JOURNAL OF OPTOELECTRONICS LASER ISSN:1005-0086

Volume 41 Issue 5, 2022

Text Document Classification Using User Defined String Kernels

N. Veeranjaneyulu[1], Jyostna Devi Bodapati[2], Santhi Sri Kurra[3] & Gayatri Ketepalli[4]
[1,3&4]Department of IT, VFSTR Deemed to be University, Vadlamudi, Guntur, India.
[2]Department of CSE, VFSTR Deemed to be University, Vadlamudi, Guntur, India

ABSTRACT With the intent growth of web-based data, document classification has become an important task that can be used in many real-time applications to handle and organize text documents. In the traditional approaches, text documents are encoded using fixed length feature vector representation. Compared to the static, fixed length representation, a text document can be better represented using variable length feature vector representation, where text documents are allowed to have variable number of features. In this paper, SVM-based classification is used as it does not suffer much from the curse of dimensionality. User-defined kernel functions such as Jaccard coefficient kernel, n-gram kernel, and string subsequence kernels are used to find the kernel value between a pair of documents. To prove the performance of the proposed method, benchmark datasets like Reuters-21578 and Reuters-8, the most widely used datasets for text classification, are used for our experimental studies. Based on our experimental studies, we claim that SVM using N-gram kernel gives better performance on Reuters-21578 dataset and SVM using string subsequence kernel gives better performance on Reuters-8 dataset. We also observed that minor modifications to the user-defined kernel improve the performance of the model.

Keywords: Kernals, Jaccard Coefficient Kernel, SVM, web-based data, N-Gram Kernel.

I. INTRODUCTION

Identifying patterns in the data is popularly known as Pattern recognition. This area received increased demand from researchers in this field as many of the real-time applications can be solved by pattern recognition algorithms. Major subfields of pattern recognition are: Classification, object detection, Clustering, content retrieval, recommender systems, regression and dimensionality reduction. Due to their real-time application these tasks have become so popular. Any machine learning algorithm is basically categorized into supervised, unsupervised or semi-supervised depending on the availability of labels for training the model. Classification and regression tasks come under the umbrella of supervised tasks as the label information is being used while training. On the other hand label information is not being disclosed to the model while training for clustering and dimensionality reduction tasks, and hence come under unsupervised learning.

Classification is a more specific category of regression. In classification task the output is a discrete value. In case of regression the outputs could be continuous values. Regression is predicting a continuous valued output to a given data point. To accomplish classification task, various models are available in the literature like KNN, GMM, MLFFNN and SVM.

Text Classification

With the intent growth of web based data, document classification has become an important task which can be used for handling and organizing text data. Text classification can be used in various real-time applications like spam filtering, in news stories to extract interesting information, web crawling, to classify customers into cohorts, targeted advertisements, sentiment analysis to mention a few. As manually classifying text documents is time consuming and error-prone, there is a need for automation of text document classification.

Text Classification is the process of assigning a given text document to one of the available classes based on the content available in the document. Text can be either structured or unstructured. Structured document is the one that can be easily searchable and is organized. Unstructured data is the one which cannot be searchable and is more like human language.

Support Vector Machines (SVM)

Recent literature shows that Support Vector Machines (SVM) based classifier is a supervised classification technique that gained prominence because it is robust, accurate and are effective even trained with small amounts of training data. SVM is one of the best learning algorithms which can guarantee minimum mis-classifications and do not suffer much from curse of dimensionality. SVM is an empirical risk minimization technique which allows minimizing the error on unseen data whereas the other shallow models minimize the risk on training data alone.

If the data is linearly separable it is possible to draw infinite number of decision boundaries that can well separate the data. On applying any linear classification model like perceptron the model returns one of these separating decision boundaries. SVM is a binary classification method that works with the objective of identifying the hyper-plane with the maximum margin given any two given classes.

With a two-dimensional the the decision boundary returned by SVM is a line and the decision boundary is a plane with three-dimensional data. The decision boundary is a hyper plane for the data with 4 or more dimensions. Margin is the distance from the nearest training example to the decision boundary. Objective of SVM is to find the decision boundary with the maximum margin.

