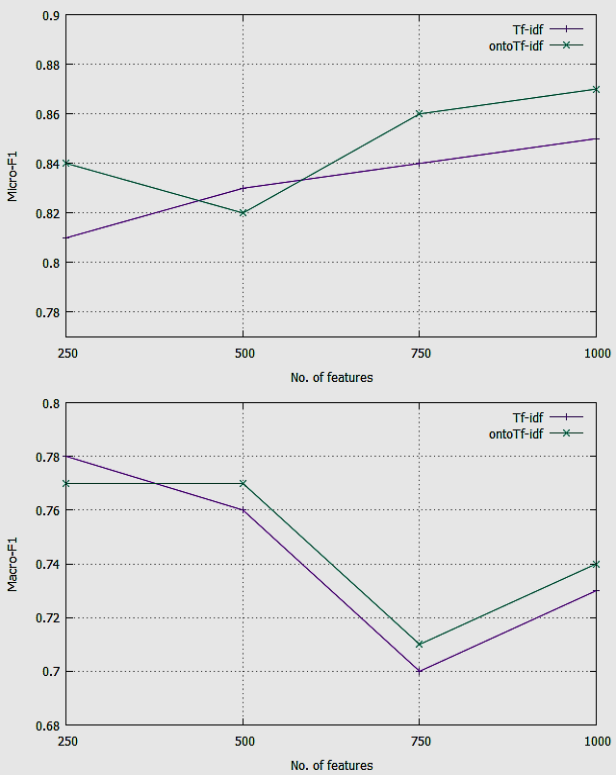

Figure 6. Micro-F_1 and Macro-F_1 values for varying count of features with Fuzzy kNN (k=15) and WebKB dataset.

In Figure 1, for Micro-F_1 values with 250-500 features, ontoTf-idf yields better results. For 500-750 features, Tf-idf yields better results and for 500-1000 features, ontoTf-idf yields better results. For 1000 features, Micro-F_1 value computed using the proposed ontoTf-idf method is increased by 1.6% as compared to Tf-idf method. For Macro-F_1 values with an increase in the number of features, ontoTf-idf outperforms Tf-idf by yielding significant results. For 1000 features, Macro-F_1 value computed using the proposed ontoTf-idf method is increased by 5.5% as compared to Tf-idf method. In Figure 2, for Micro-F_1 values with 250-500 features, Tf-idf yields better results. For 500-750 features, ontoTf-idf yields better results and for 500-1000 features, ontoTf-idf yields better results. For 1000 features, Micro-F_1 value computed using the proposed ontoTf-idf method is increased by 2.8% as compared to Tf-idf method. For Macro-F_1 values with 250-500 features, ontoTf-idf yields better results. For 500-750 features, Tf-idf yields better results and for 500-1000 features, ontoTf-idf yields better results. For 1000 features, Macro-F_1 value computed using the proposed ontoTf-idf method is increased by 1.5% as compared to Tf-idf method.

In Figure 3, for Micro-F_1 values with 250-500 features, Tf-idf yields better results. For 500-750 features, ontoTf-idf yields better results and for 500-1000 features, ontoTf-idf yields better results. For 1000 features, Micro-F_1 value computed using the proposed ontoTf-idf method is increased by 2.6% as compared to Tf-idf method. For Macro-F_1 values with 250-750 features, Tf-idf outperforms ontoTf-idf and for 750-1000 features, ontoTf-idf yields better results. For 1000 features, Macro-F_1 value computed using the proposed ontoTf-idf method is increased by 1.4% as compared to Tf-idf method. In Figure 4, for Micro-F_1 values, there is a rise in the ontoTf-idf curve till 250 features and then there is a fall till 750 features. For 1000 features, Micro-F_1 value computed using proposed ontoTf-idf method is increased by 1.4% as compared to Tf-idf method. For Macro-F_1 values, ontoTf-idf curve increases with an increase in feature size and performs significantly better than Tf-idf curve. For 1000 features, Macro-F_1 value computed using proposed ontoTf-idf method is increased by 4.6% as compared to Tf-idf method.

