Hindawi
Wireless Communications and Mobile Computing
Volume 2020, Article ID 8879054, 10 pages
https://doi.org/10.1155/2020/8879054

Research Article

An Embedded-Based Weighted Feature Selection Algorithm for Classifying Web Document

G. Siva Shankar,[1] **P. Ashokkumar**,[1] **R. Vinayakumar**,[2] **Uttam Ghosh**,[3] **Wathiq Mansoor**,[4] **and Waleed S. Alnumay**[5]

[1]*Department of Computer Science, Sri Ramachandra Institute of Higher Education and Research, Chennai 600116, India*
[2]*Division of Biomedical Informatics, Cincinnati Children's Hospital Medical Center, Cincinnati, OH, USA*
[3]*Vanderbilt University, USA*
[4]*Computer Engineering, University of Dubai, Dubai, UAE*
[5]*King Saud University, Riyadh, Saudi Arabia*

Correspondence should be addressed to P. Ashokkumar; ashok05002@gmail.com and Wathiq Mansoor; wmansoor@ud.ac.ae

Received 8 June 2020; Revised 30 July 2020; Accepted 17 August 2020; Published 15 September 2020

Academic Editor: Xiaojie Wang

With the exponential increase in a number of web pages daily, it makes it very difficult for a search engine to list relevant web pages. In this paper, we propose a machine learning-based classification model that can learn the best features in each web page and helps in search engine listing. The existing methods for listing have lots of drawbacks like interfacing the normal operations of the website and crawling lots of useless information. Our proposed algorithm provides an optimal classification for websites which has a large number of web pages such as Wikipedia by just considering core information like link text, side information, and header text. We implemented our algorithm with standard benchmark datasets, and the results show that our algorithm outperforms the existing algorithms.

1. Introduction

With the rapid growth of the number of web pages every day, it makes a web crawler difficult to read and organize the web pages. This problem makes the web classification process to get more important day by day. Web classification algorithms have many uses across various domains which include spam detection, web searching, document organizing, and cybersecurity [1]. In this paper, we are targeting web searching and aim at optimizing the search engine for providing quick and efficient search results to the users.

The web classification process can be done as follows: firstly, fix the number of classes and the properties of each class; secondly, given a set of training documents, the goal is to find the probability of all the classes that each document can fall in; lastly, the classification chooses the class which has the highest probability. There are lots of machine learning algorithms for the web classification process such as Support Vector Machine (SVM), Naive Bayes, and k-Nearest Neighbours (kNN) [2]. However, most of the machine learning models fail to produce desirable accuracy because there exist lots of features in a single web page. Hence, a good feature selection algorithm is combined with the machine learning model to increase the accuracy of the classification process. The goal of a feature selection algorithm is to get rid of most of the irrelevant features in the web page so that the input size that is fed to the machine learning model is reduced. As the input size is minimum, the machine learning model gets easier to learn the corelation between various features and performs better in terms of accuracy.

The main problem for the web crawler is to decide on what features (or words) to pick to simplify the web classification process and to achieve good accuracy. Most of the feature selection model falls under three categories, namely, Wrapper, Filter, and Embedded. Wrapper-based methods iteratively calculate the subset of features until an optimal subset is found that has the maximum performance. It starts with a zero-sized subset and iteratively adds/removes the features

and finds the accuracy. It recommends the best feature subset that yields the maximum accuracy. Sometimes, the wrapper methods work backward also. Filter-based methods try to rank each feature based on a few metrics such as Pearson correlation and Analysis of Variance. Then, it recommends top N features. Embedded methods are the hybrid of the other two methods. The feature selection that is going to be used in this paper is going to be the Embedded-based one.

Almost all of the existing works rely on term frequencies. If a term is present across all documents of the same class and is absent across all documents of other classes, then that particular term has a high weight in representing the class of the document because it is unique. The aim of this study is to not fully rely on term frequencies alone, instead considering other parts of the document which has the ability to represent the document class more accurately than the term frequencies.

Embedded-based feature selection has lots of difficulties for picking the required feature among the web pages for two main reasons; firstly, each web page has a different template; secondly, the number of web documents increases exponentially day by day. To solve the two problems, the proposed method extracts few information on the web page such as headings, link texts, and side information, which includes meta tags. A higher rank is been given to this extracted information, and the rest of the texts are given a lower rank based on Pearson correlation. Later, these features are fed into the machine learning model and got satisfying results while compared to existing methods.

1.1. Contribution. The main contributions of this paper are as follows:

(i) To increase the speed and accuracy of the web classification algorithm by reducing the dimensions of the document matrix

(ii) Side information such as meta tags, heading tags, table captions, and image descriptions is considered for classification purposes

(iii) The relationship between two web documents is calculated by the link between them. Using these relationships, a new way of classification is proposed

The rest of the paper is as follows, Section 2 describes the literature survey done by researchers on the field of link classification. Section 3 explains the working of our proposed methodology, and Section 4 compares our work with standard existing algorithms. Section 5 contains the discussions. The conclusion and future work are then proceeded by Section 6.

2. Related Works

In this subsection, we present a brief literature review on the recent research based on link-based web classification algorithms. The main goal of these algorithms is to use a clustering algorithm that relies on distance calculation steps to minimize the average distance among the web pages belonging to the same class and maximize the average distance among the web pages belonging to different classes. [3] explains the use of the tokenization of a web page to extract features; the class for a web page is then assigned based on the calculating distance between the features. Some researchers use machine learning algorithms such as support vector machines for classifying the web pages which extracts a large number of web links [4].

Few works like [5] calculate the fitness value on each iteration for classifying the documents. The fitness function gets more accurate at each successive iteration, and the iteration stops when there is no future increase in the fitness function. There are many works in the area of swarm optimization which is a nature-inspired algorithm where each member from a group of birds searches for food in different places and finally converges when food is found. The same algorithm is used to classify the web documents by searching for links across the documents in the group of web pages, and finally, all the partial solutions are merged to form a super optimal solution [6, 7].