The basic SVM classifier results in a linear boundary which can separate linearly separable data. This linear SVM based classifier can be extended to achieve a more complex decision boundary by transforming the input from its original space to a higher dimensional space. If the relation between the input space and transformed space is non-linear space then SVM can achieve a more complex decision boundary which is a non-linear boundary. The functions used for transforming data to a non-linear space are called as Kernel functions.

SVM is an empirical risk minimization technique which allows minimizing the error on unseen data whereas the other shallow models minimize the risk on training data alone. Thus, the SVM has very good generalization ability compared to the other models. In simple words, given a set of training samples $\{x_1, x_2, x_3 \ldots\ldots\ldots x_n\}$ and associated labels $\{y_1, y_2, y_3 \ldots\ldots\ldots y_n\}$ where $x_i \in R^n$ and $y_i \in \{-1,1\}$, n is the number of examples used for training and the data is belonging to 2 classes (positive and negative). Given the training data the objective of SVM is to find the maximum separating decision boundary (hyper plane). The equation of the hyper-plane can be given by $W^T x + b$ that separate the two classes with the maximum margin. Given the test data point x_k, the decision of assigning it to a class can be taken based on the value of $w^T x_k + b$. If the value is positive then the test point has to be assigned to positive class (class1) and it can be assigned to negative class (class2) if the decision value is negative. For all the test points (x_k), for which $w^T x_k + b = 0$ indicates that x_k are within the margin.

The optimizing function of SVM is designed such that the distance between the two margin hyper-planes given by $\frac{1}{||w||}$ is maximized. The same problem can be posed in a different way as minimizing ||w|| subject to the constraint that: $y_i(w_i^T x_i + w_0)1 \geq 1, \quad \forall i \ldots\ldots\ldots\ldots (1)$

Based on the above two constraints, SVM loss function can be formulated as follows:

$$L_p = \frac{1}{2}||w||^2 - \sum_{i=1}^{n} \alpha_i \{ y_i(w_i^T x_i + w_0) - 1 \} \ldots\ldots\ldots\ldots (2)$$

The above optimization problem can be solved by using QP solver. When the data is linearly separable the above loss function can be solved easily. If the non-linearly separable then the loss function becomes more complex and difficult to solve. But unfortunately, many of the real time applications like text data classification are very complex in nature and a linear decision boundary would not be sufficient to separate the classes.

SVMs can also be extended to achieve a more complex non-linear boundary by projecting the data into a different space than the original. The degree of non-linearity of the decision boundary depends on the transformation. To handle the non-linearity a slack variable ξ_i is introduced in the loss function. After introducing the slack, the loss function can be reformulated as follows:

$$min \ \frac{||w||^2}{2} + C \ (\sum_i \varepsilon_i)^{\ k} \ldots\ldots\ldots\ldots (3)$$

Subject to the constraint that $\left(w^T x_i + w_0\right) \geq 1 - \varepsilon_i, \quad \forall y_i = 1 \ldots\ldots\ldots\ldots (4)$

$$\left(w^T x_i + w_0\right) \geq -1 + \ \varepsilon_i, \quad \forall y_i = -1 \ and \ \varepsilon_i \geq 0 \quad \ldots\ldots\ldots\ldots (5)$$

The above loss function is in primal form, can be converted to its equivalent dual form and can be represented as follows:

$\Sigma\alpha_i - (1/2) * \Sigma\alpha_i \Sigma \alpha_j y_i y_j x_i^T x_j$ subject to $0 \leq \alpha_i \leq C$ and $\Sigma\alpha_i y_i = 0$ ……..(6)

In (6), the term $x_i^T x_j$ is a dot product of the two data points x_i and x_j. A dot product can be used to find the similarity between two data points. It can be replaced by any kernel function $K(x_i, x_j)$. A kernel function is basically determining the similarity between two data points. If $x_i^T x_j$ is replaced with $K(x_i, x_j)$ then equation (6) can be written as:

$\Sigma\alpha_i - (1/2) * \Sigma\alpha_i \Sigma \alpha_j y_i y_j K(x_i, x_j)$ subject to $0 \leq \alpha_i \leq C$ and $\Sigma\alpha_i y_i = 0$ ……..(7)

The kernel function K(x_i, x_j) represents the similarity between x_i and x_j in the transformed space. This transformation is often a non-linear transformation usually to higher dimensional space using a transformation function 'Φ' and can be represented as: $K(x_i, x_j) = \emptyset(x_i)^T \emptyset(x_j)$.