In Figure 5, for Micro-F_1 values the ontoTf-idf curve decreases with an increase in feature size, then the curve increases at 750 features. For 1000 features, Micro-F_1 value computed using proposed ontoTf-idf method is decreased by 4% as compared to Tf-idf method. For Macro-F_1 values the ontoTf-idf curve increases gradually till 500 features and remain constant with the increase in feature size. For 1000 features, Macro-F_1 value computed using the proposed ontoTf-idf method is increased by 1.3% as compared to Tf-idf method. In Figure 6, for Micro-F_1 values, the ontoTf-idf curve decreases to 500 features, and then there is a significant increase. For 1000 features, Micro-F_1 value computed using the proposed ontoTf-idf method is increased by 2.3% as compared to Tf-idf method. For Macro-F_1 values the ontoTf-idf curve remains constant till 500 features, then the curve decreases till 750 features followed by an increase. For 1000 features, Macro-F_1 value computed using proposed ontoTf-idf method is increased by 1.3% as compared to Tf-idf method respectively.

B. Performance on the Reuters-21578(R8) dataset

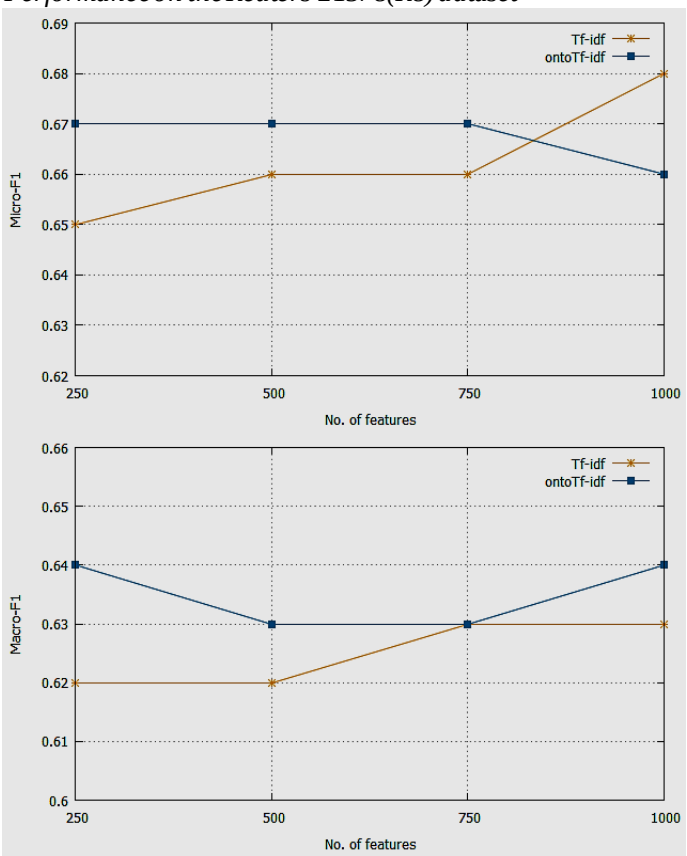

Figure 7. Micro-F_1 and Macro-F_1 values for the varying count of features with kNN (k=5) and Reuters (R8) dataset.

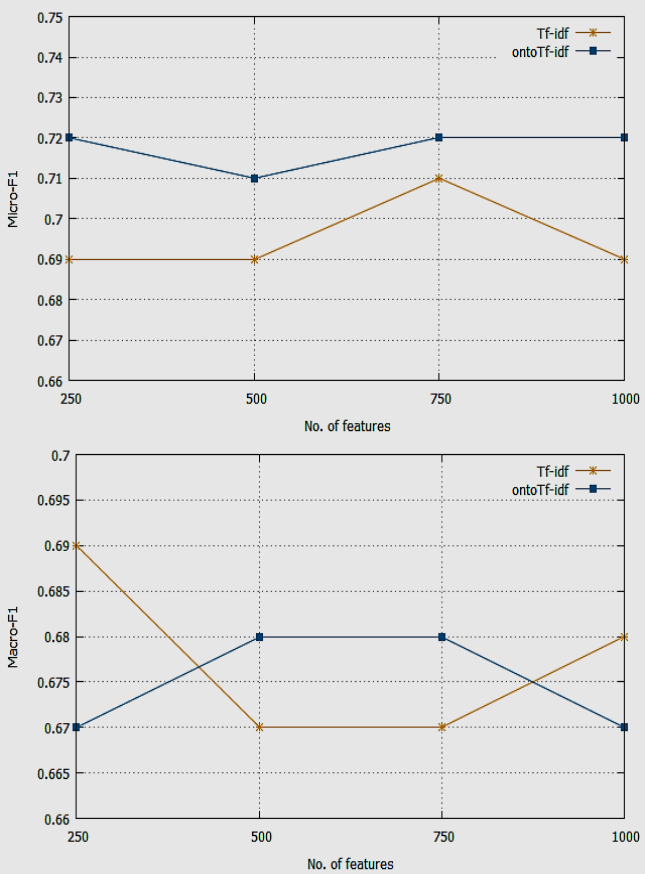

Figure 8. Micro-F_1 and Macro-F_1 values for the varying count of features with kNN (k=10) and Reuters (R8) dataset.