The majority of the works in this area are based on the bag of words model [8]. The links in the document are converted into vectors, and then few algorithms like Term Frequency-Inverse Document Frequency (TF-IDF) are applied to extract the more frequent words present across the web pages [9]. These frequent words are used as features to classify the web pages [10, 11]. TF-IDF can only be efficient as long as each word in the document matrix is independent of each other, in case two or more words are synonym of each other, then it becomes difficult to represent the exact relationship; the research work [12] proposes a method which calculates the correlation between each pair of words to detect the synonym.

[13] proposes a kNN-based classification algorithm that uses the TF-IDF metric for classifying the Lao text; the uses of their research has wide use in Natural Language Processing (NLP). The input size used in their research has 7 attributes. The feature selection method they used is principal component analysis. The principal component analysis is an algorithm that can map an N feature vector space to a K feature vector space, where $k¡N$. Only 3 features are considered after the PCA feature selection method, and the accuracy of their proposed work is 71.4%.

[14] optimizes the support vector machine to be used in multiclass classification. SVM is good when there is only two target class, that is binary classification, but when there are more than 3 classes, then the computational cost of training is increased. In their research, they proposed a hierarchical classification model in which the SVM is optimized for a multiclass classification process. They compared their work with decision tree classifiers and kNN classifiers and produced better performance. Various optimization [15, 16] was done in the SVM classifier to produce better results for multiclass classification.

In recent years, the amount of data generated is huge and very complex [17], for example, lots of time series data are generated [18]. To tackle this issue, deep learning technology [18] is used for feature extraction. The deep learning approach performs better than the other existing machine learning algorithms [19].

There is one more problem that text feature selection can face, that is redundancy. Two or more different features having the same meaning are called redundant (sometimes called a synonym problem). [12] focuses on solving this

Table 1: The usage of machine learning algorithms for document classification—a summary.

Reference	Method	Summary
[30]	DragPushing	(i) Proposes kNN optimization which automatically balances the data points evenly across all the classes to avoid model misfits. (ii) They have set the value of k to 7 for their experiment. (iii) Three datasets are used Reuter-21578, industry sector, and TDT-5.
[31]	Prototype selection	(i) Much faster than [30]. (ii) Prototype selection recommends the most portable prototypes for training purposes. (iii) [31] eliminates most of the data points in training to increase speed.
[32]	ForesTexter	(i) The Gini index can easily predict the skewness in the majority class and creates many subtrees to balance the data points. (ii) Can work faster than [30, 31]. (iii) [32] combines both feature subspace selection and splitting criterion to create multiple subtrees to balance the data.
[33]	Resampling	(i) Handles the imbalance problem better than [32] by performing resampling. (ii) Instance weighting enables one to assign few weights for the imbalanced class so that the end performance (in terms of accuracy) is balanced. (iii) They validated the proposed method with SVM classifier.
[34].	Topic modelling	(i) Instead of balancing the data points across all the classes, this method uses the topic model to construct new data points in each class to create a complete dataset. (ii) This method considers more data points than [30, 32, 33] because of topic modeling which can construct new data points.
[35]	Bag of concepts	(i) Aims to reduce the dimensions of the document matrix representation. (ii) Instead of recommending data points from the data set (such as [31]), the bag of concepts groups one or more data points into topics. (iii) Bag of concepts solves many problems in the traditional bag of words models such as high dimensionality and sparsity issues.
[36]	Ontology-based deep learning model	(i) Enhances the problem of [35] by not considering the relationships among the documents. (ii) The features and their relationships are extracted based on deep learning. (iii) The ontology enhancement proposed in [36] helps to reduce the high dimensions. (iv) This method consumes more time in training the samples.
Proposed	Weighted feature selection	(i) Resolves the imbalance problem by assigning weights to the most important features. (ii) Three classifiers are used, namely, kNN, SVM, and Naïve Bayes. (iii) [35] fails to detect the relationships among the documents. In this paper, the proposed system detects the relationship and considers it for classification.

redundancy by considering semantic relationships along with corelation. Table 1 compares few existing works with the proposed feature selection model. Recently, many classification tasks in documents, such as [20], are done by deep learning. [21] recommends the use of three hidden layers with 1024 neurons yields in good accuracy; they also found that if the proper preprocessing (such as synonym elimination) is been applied, then the F1-Score is increased by 5 points. Different research papers use different methods for doing preprocessing; in the research work [22], each word is mapped to its concept vector which is a set of semantic words. In this paper, the proposed method uses the concept of link words and High-End Features to achieve dimension reduction. Furthermore, few researches [23, 24] focus more on finding out the relationship among various parts of the document towards the document class and recommended which parts to consider to increase the accuracy of the classification process.

3. Embedded-Based Feature Selection Method

The proposed method makes use of both Wrapper (selection of headers, link text, and side information) and Filter (correlation). The proposed method first assigns a few feature weights to the selected features and then assigns a few class weights to each document. Then, the original feature selection is implemented for selecting top N features for the web classification. The feature and class weights are calculated based on two metrics

(1) High end features (HEF)

(2) Weightage based on Links

After weights are assigned, the filter-based correlation feature selection is implemented to rank the features. Figure 1 shows the overall model of the proposed algorithm; the input documents are fed into the feature selection algorithm. There are two phases, the first phase assigns few weights to selective features, and the second phase computes the correlation for all the features. After the two phases are completed, the ranking step recommends top N features to the classifier.