Kernel Methods

According to the information theorist Thomas M.Cover, non-linearly separable data in the original space can be transformed to a space where it can be linearly separable by applying some non-linear transformation. This non-linear transformation is called as the kernel function. The difficulty here is to identify the suitable kernel function which can transform the data such that it can be linearly separable in the projected space.

The expected transformation from original space to higher dimensional space where the data can be linearly separable is possible only with the Mercer kernel. A mercer kernel has to satisfy certain properties [*]. In few types of kernels it is not possible to represent data in the transformed data. For example, in case of Radial basis function kernel (RBF) the dimensionality of the data in transformed space is infinite. It is enough if the dot products of two data points in the projected space can be represented. In simple words the pair-wise inner-product i.e. $\emptyset(x_i)^T \emptyset(x_j)$ can be computed without computing $\emptyset(x_j)$ explicitly. This is known as kernel trick. Between every pair of examples, we need to find, $K(x_i, x_j) = \emptyset(x_i)^T \emptyset(x_j)$. Kernel matrix matrix of size n x n where n is the number of examples in the dataset, where the cell indicated by i^{th} row j^{th} column represents the kernel value between the examples x_i and x_j.

II. BACKGROUND WORK

The task of text document classification involves the process of assigning one of the available labels to every text document on hand. SVM can be used to classify many real-world problems especially used successfully for text document classification [19]. In this task SVM is used to classify the given text documents into one of the given classes. Experiments are performed extensively using various user-defined kernel functions. However, the generalization ability of SVM is dictated by the type of kernel used for data mapping. In other words if the kernel selected is able to transform the data such that it is separable by linear boundary in the high-dimensional space then the performance of the classifier would be high. But which type of kernel has to be used on the data is still an open problem. Lot research has been going on in the recent past to develop algorithms that helps us to choose the appropriate type of kernel for the given dataset.

As of now there is no standard method to select the appropriate kernel and the type of kernel has to be chosen empirically with trial and error methods. If the data is 1D, 2D or 3D then it is possible to visualize the data and the type f the kernel can be selected easily. As the dimensionality of the data grows it is not possible to visualize the data and the kernel selection becomes complex [5]. Many different types of kernels are available in the literature and the most widely used kernels are listed below in Table1. These are also known as general purpose kernels:

Table1: List of general purpose kernels

Kernel Type	Transformation Function	Hyper parameters
Linear	$K(x_i, x_j) = (x_i^T x_j)$	*****
Polynomial	$K(x_i, x_j) = (1 + x_i^T x_j)^p$	p
Gaussian	$K(x_1, x_2) = \frac{e^{-\|\|x_1 - x_2\|\|^2}}{2\sigma^2}$	σ
Sigmoid	$K(x_i, x_j) = \tanh(\gamma x_i^T x_j + C)$	γ, C

Gaussian kernel, also known as Radial basis function (RBF) kernel projects data onto an infinite dimensional space. In RBF $K(x_i, x_j)$ is close to 1 when x_i and x_j are belonging to the same class and $K(x_i, x_j)$ is close to 0 when x_i and x_j are not belonging to the same class. The function involved in all the general purpose kernels is provided in Table1.