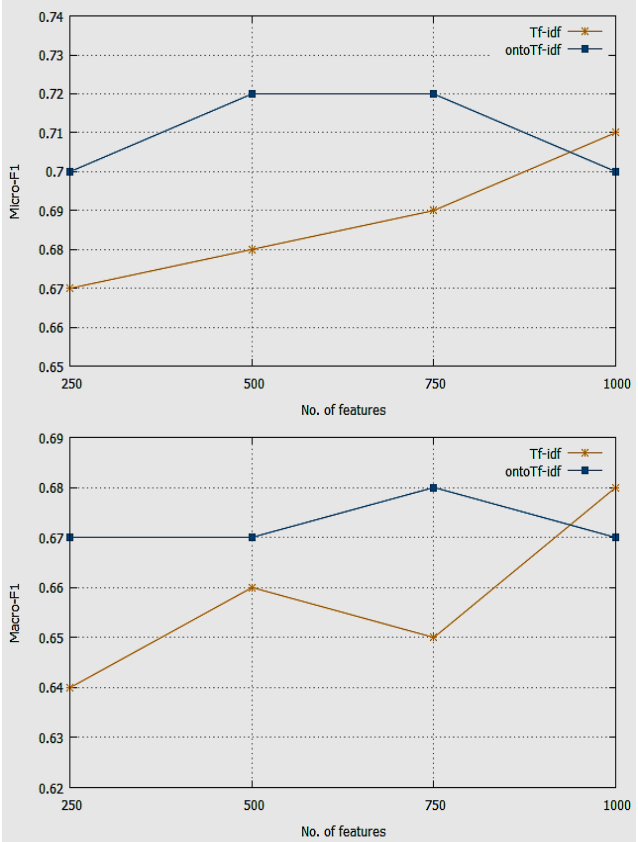

Figure 10. Micro-F_1 and Macro-F_1 values for the varying count of features with Fuzzy kNN (k=5) and Reuters (R8) dataset.

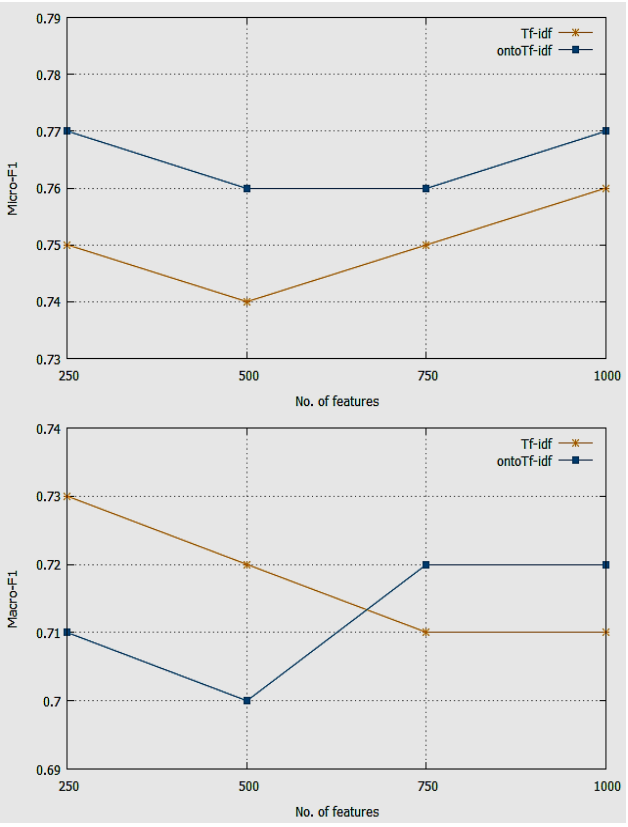

Figure 9. Micro-F_1 and Macro-F_1 values for the varying count of features with kNN (k=15) and Reuters (R8) dataset.