3.1. High End Features (HEF). Since each web page may have different templates, it is difficult to choose which context has a high correlation between the target class. A HEF is a text that

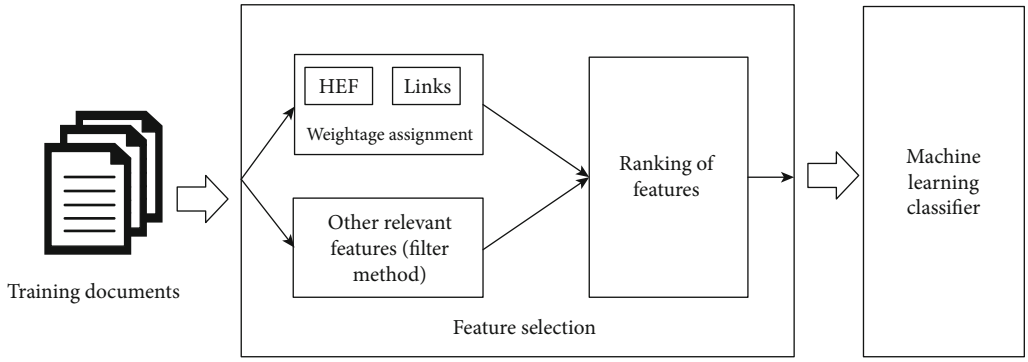


Figure 1: The model of the proposed algorithm.

determines a high correlation between the document and the target class. In this step, the proposed algorithm considers the texts present in headers, meta tags, and link text as HEF because they provide more information towards the determination of the target class while compared to other texts present in other areas. These HEF are given additional weightage for calculation of ranks; equation (1) describes the weightage process. The weightage W vector = $\{W1, W2, \cdots, Wn\}$, W is an n-dimensional vector (n denotes the number of unique words in the webpage) consisting of weights for each word.

$$\text{Weightage } W = \begin{cases} 1, \text{if word is not in HEF,} \\ 1.5, \text{if word is in HEF.} \end{cases} \quad (1)$$

3.2. Link Texts. The second metric that is considered is the link between the documents. Web documents are easy to classify while compared to normal text documents because of the links. A link between the two documents indicates a high correlation between the two documents. In this step, the proposed method selects only a few documents and run the machine learning model to predict the class. Then, the algorithm assigns the identified class of the selected documents to the rest of the documents based on the degree in which they are connected by links. The selected documents are called as leaders. The leaders are calculated by Link Inbound Outbound based classification algorithm (LIOBC). The algorithm is given at Algorithm 1.

3.3. LIOBC: Link Inbound Outbound Based Classification Algorithm. The algorithm first constructs a graph-based model from the collection of web pages, where each web page is represented as a node in the graph, and a link from two nodes is formed only when there is a link between them. A bag of the model is represented using the links available in a graph and then the Algorithm 1 (graph creation algorithm) connects the nodes (web pages) based on the hyperlinks. Thus, a web page having $n1$ hyperlinks will have $n1$ outbound and a web page that has $n2$ referring from other web documents will have $n2$ inbound.

There are lots of preprocessing steps that need to be done before executing the LIOBC algorithm such as detecting fake URLs [25], removing useless information like advertisements, images [26], and reducing the dimensionality of the contents [27, 28].

```
1:  procedure GraphGeneration
2:      N ⟵ List of web pages
3:  begin:
4:      for each document i in N do
5:          for each link l in i do
6:              J = Target of l
7:              create link between J and i
```

Algorithm 1: Graph generation.

Once the graph is generated, then it becomes easy to group related documents into separate distinct classes. A single document can belong to more than one class based on its similarity. The output of Algorithm 1 will create all links between web pages as shown in Figure 2(a). Two webpages can have more than one in links or out links or both. After creating links between web pages, the next step is to identify the leaders. A leader is a node that has several inbound than a specific fixed threshold TH1. Each leader is then given as an input to the machine learning model that assigns classes to leaders.

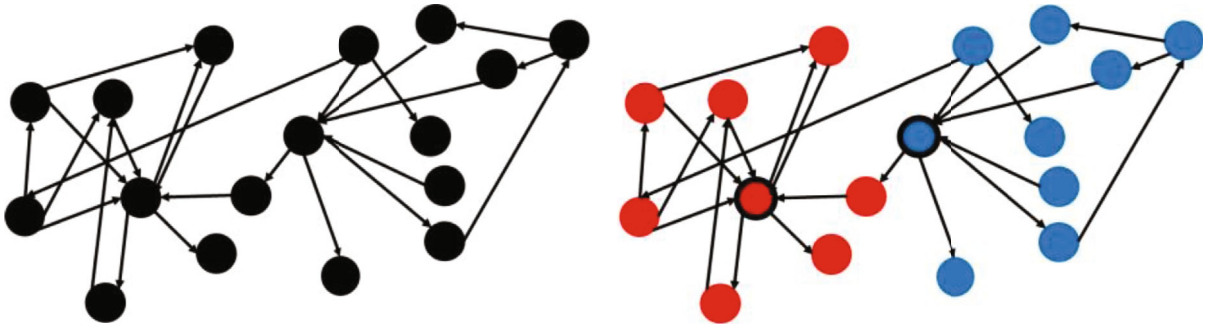
Leader (definition): a webpage that has the number of links higher than TH1. This signifies the high coupling. This coupling information can be used to increase the accuracy of the classifier.