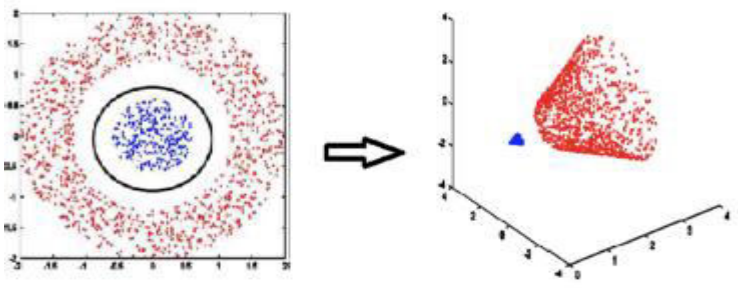

Figure1: Data transformed from a non-linearly separable space to linearly separable space

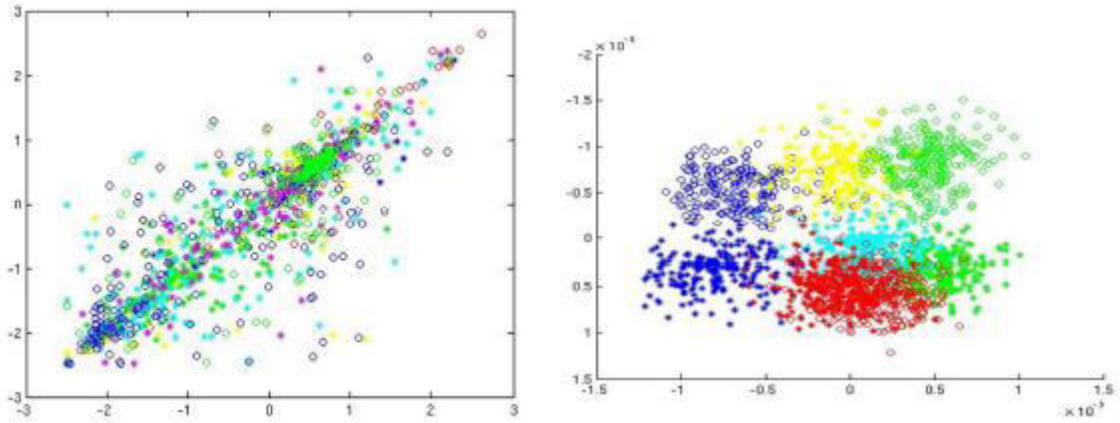

Figure2: Data transformed from 2-D space to 7-D space

Related work

Though SVM exhibits good generalization ability, features extraction has to be done before applying the classification process. It is not possible to extract the features using the readily available extraction techniques for certain types of data such as text, graphs, DNA sequences. One simple solution is to apply suitable kernel to extract the features. In this work we focus on different kernels available in literature and that can be applied on text documents.

Though bag of words representation is used extensively to represent text documents, it is neither effective nor efficient to use them straightway on the text according to [1]. In [2], the authors demonstrate why SVMs are appropriate and effectively used for text document classification. In addition the authors also analyzed what are the important properties that are be learnt to efficiently classify the data. In [3] proposes how the inner products can be computed using dynamic programming techniques to reduce the computational time. In this work approximation kernels are also introduced which can be efficiently applied on larger datasets. In [4] a novel algorithm for active learning with SVM based classifier has been introduced. In [5] semantic kernel is proposed to improve the performance of (BoW) representation by incorporating the knowledge available from Wikipedia. In [6], an extensive study on different feature selection methods has been introduced for text document classification. Sentences can be represented as trees and in [7] different tree kernels are introduced to find the similarity.

A novel sequence similarity measure called spectrum kernel is introduced in [8] and is applied on protein sequence data to exhibit the performance of spectrum kernel. In [9] a class of string kernels called mismatch kernels has been used on protein sequence data and remote homology detection. These kernels are used in a discriminative approach with SVM based classifier. A key of prominent substrings are considered while applying BoW approach to improve the performance of text document classification [10].

III. USER DEFINED KERNEL FUNCTIONS

Need of user defined kernels

Using appropriate transformation function with SVM can improve the generalization ability of the classifier. General purpose kernels may not give good performance for real time applications like image and text data. Literature shows that for image data user defined kernels like histogram intersection kernel gives better accuracy [3] compared to the standard kernels. In addition to this major constraint to apply any standard kernels is that the data points must be represented using static feature vector representation. In static feature vector representation all the examples in the dataset are represented as a fixed length feature vectors. Then only it is feasible to compute dot product (kernel function) between every pair of examples.