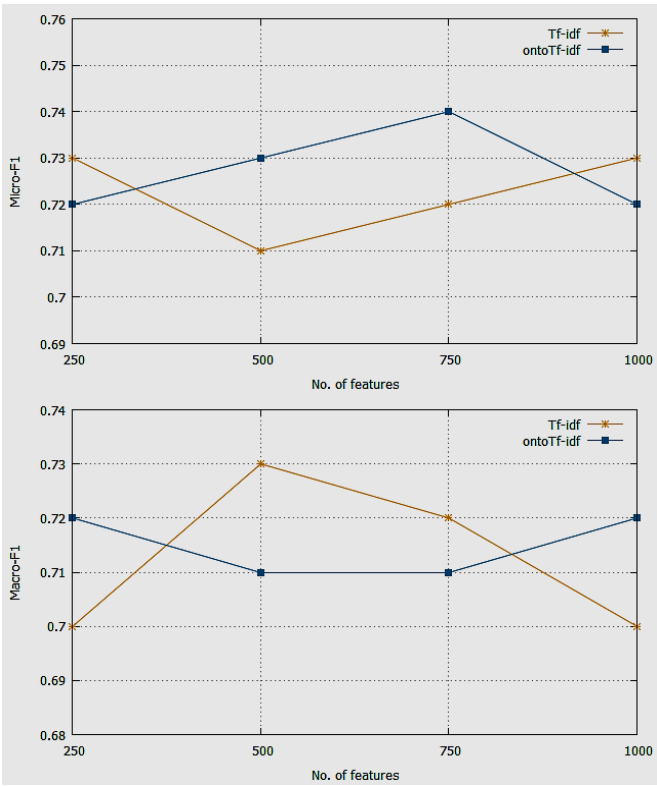

Figure 11. Micro-F_1 and Macro-F_1 values for the varying count of features with Fuzzy kNN (k=10) and Reuters (R8) dataset.

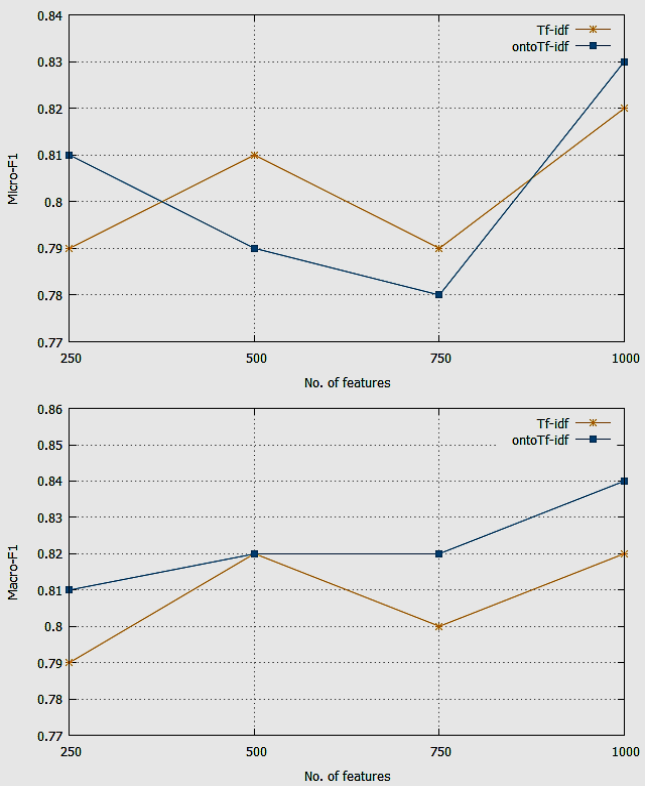

Figure 12. Micro-F_1 and Macro-F_1 values for the varying count of features with Fuzzy kNN (k=15) and Reuters (R8) dataset.