After the classes of the leaders are found, it is very easy for assigning classes for the nonleader nodes. The assignment is as follows:

(i) A nonleader node which is having several outbound to only one leader greater than a prefixed threshold TH2, then the class of leader is fully assigned to the class of nonleader

(ii) A nonleader node which is having some outbound to more than one leader greater than a prefixed threshold TH2, then the classes of all leaders are equally assigned to the nonleader

(iii) If a nonleader node is not having enough outbound (number of outbound is less than TH2), then the machine learning model is used to run on nonleader independently considering it as a leader



(a) (b)

Figure 2: The class assignment of a document from leaders. (a) shows the output of Algorithm 1, and (b) shows the class assignment from leaders (leaders are bordered with dark black).

```
1:  procedure FeatureSelection
2:    BOW ⟵ Bag of Word representation of the corpus
3:    L ⟵ Set of Leaders from Algorithm 1
4:    G ⟵ The graph model from Algorithm 1
5:    begin:
6:    W={}
7:    for each Word w in BOW do
8:      if w thenord ∈ HEF such as link text, headings, meta informations, image descriptions etc...
9:        W[w]=BOW[w]∗1.5
10:     else
11:       W=[w]=BOW[w]
12:   DocClass={} - Represent the Document-Class vector, all documents are initialized 1 to all classes.
13:   for each leader l in L do
14:     class,prob=get class and probability of l from the classifer
15:     update DocClass, value=l, class and prob, weight =1.5
16:     for each neighbor n of l in G do
17:     update DocClass, value=n, class and prob, weight =1.5
18:   for each document d in the corpus do
19:   class,prob=get class and probability of d from the classifer
20:   update DocClass, value=l, class and prob, weight =1
21:   Assign the class which is having the highest prob∗weight value to d.
```

Algorithm 2: Embedded-based weighted feature selection algorithm.

The values of TH1 and TH2 can be dynamically fixed to get better accuracy. At the end of this step, each document will have a partial association (weightage) with a class by its leaders. Equation (2) represents this partial weightage association.

$$\text{Document Class Vector (DCV)} = \begin{cases} 1, \text{ for nonassociated class,} \\ 1.5, \text{ for associated class} \\ \text{(from leaders).} \end{cases} \tag{2}$$

3.4. The Ranking Method. Once the feature weights and class weights are assigned, each feature on the web page is ranked based on the Pearson Correlation. The final weight for each feature is calculated using equation (3). Then, top N features are recommended to the machine learning model. If the features are having the same final weight, then the feature with a high Pearson Correlation is been recommended. If the Pearson Correlation is also the same, then the features are recommended stochastically.

Table 2: Dataset description.

Dataset name	Number of documents
ClueWeb12	25 K
DBpedia	25 K

$$\text{Final Weight} = \text{Pearson Correlation} * W. \tag{3}$$

3.5. Web Page Classification. The machine learning models such as Naive Bayes, SVM, and kNN are implemented based on the top N final weight features, and each documents final class is calculated by equation (4). The class which is having

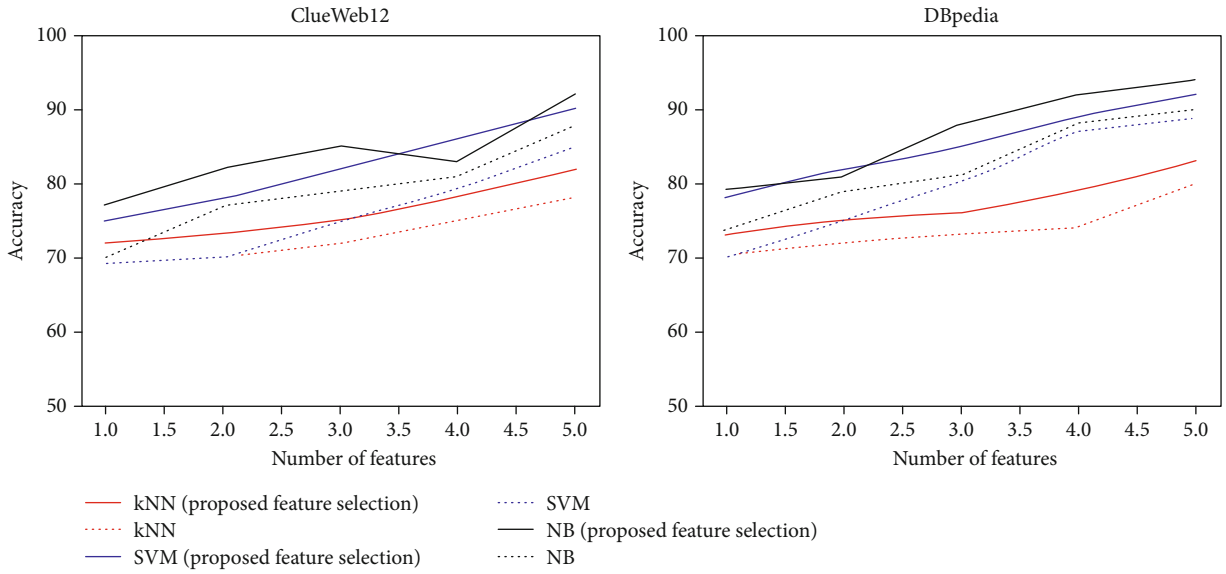


Figure 3: The accuracy of all the three classifiers when the number of classes = 5.