In case of text document classification, text documents are better represented using variable length feature vector representation compared to static feature vector representation. In variable length feature vector representation different examples in the dataset are allowed to have different number of features. So on text data it is not possible to apply standard kernels when they are represented using variable length feature vector representation.

User defined String Kernels

We have applied three different user defined string kernels for the task of document classification task.

Jaccard Coefficient Kernel

The Jaccard Similarity Index can be used to represent the similarity between two documents [5], [6]. A value "0" indicates that the documents are completely dissimilar to each other and value "1" means that they are identical to each other. Values between 0 and 1 represent degree of similarity between the documents. The Jaccard index between a pair of documents x_i and x_j can be represented as the ratio between the size of the intersection of the two documents and the size of the union of the two documents. The Jaccard coefficient between two text documents x_1 and x_2 can be measured as:

$$K(x_i, x_j) = \frac{x_i \cap x_j}{x_i \cup x_j} = \frac{x_i \cap x_j}{|x_i|+|x_j|-|x_i \cap x_j|} \quad \text{.....(8)}$$

In (8), $K(x_i, x_j)$ represents the Jaccard similarity between the documents x_i and x_j. The denominator term of the equation (8) indicates the union operation negates the shared terms in xi and x_j.

String Subsequence Kernel

This approach considers the documents as symbol sequences [7] [8]. The feature space in this kernel is formed by the set of all substrings of k-symbols. Most interesting part in this subsequence kernel is that the substring we are looking for need not to be contiguous in the original document. This kernel can be defined on any two text documents by comparing the substrings that exist in both the documents. This kernel indicates that of there exists more number of substrings that are common in the documents, and then the documents are more similar. Depending on the degree of contiguity of a substring in a document, a weight value will be assigned in the comparison. Consider two strings: Captain and Champion. The substring 'c-a-p' is present in both the strings but with different weights. Given any two documents s and t, the string subsequence kernel on can be defined as:

$$K(s,t) = \sum_{u \in \sum^k} \sum_{i:u=s[i]} \sum_{j:u=t[j]} \lambda^{l(i)+l(j)} \quad \text{... ... ... (9)}$$

In (9) k and λ are the hyper-parameters. k represents subsequence length and λ represents decay factor where λ∈(0, 1). Then for any two given strings s and t, $\sum^k$ is the set of all substrings of length k, i is the vector of indices such that $u_j = s_{ij}$; for j = 1…..|u|, similarly the vector j for string t, and the length l(i) of the subsequence in s is $i_{|u|} - i_1 + 1$.
The advantage of using this substring matching kernel is that it can capture the word order information.

N-Gram Kernel

This kernel is independent of the language and can be used with structured data like text documents. The only parameter in this kernel is 'N'. It transforms documents to a different space which is usually higher dimensional and finds similarity in that projected space. Each feature vector in the projected space corresponds to a substring. N-gram can be considered as a substring (N adjacent characters) of length n of the alphabet A. The number of possible n-grams is typically less than or equal to $|A|_n$. Based on this it is obvious that the length of the feature vector represented using n-grams is very large even with medium values of n. But all the n-grams listed are not necessarily occur in a document. Thus the resultant feature vector would be sparse and reducing the dimensionality is a substantial task. For example consider the list of all 3-grams present in the string "Document classification" are: {"Doc", "ocu", "cum", "ume", "men", "ent", "nt ", t c", " cl", "cla", "las", "ass", "ssi", "sif", "ifi", "fic", "ica", "cat", "ati", "tio" "ioo", "oon"}.

Some kind of pre-processing has to be done before applying the n-grams feature vector formation. Initially the whole text in the document is converted to lower-case and all the punctuation symbols are replaced by a space. The resultant feature vectors are then normalized to guarantee scaling within the features.