In Figure 7, for 750 features, Micro-F_1 value computed using the proposed ontoTf-idf method is increased by 1.5% and Macro-F_1 value remains the same as compared to Tf-idf method. In Figure 8 for 750 features, Micro-F_1 value computed using the proposed ontoTf-idf method is increased by 1.4% and Macro-F_1 value is increased by 1.5% as compared to Tf-idf method. In Figure 9, for 750 features, Micro-F_1 value computed using the proposed ontoTf-idf method is increased by 1.3% and Macro-F_1 value is increased by 1.4% as compared to Tf-idf method. In Figure 10, for 750 features, Micro-F_1 value computed using proposed ontoTf-idf method is increased by 4.3% and Macro-F_1 value is increased by 4.6% as compared to Tf-idf method. In Figure 11, for 750 features, Micro-F_1 value computed using proposed ontoTf-idf method is increased by 2.7% and Macro-F_1 value is decreased by 1.4% as compared to Tf-idf method. In Figure 12, for 750 features, Micro-F_1 value computed using the proposed ontoTf-idf method is increased by 2.5% and Macro-F_1 value is increased by 4% as compared to Tf-idf method.

Table 4. Classification accuracy on WebKB dataset

Classifier		Size	Macro-F1		Micro-F1	
			Tf-idf	ontoTf-idf	Tf-idf	ontoTf-idf
kNN	k=5	250	0.71	0.75	0.80	0.83
		500	0.73	0.76	0.82	0.81
		750	0.72	0.75	0.83	0.87
		1000	0.7	0.77	0.86	0.86
	k=10	250	0.77	0.77	0.8	0.83
		500	0.75	0.75	0.83	0.82
		750	0.70	0.76	0.8	0.85
		1000	0.73	0.74	0.85	0.87
	k=15	250	0.76	0.75	0.82	0.84
		500	0.76	0.75	0.81	0.82
		750	0.7	0.71	0.84	0.86
		1000	0.72	0.73	0.84	0.87
Fuzzy kNN	k=5	250	0.77	0.77	0.80	0.84
		500	0.76	0.77	0.83	0.83
		750	0.7	0.71	0.83	0.86
		1000	0.72	0.74	0.83	0.85
	k=10	250	0.73	0.76	0.81	0.84
		500	0.76	0.76	0.83	0.82
		750	0.7	0.72	0.84	0.86
		1000	0.73	0.73	0.85	0.87
	k=15	250	0.74	0.77	0.81	0.82
		500	0.76	0.76	0.81	0.83
		750	0.7	0.72	0.82	0.84
		1000	0.71	0.73	0.85	0.86

Table 4. Classification accuracy on Reuters-21578(R8) dataset

Classifier		Size	Macro-F1		Micro-F1	
			Tf-idf	ontoTf-idf	Tf-idf	ontoTf-idf
kNN	k=5	250	0.78	0.77	0.81	0.84
		500	0.76	0.77	0.83	0.82
		750	0.7	0.71	0.84	0.86
		1000	0.73	0.74	0.85	0.87
	k=10	250	0.78	0.77	0.81	0.84
		500	0.76	0.77	0.83	0.82
		750	0.7	0.71	0.84	0.86
		1000	0.73	0.74	0.85	0.87
	k=15	250	0.78	0.77	0.81	0.84
		500	0.76	0.77	0.83	0.82
		750	0.7	0.71	0.84	0.86
		1000	0.73	0.74	0.85	0.87
Fuzzy kNN	k=5	250	0.78	0.77	0.81	0.84
		500	0.76	0.77	0.83	0.82
		750	0.7	0.71	0.84	0.86
		1000	0.73	0.74	0.85	0.87
	k=10	250	0.78	0.77	0.81	0.84
		500	0.76	0.77	0.83	0.82
		750	0.7	0.71	0.84	0.86
		1000	0.73	0.74	0.85	0.87
	k=15	250	0.78	0.77	0.81	0.84
		500	0.76	0.77	0.83	0.82
		750	0.7	0.71	0.84	0.86
		1000	0.73	0.74	0.85	0.87

C. Performance of ontoTf-idf method using similarity measures and Fuzzy kNN classifier

The Figure 13 shows Micro-F_1 values for varying count of features computed by using the proposed ontoTf-idf method with Fuzzy kNN (k=15) classifier based on similarity measures for WebKB dataset. For 250 features, it is observed that the proposed ontoTf-idf method using Fuzzy kNN classifier and Jaccard coefficient yields better Micro-F_1 value as compared to the other similarity measures. For 250-500 features, the Jaccard coefficient yields better results, and for 500-750 features, Euclidean distance yields better results. For 750-1000 features, it is observed that the proposed ontoTf-idf method using Fuzzy kNN classifier and Cosine similarity yields better Macro-F_1 values as compared to the other similarity measures. In Figure 14, for 1000 features, it is observed that the proposed ontoTf-idf method using Fuzzy kNN classifier and Cosine similarity yields better Micro-F_1 and Macro-F_1 values as compared to the other similarity measures.