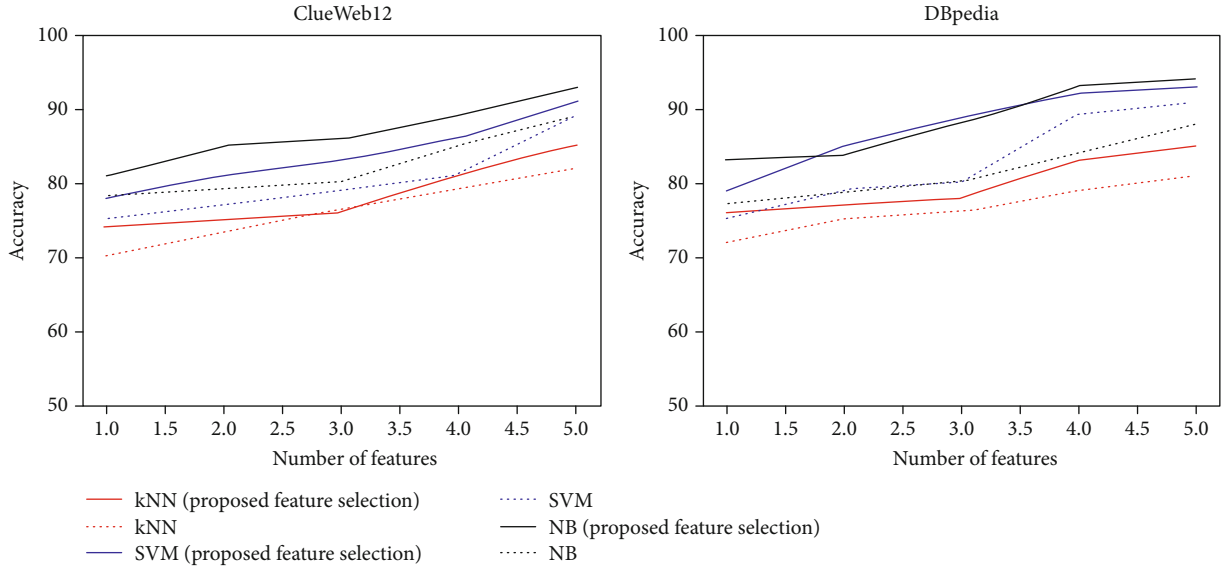


Figure 4: The accuracy of all the three classifiers when the number of classes = 7.

the highest probability is assigned to the web page. Algorithm 2 shows the overall working of the proposed model.

$$\text{Class Weight} = \text{Predicted} * \text{DCV}. \tag{4}$$

4. Experiment Result and Analysis

We have implemented our proposed algorithm on two standard datasets which are shown in Table 2. We took random 25 K documents from the two standard datasets and ran the experiment with three classifiers kNN, SVM, and Naive Bayes (NB). We have used Python 3.7 for implementing our experiment.

4.1. Preprocessing Stage. The datasets are preprocessed as follows: first, all the characters are converted into the lower case; second, the digits, punctuation marks, and stop words are filtered out; third, the stemming operation is performed and finally all the words are transformed into normalized words by adjusting singular, plural.

4.2. Performance Analysis. The efficiency of the classification model can be calculated using the accuracy metric. The calculation of accuracy is as per equation (5).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}. \tag{5}$$

TP, TN, FP, and FN are first calculated for each feature in the dataset and then finally cumulated. The definitions for TP, TN, FP, and FN are as follows (for each feature)

(i) True Positive (TP) - determines how many documents classified as the positive class which contains the feature

(ii) True Negative (TN) - determines how many documents classified as the negative class which does not contain the feature

(iii) False Positive (FP) - determines how many documents classified as the negative class which contains the feature

(iv) False Negative (FN) - determines how many documents classified as the positive class which does not contain the feature

The better the accuracy the more efficient the classification model is. The comparison in Figure 3 (number of class = 5) and Figure 4 (number of class = 7) shows that our algorithm has better stats because the existing algorithm (without the proposed feature selection algorithm) uses the full documents which often leads to high noise. Weighted words are better comparison parameters than ordinary text because the weighted words tell more about the content of the document than normal text do. We have tested all algorithms by having several classes 5 and 7. The class topic name is listed in Tables 3 and 4, respectively. Figure 5 shows how many link texts are recommended in the proposed method. If the proposed feature selection technique is not used, then the classifier is given the whole text corpus features as an input to perform the classification process.

Classification accuracy can be measured using another parameter called purity, which is defined as a fraction of documents classified to the correct class. The higher value of purity will yield better classification results. Purity is found as per equation (6).

$$\text{Purity} = \frac{\sum_{i=1}^{N} D_i}{\sum_{i=1}^{N} N_i}. \quad (6)$$

Purity measure is calculated for all the three algorithms as per equation (6), where N denotes the total number of classes, D_i denotes the documents classified to the class i, N_i denotes the total number of documents belonging to the class i. Graphs in Figures 6 and 7 show the result of purity measure when the number of classes is 5 and 7, respectively.

5. Discussion

Web documents are having an advantage over normal text documents regarding classification that is it provides lots of additional information regarding the correlation towards the topic. This additional information includes links (which represent a strong relationship of correlation between the two documents), headings (the header tags represent the heading or topic of the content), and other side information which includes meta tags, title tags, and description tags.

While most of the research work carried out on the field of feature selection focus on ranking the features based on the number of times it appears on the document, the proposed method makes use of the abovementioned additional information and gives extra weightage to them. Thus, prioritizing this additional information over normal text helps to reduce the noise and improve the accuracy of the classifier, for example, let us take a web document [29] as shown in Figure 8. The additional information like link text is circled, while the normal text is rectangular bordered. From the image, it is clear that the additional information words like

Table 3: Topic heads (For #classes = 5).

#	Topic head name
1	People and animals
2	Tourism
3	Health
4	Entertainments
5	Organizations

Table 4: Topic heads (For #classes = 7).

#	Topic head name
1	Culture
2	Animals
3	Tourism
4	Health
5	Games
6	Films
7	Organizations

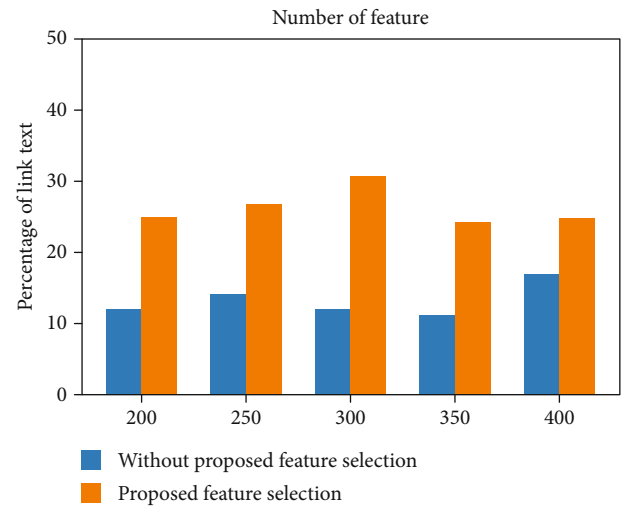


Figure 5: The number of link text recommended in top N features.