Initially, n value is fixed where n stands for the gram size and then all the possible n-grams are listed in the training corpus. Once n-gram identification is over, then each sample of the training data has to be represented as a Boolean feature vector. Presence of an n-gram in the document is represented by a '1' and absence of the n-gram is represented by a '0'. In this approach each document (or string) is represented by a vector x. Thus the similarity between two documents x_m and x_n can be defined as:

$$K(x_m, x_n) = \frac{x_m^t x_n}{\sqrt{x_m^t x_m}\sqrt{x_n^t x_n}}$$

Where x_m and x_n are the n-gram representations of the documents d_m and d_n respectively. Instead of considering the Boolean values count of the sub-string occurred can also be measured. Illustration of computing kernel value: For simplicity assume two documents contain strings "Computer" and "Compiler" respectively in d_1 and d_2. Let x_1 and x_2 be the 3-gram representation of the documents d_1 and d_2 respectively.

	com	Omp	mpu	put	ute	ter	mpi	pil	ile	ler

x1	1	1	1	1	1	1	0	0	0	0
x2	1	1	0	0	0	0	1	1	1	1

$$K(x_1, x_2) = \frac{2}{\sqrt{6}\sqrt{6}} = 0.4098$$

This method can be extended to large documents to find the similarity between documents.

IV. PROPOSED METHOD

Modifications to the user defined kernels

We have used all the kernels (Jaccard Coefficient Kernel, N-Gram Kernel, string Subsequence Kernel) described in the previous section to find the similarity between every pair of documents in the dataset. As the documents considered are web pages, identify the number of hash tags used, capital letters, positive emoji, negative emoji, symbols, etc. and created a 13-feature vector for each line. This 13 feature vector representation is used to compare one line to the other. That similarity is then added to the Kernel function. Similarly, for all pairs of documents, the value is found, and the whole kernel matrix is populated for the training dataset. Now using this predefined kernel and the labels, the test documents are classified using the SVM based classifier.

Figure3 presents the detailed workflow of the proposed classification model for text document classification task.

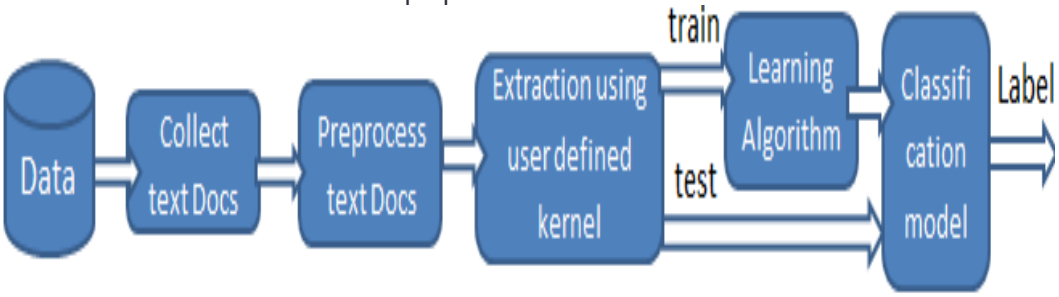

Figure3: Steps involved in proposed text classification model

The objective of this work is to classify the given text documents efficiently with minimum number of mis-classifications.

5 EXPERIMENTAL RESULTS AND COMPARATIVE RESULTS

Data set Description

For experimental results we have used the Reuters dataset. This is the most popular bench mark text classification dataset. The Reuters datasets Reuters-21578 and R8 are widely used datasets in this field.

The Reuters-21578 dataset: It is a collection of reports/texts from different classes like 'usa', 'canada', 'corn', 'crude' and 'acq'. For our experiments we have considered the limited dataset with four classes (earn, trade, crude and acq) of articles from the Reuters-21578 dataset for performing our classification task.

The Reuters-8 dataset (R8): Summary of this R8 dataset is provided in table1 along with the documents per class statistics:

Table 2: Reuters-8 (R8) benchmark dataset summary

R8			
Class	**# train docs**	**# test docs**	**Total # docs**
acq	1596	696	2292
crude	253	121	374
earn	2840	1083	3923
grain	41	10	51
interest	190	81	271
money-fx	206	87	293
ship	108	36	144

trade	251	75	326
Total	**5485**	**2189**	**7674**

Reuters-8 data set contains a total of 5485 train documents and 2189 test documents. For our experiments a subset of this dataset is chosen by randomly picking 25 examples from each of the class for training and 25 examples from each class for testing.