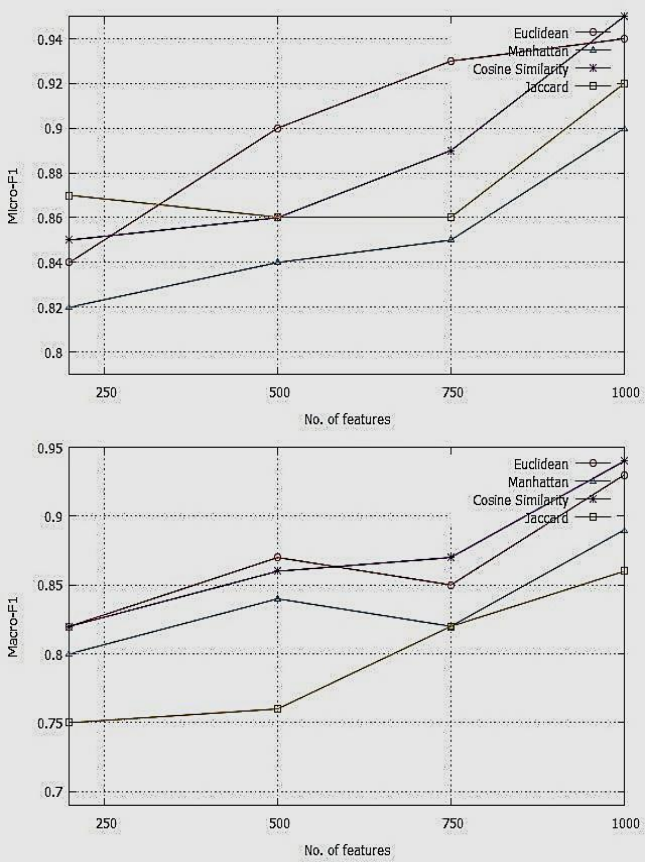

Figure 13. Micro-F_1 and Macro-F_1 values for the varying count of features with Fuzzy kNN (k=15) and WebKB dataset.

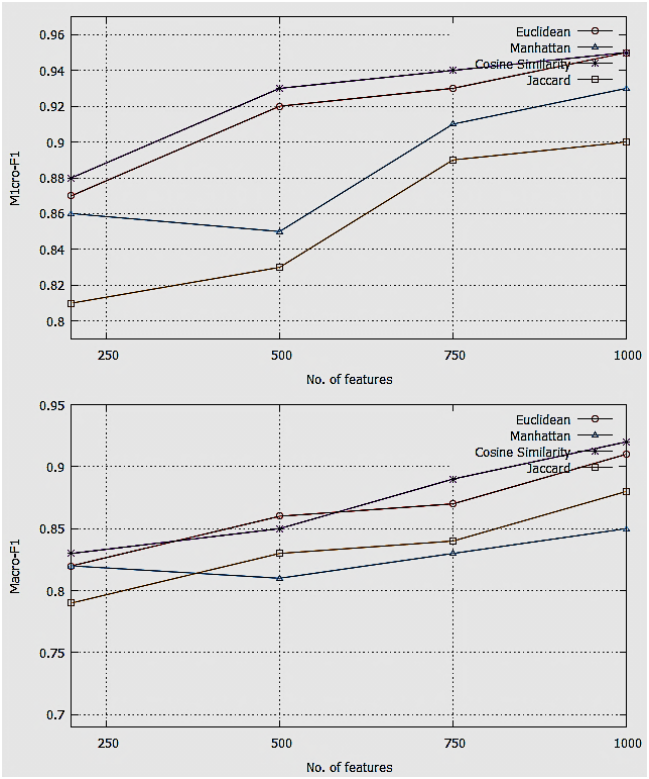

Figure 14. Macro-F_1 values for the varying count of features with Fuzzy kNN (k=15) based on similarity measures for Reuters (R8) dataset.