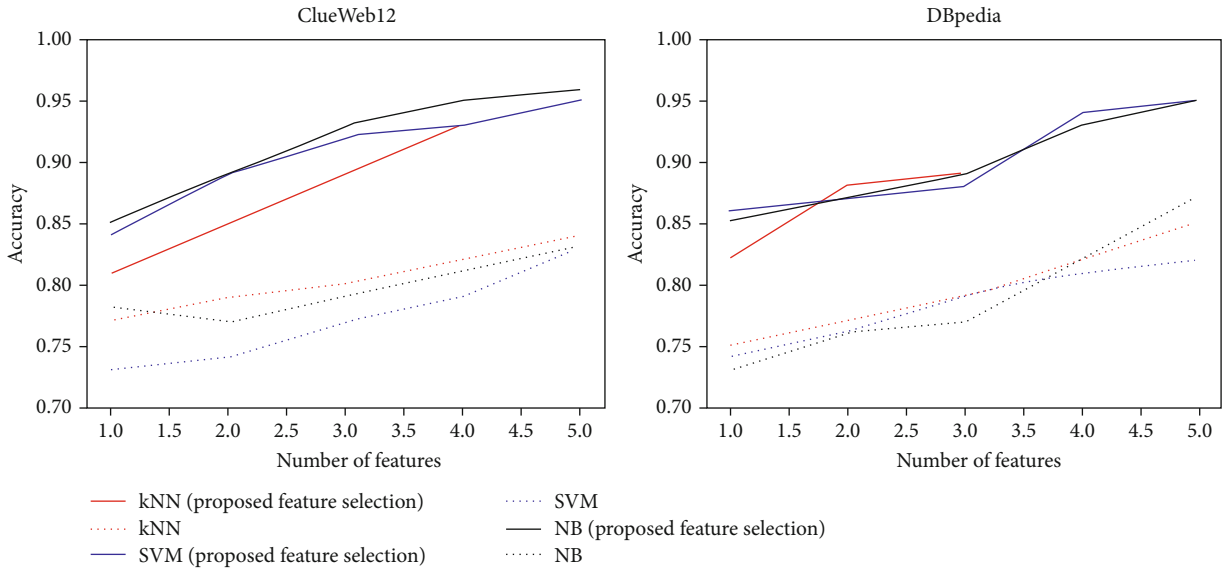


Figure 6: The purity of all the three classifiers when the number of classes = 5.

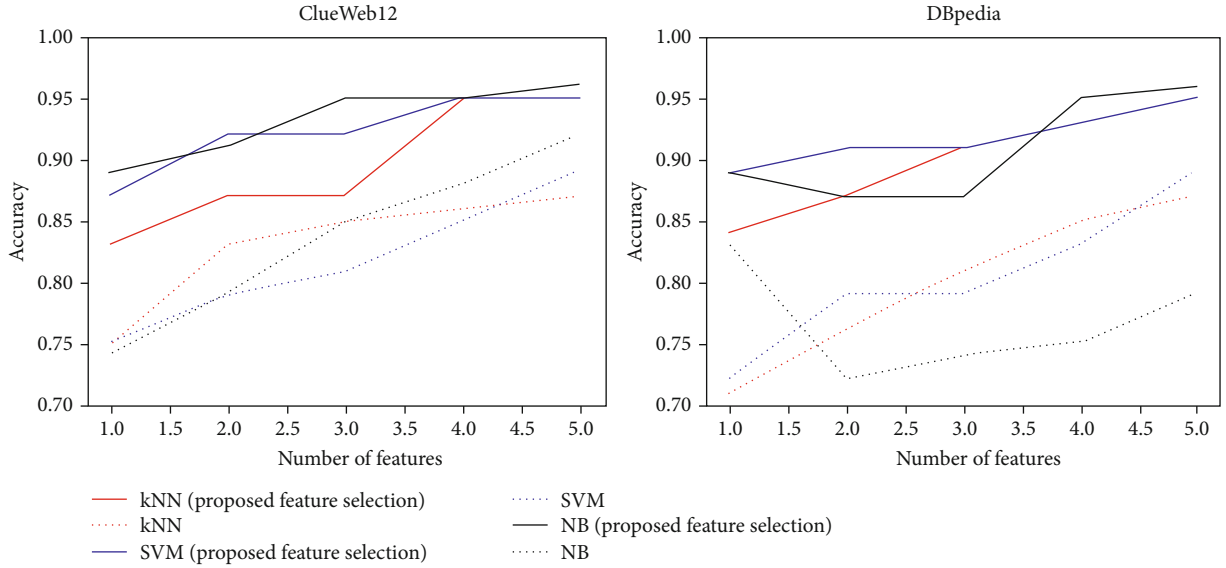


Figure 7: The purity of all the three classifiers when the number of classes = 7.

the immune system, small intestine, are related to health topics, while the other words in the documents such as group, major, have no relation with the real topic (health). Table 5 lists out a few recommended features; it can be easily noted that the features recommended by the proposed system have more correlation towards the class than the features that were recommended based on the high-frequency count. This is why additional information is given more weightage.

6. Conclusion

In this paper, we have proposed an embedded-based feature selection algorithm for recommending top N features to the machine learning model for predicting the correct class for a web document. Webpages in a corpus have links connecting each other; these links provide an additional advantage over the normal text document for classification. The links represent the coupling information, if a webpage has lots of links to a set of webpages, then it has high coupling—that means the target class also have a strong connection. The proposed feature selection algorithm in the paper makes use of this information for classification purposes. Along with the links, the proposed method considers the side information for improving classification accuracy. We have tested our classification model with two real-time benchmark datasets, and the results show that our work has a promising positive effect on classifying web documents. In future work, we will consider knowledge ontology and traffic connections as an additional parameter into account for classification.