Performance of Word Embeddings on Reuters-21578 dataset

N-gram kernel is applied on Reuters Dataset Reuters-21578 with different n-values. We have tried 1-gram, 3-grams and 4-grams to find the resultant kernel value. The number of features extracted is: 119, 11062 and 39081 respectively for n=1, 3 and 4. Table 3 shows the Performance of n-gram kernel on Reuters-21578 for different values of n.

Table 3: Accuracies of Reuters-21578 dataset using n-gram kernel

N (N-grams)	Accuracy
1	72.37
3	94.46
4	**95.53**

Table4 shows the Performance of String Subsequence Kernel on Reuters-21578 for different values for the hyper-parameters k and λ are used in our experimental studies.

Table4: Accuracies of Reuters-21578 dataset using string subsequence kernel

Decay (λ)	Length of the substring (K)		
	K=2	K=3	K=5
0.01	89.25%	86.00%	62.25%
0.1	90.00%	86.75%	63.00%
0.5	91.00%	86.50%	70.25%
0.9	93.75%	88.75%	68.25%
0.95	**94.25%**	88.75%	70.25%
0.99	93.75%	89.50%	68.25%

Best accuracy of 94.25% is reported using String Subsequence Kernel for the classification of the documents of the Reuters-21578 dataset. This approach captures better information about the documents compared to the normal bag of words representation. We feel that this String Subsequence Kernel is able to preserve the underlying sequence information which is completely ignored by the bag of words representation. This kernel does not bother about the content available in the document. In other words this kernel treats a document as a long sequence of strings. Hence this kernel can be applied to any type of text documents irrespective of the content in the documents. On the other side this String subsequence kernel takes more time and space as it is dependent on the dynamic programming approach.

Performance of Word Embeddings on Reuters-8 dataset

N-gram kernel is applied on Reuters Dataset Reuters-8 with different n-values. We have tried all the values from n=3 to 10 to compute n-grams and then find the resultant kernel value. Table5 shows the confusion matrix of Reuters-8 dataset using N-gram kernel with N=7.

Table 5: confusion matrix of Reuters-8 dataset using 7-gram kernel

	acq	crude	earn	grain	interest	money-fx	ship	trade
Acq	25	0	0	0	0	0	0	0
Crude	0	24	1	0	0	0	0	0
Earn	0	0	25	0	0	0	0	0
Grain	0	0	0	25	0	0	0	0
interest	0	0	0	0	19	4	0	2

money-fx	0	0	0	0	0	24	0	1
Ship	0	0	1	0	0	0	24	0
Trade	0	0	0	1	0	0	0	23

Table 5 shows confusion matrix of Reuters-8 dataset using 7-gram kernel. Table6 shows the Performance of n-gram kernel on Reuters-8 for different values of n.

Table 6: Accuracies of Reuters-8 dataset using n-gram kernel

N (N-grams)	3	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Accuracy	28.15%	66.83%	75.37%	76.36	76.88%	75.37%	74.37%	70.85%

Table8 shows the Performance of subsequence kernel on Reuters-8 for different values of k.

Table7: Accuracies of Reuters-8 dataset using subsequence kernel

N (N-grams)	3	**4**	**5**	**6**	**7**	**8**
Accuracy	36%	58%	66.5%	60%	57%	54.5%

For all the experiments on Reuters-8 dataset using subsequence kernel Decay value λ is set to 0.5. Best accuracy of 76.88% is reported using 7-gram kernel for the classification of the documents of the Reuters-8 dataset. Performance of n-gram kernel on R8 dataset gives better performance compared to the other kernels. As the N-gram kernel picks up more number of features it gives better performance.

Performance comparison of different kernels:

In this experiment the performance of subsequence kernel is compared with the performance of n-gram kernel. Table8 summarizes the accuracy of the models on Reuters-21578 and Reuters-8 datasets.