VI. CONCLUSION

In this paper, an experimental analysis of the proposed ontoTf-idf method with kNN and a Fuzzy kNN classifier is presented. It is observed that the proposed ontoTf-idf term weighting method using Fuzzy kNN classifier yields better Micro-F_1 and Macro-F_1 values. The comparison is done with the traditional Tf-idf method on the WebKB and Reuters-21578(R8) benchmark datasets. The experimental results of the proposed ontoTf-idf method using different similarity measures, namely, Euclidean distance, Manhattan distance, Cosine similarity and Jaccard coefficient with Fuzzy kNN classifier are presented. It is observed that the proposed ontoTf-idf method using Fuzzy kNN classifier and Cosine similarity yields better Micro-F_1 and Macro-F_1 values as compared to the other similarity measures on the benchmark datasets. The proposed ontoTf-idf method can be further improved to obtain better features for document classification. As a future work, the proposed method can be applied for multi-label classification using deep learning. Also, the proposed work can be modified for class-dependent term weighting to address the real-world text classification issues.

Acknowledgments

The author would like to thank the reviewers for their constructive comments.

Disclosure statement

No potential conflict of interest was reported by the author.

REFERENCES

[1] Ibtihel BL, Lobna H, Maher BJ (2018) A Semantic Approach for Tweet Categorization. Procedia Computer Science 126:335–344. https://doi.org/10.1016/j.procs.2018.07.267

[2] Kim D, Seo D, Cho S, Kang P (2019) Multi-co-training for document classification using various document representations: TF–IDF, LDA, and Doc2Vec. Information Sciences 477:15–29. https://doi.org/10.1016/j.ins.2018.10.006

[3] Patel AD, Sharma YK (2019) Web Page Classification on News Feeds Using Hybrid Technique for Extraction. In: Satapathy SC, Joshi A (eds) Information and Communication Technology for Intelligent Systems. Springer Singapore, Singapore, pp 399–405

[4] Cheng L, Yang Y, Zhao K, Gao Z (2020) Research and Improvement of TF-IDF Algorithm Based on Information Theory. In: Liu Q, Mısır M, Wang X, Liu W (eds) The 8th International Conference on Computer Engineering and Networks (CENet2018). Springer International Publishing, Cham, pp 608–616

[5] Uddin MN, Duong TH, Nguyen NT, et al (2013) Semantic similarity measures for enhancing information retrieval in folksonomies. Expert Systems with Applications 40:1645–1653.

[6] Sebastiani F (2002) Machine learning in automated text categorization. ACM Computing Surveys (CSUR) 34:1–47. https://doi.org/10.1145/505282.505283

[7] Luo Q, Chen E, Xiong H (2011) A semantic term weighting scheme for text categorization. Expert Systems with Applications 38:12708–12716. https://doi.org/10.1016/j.eswa.2011.04.058

[8] Sabbah T, Selamat A, Selamat MdH, et al (2016) Hybridized term-weighting method for Dark Web classification. Neurocomputing 173:1908–1926. https://doi.org/10.1016/j.neucom.2015.09.063

[9] Hu L-Y, Huang M-W, Ke S-W, Tsai C-F (2016) The distance function effect on k-nearest neighbor classification for medical datasets. SpringerPlus 5:. https://doi.org/10.1186/s40064-016-2941-7

[10] Abu Alfeilat HA, Hassanat ABA, Lasassmeh O, et al (2019) Effects of Distance Measure Choice on K-Nearest Neighbor Classifier Performance: A Review. Big Data 7:221–248. https://doi.org/10.1089/big.2018.0175

[11] Ryu D, Jang J-I, Baik J (2015) A Hybrid Instance Selection Using Nearest-Neighbor for Cross-Project Defect Prediction. Journal of Computer Science and Technology 30:969–980. https://doi.org/10.1007/s11390-015-1575-5

[12] Biswas N, Chakraborty S, Mullick SS, Das S (2018) A parameter independent fuzzy weighted k -Nearest neighbor classifier. Pattern Recognition Letters 101:80–87. https://doi.org/10.1016/j.patrec.2017.11.003

[13] Chen L, Jiang L, Li C (2021) Using modified term frequency to improve term weighting for text classification. Engineering Applications of Artificial Intelligence 101:104215. https://doi.org/10.1016/j.engappai.2021.104215

AUTHOR PROFILE

Dr. Aijazahamed Qazi is currently working as an Assistant Professor, Dept. of CSE, SDMCET, Dharwad. He has 10 years of teaching and research experience. He has published papers in indexed journals. His areas of interest are Artificial Intelligence and Semantic web.