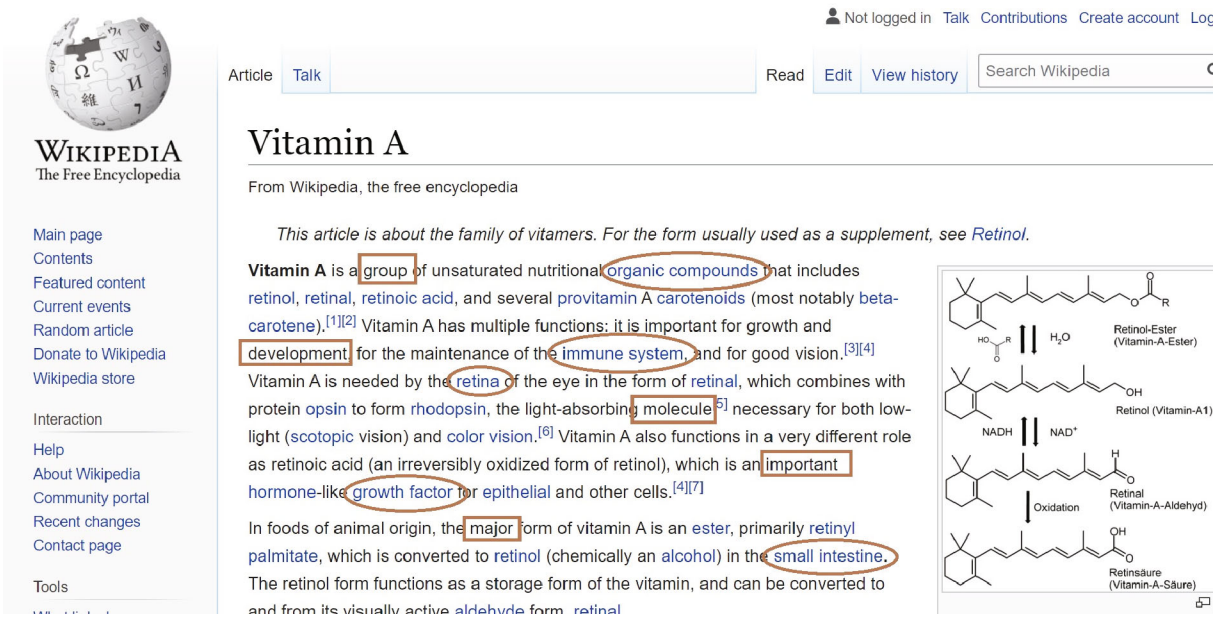


Figure 8: An example that demonstrates the importance of link text over normal text. The link text is circle bordered; normal text is rectangle bordered. It can be noted that circle bordered text talks more about the topic rather than rectangle bordered text.

Table 5: Some of the text features recommended in Figure 8 by existing (based on high-frequency count) and proposed (based on HEF and link text) algorithms.

#	High-frequency words	HEF and link text
1	Database	Medical
2	Reference	Supplementation
3	Functions	Metabolic
4	Reviews	Gene
5	Effects	Immune

Data Availability

Data available on request. The data are available by contacting Ashokkumar P (ashok05002@gmail.com).

Conflicts of Interest

The authors declare that there is no conflict of interest.

Authors' Contributions

P. Ashokkumar and G. Siva Shankar conceived the project, designed experiments, analyzed data and wrote the manuscript. R. Vinayakumar, Uttam Ghosh, Wathiq Mansoor, Waleed S. Alnumay analyzed data and edited manuscript.

Acknowledgments

This research work is supported by the project number (RSP-2020/250), King Saud University, Riyadh, Saudi Arabia. We thank all the reviewers and the special issue editor for proving valuable feedbacks which helped us to improve the quality of our research.

References

[1] K. L. Goh and A. K. Singh, "Comprehensive literature review on machine learning structures for Web Spam classification," *Procedia Computer Science*, vol. 70, pp. 434–441, 2015.

[2] B. Baharudin, L. H. Lee, and K. Khan, "A review of machine learning algorithms for text-documents classification," *Journal of Advances in Information Technology*, vol. 1, no. 1, 2010.

[3] D. Kim, D. Seo, S. Cho, and P. Kang, "Multi-co-training for document classification using various document representations: TF–IDF, LDA, and Doc2Vec," *Information Sciences*, vol. 477, pp. 15–29, 2019.

[4] W. Bai, J. Ren, and T. Li, "Modified genetic optimization-based locally weighted learning identification modeling of ship maneuvering with full scale trial," *Future Generation Computer Systems*, vol. 93, pp. 1036–1045, 2019.

[5] L.-L. Li, J. Sun, M.-L. Tseng, and Z.-G. Li, "Extreme learning machine optimized by whale optimization algorithm using insulated gate bipolar transistor module aging degree evaluation," *Expert Systems with Applications*, vol. 127, pp. 58–67, 2019.

[6] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pp. 39–43, Nagoya, Japan, 1995.

[7] X. Xu, H. Rong, E. Pereira, and M. Trovati, "Predatory search-based chaos turbo particle swarm optimisation (PS-CTPSO): a new particle swarm optimisation algorithm for web service combination problems," *Future Generation Computer Systems*, vol. 89, pp. 375–386, 2018.

[8] L. M. Francis and N. Sreenath, "Robust scene text recognition: using manifold regularized twin-support vector machine," *Journal of King Saud University - Computer and Information Sciences*, 2019.

[9] A. Alaei, P. P. Roy, and U. Pal, "Logo and seal based administrative document image retrieval: a survey," *Computer Science Review*, vol. 22, pp. 47–63, 2016.

[10] B. Al-Salemi, M. Ayob, G. Kendall, and S. A. M. Noah, "Multi-label arabic text categorization: a benchmark and baseline comparison of multi-label learning algorithms," *Information Processing & Management*, vol. 56, no. 1, pp. 212–227, 2019.

[11] T. Dogan and A. K. Uysal, "Improved inverse gravity moment term weighting for text classification," *Expert Systems with Applications*, vol. 130, pp. 45–59, 2019.

[12] S. Yang, R. Wei, J. Guo, and H. Tan, "Chinese semantic document classification based on strategies of semantic similarity computation and correlation analysis," *Journal of Web Semantics*, vol. 63, article 100578, 2020.

[13] Z. Chen, L. J. Zhou, X. D. Li, J. N. Zhang, and W. J. Huo, "The lao text classification method based on knn," *Procedia Computer Science*, vol. 166, pp. 523–528, 2020.

[14] P. Hao, J. Chiang, and Y. Tu, "Hierarchically svm classification based on support vector clustering method and its application to document categorization," *Expert Systems with Applications*, vol. 33, no. 3, pp. 627–635, 2007.

[15] E. H. Houssein, M. R. Saad, K. Hussain, W. Zhu, H. Shaban, and M. Hassaballah, "Optimal sink node placement in large scale wireless sensor networks based on Harris' hawk optimization algorithm," *IEEE Access*, vol. 8, pp. 19381–19397, 2020.

[16] E. H. Houssein, M. E. Hosney, D. Oliva, W. M. Mohamed, and M. Hassaballah, "A novel hybrid Harris hawks optimization and support vector machines for drug design and discovery," *Computers & Chemical Engineering*, vol. 133, article 106656, 2020.

[17] Z. Ning, K. Zhang, X. Wang et al., "Intelligent edge computing in internet of vehicles: a joint computation offloading and caching solution," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–14, 2020.

[18] Z. Ning, R. Y. K. Kwok, K. Zhang et al., "Joint computing and caching in 5G-envisioned internet of vehicles: a deep reinforcement learning based traffic control system," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–12, 2020.

[19] Z. Ning, P. Dong, X. Wang et al., "Mobile edge computing enabled 5G health monitoring for internet of medical things: a decentralized game theoretic approach," *IEEE Journal on Selected Areas in Communications*, 2020.

[20] M.-J. Tsai, Y.-H. Tao, and I. Yuadi, "Deep learning for printed document source identification," *Signal Processing: Image Communication*, vol. 70, pp. 184–198, 2019.

[21] Z. Kastrati, A. S. Imran, and S. Y. Yayilgan, "The impact of deep learning on document classification using semantically rich representations," *Information Processing & Management*, vol. 56, no. 5, pp. 1618–1632, 2019.

[22] Z. Wu, H. Zhu, G. Li et al., "An efficient wikipedia semantic matching approach to text document classification," *Information Sciences*, vol. 393, pp. 15–28, 2017.

[23] M. Mittal, P. Siriaraya, C. Lee, Y. Kawai, T. Yoshikawa, and S. Shimojo, "Accurate spatial mapping of social media data with physical locations," in *2019 IEEE International Conference on Big Data (Big Data)*, pp. 9–12, Los Angeles, CA, USA, December 2019.

[24] P. Siriaraya, Y. Zhang, Y. Wang et al., "Witnessing crime through tweets: a crime investigation tool based on social media," in *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 568–571, Chicago, United States, 2019.

[25] S.-H. Hong, S.-K. Lee, and J.-H. Yu, "Automated management of green building material information using web crawling and ontology," *Automation in Construction*, vol. 102, pp. 230–244, 2019.

[26] L. K. Shih and D. R. Karger, "Using urls and table layout for web classification tasks," in *Proceedings of the 13th conference on World Wide Web - WWW '04*, pp. 193–202, New York, NY, USA, 2004.

[27] X. Zhu, X. Yang, C. Ying, and G. Wang, "A new classification algorithm recommendation method based on link prediction," *Knowledge-Based Systems*, vol. 159, pp. 171–185, 2018.

[28] W. Zhang, T. Yoshida, and X. Tang, "A comparative study of TF∗IDF, LSI and multi-words for text classification," *Expert Systems with Applications*, vol. 38, no. 3, pp. 2758–2765, 2011.

[29] "Vitamin a - wikipedia," 2020, https://en.wikipedia.org/wiki/Vitamin_A.

[30] S. Tan, "An effective refinement strategy for knn text classifier," *Expert Systems with Applications*, vol. 30, no. 2, pp. 290–298, 2006.

[31] J. Calvo-Zaragoza, J. J. Valero-Mas, and J. R. Rico-Juan, "Improving knn multi-label classification in prototype selection scenarios using class proposals," *Pattern Recognition*, vol. 48, no. 5, pp. 1608–1622, 2015.

[32] Q. Wu, Y. Ye, H. Zhang, M. K. Ng, and S.-S. Ho, "Forestexter: an efficient random forest algorithm for imbalanced text categorization," *Knowledge-Based Systems*, vol. 67, pp. 105–116, 2014.

[33] A. Sun, E.-P. Lim, and Y. Liu, "On strategies for imbalanced text classification using svm: a comparative study," *Decision Support Systems*, vol. 48, no. 1, pp. 191–201, 2009.

[34] S. Liu, K. Lee, and I. Lee, "Document-level multi-topic sentiment classification of email data with bilstm and data augmentation," *Knowledge-Based Systems*, vol. 197, p. 105918, 2020.

[35] P. Li, K. Mao, Y. Xu, Q. Li, and J. Zhang, "Bag-of-concepts representation for document classification based on automatic knowledge acquisition from probabilistic knowledge base," *Knowledge-Based Systems*, vol. 193, p. 105436, 2020.

[36] N. Phan, D. Dou, H. Wang, D. Kil, and B. Piniewski, "Ontology-based deep learning for human behavior prediction with explanations in health social networks," *Information Sciences*, vol. 384, pp. 298–313, 2017.