Table8. Performance Comparison of different kernels on Reuters-8 and Reuters-21578

Dataset	**Kernel type**	
	N (N-grams)	**String-Subsequence**
Reuters-21578	95.33%	94.25%
Reuters-8	76.88%	66.5%

6 CONCLUSION

User defined kernels can be used for classification of text documents. String Subsequence Kernel gives very good accuracy for the classification of the documents of the Reuters-21578 dataset. This approach captures more information about the document than the normal bag of words classification for texts. Mainly it captures the word order which is disregarded by the bag of words model. This kernel does not use any domain knowledge, in the sense that it considers the document just as a long sequence. The N-gram kernel performs the best in almost all the cases of Reuters-8 dataset. The N-gram kernel is able to pick more prominent features. The String subsequence kernel takes more time as it is dependent on the dynamic programming approach.

REFERENCES

[1] Vikas Thada and Dr Vivek Jaglan, “Comparison of Jaccard, Dice, Cosine Similarity Coefficient To Find Best Fitness Value for Web Retrieved Documents Using Genetic Algorithm”, International Journal of Innovations in Engineering and Technology (IJIET), Vol. 2 Issue 4, SSN: 2319-1058, August 2013.

[2] Suphakit Niwattanakul, Jatsada Singthongchai, Ekkachai Naenudorn and Supachanun Wanapu, “Using of Jaccard Coefficient for Keywords Similarity”, Proceedings of the International Multi Conference of Engineers and Computer Scientists 2013 Vol I, IMECS 2013, March 13-15, 2013, Hong Kong.

[3] Hassiba Nemmour, Youcef Chibani, “New Jaccard-Distance Based Support Vector Machine Kernel for Handwritten Digit Recognition”, ICTTA, 2008.

[4] Bodapati, J.D.; Naralasett, V.; Shareef, S.N.; Hakak, S.; Bilal, M.; Maddikunta, P.K.R.; Jo, O. Blended Multi-Modal Deep ConvNet Features for Diabetic Retinopathy Severity Prediction. *Electronics* **2020**, *9*, 914.

[5] Jyostna Devi Bodapati and N. Veeranjaneyulu, "Abnormal Network Traffic Detection Using Support Vector Data Description." Proceedings of the 5th International Conference on Frontiers in Intelligent Computing: Theory and Applications. Springer, Singapore, 2017.

[6] Jyostna Devi Bodapati and N. Veeranjaneyulu, "Performance of different Classifiers in non-linear subspace" International Conference on Signal and Information Processing (IConSIP-2016).

[7] Veeranjaneyulu N and Jyostna devi Bodapati, "Scene classification using support vector machines with LDA", Journal of theoretical and applied information technology, 2014, 63(3), pp.741

[8] Bodapati, J.D., Shaik, N.S. & Naralasetti, V. Deep convolution feature aggregation: an application to diabetic retinopathy severity level prediction. *SIViP* **15,** 923–930 (2021). https://doi.org/10.1007/s11760-020-01816-y

[9] Bodapati, J.D., Shaik, N.S., Naralasetti, V. *et al.* Joint training of two-channel deep neural network for brain tumor classification. *SIViP* **15,** 753–760 (2021). https://doi.org/10.1007/s11760-020-01793-2

[10] Bodapati, J.D., Krishna Sajja, V.R., Mundukur, N.B., Veeranjaneyulu, N. (2019). Robust cluster-then-label (RCTL) approach for heart disease prediction. Ingénierie des Systèmes d'Information, Vol. 24, No. 3, pp. 255-260. https://doi.org/10.18280/isi.240305

[11] Bodapati J.D., Veeranjaneyulu N., Shaik S. (2019). Sentiment analysis from movie reviews using LSTMs, Ingenierie des Systemes d'Information, Vol. 24 No. 1, pp. 125-129. https://doi.org/10.18280/isi.240119

[12] Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, **"**Text Classification using String Kernels", Journal of Machine Learning Research, VOl: 2, PP: 419-444, 2012

[13] Berna Altınel, Murat Can Ganiz, Banu Diri, "A corpus-based semantic kernel for text classification by using meaning values of terms", Engineering Applications of Artificial Intelligence, Volume 43, August 2015, Pages 54–66

[14] Pu Wang and Carlotta Domeniconi**, "**Building Semantic Kernels for Text Classification using Wikipedia", KDD'08, August 24–27, 2008.

[15] Veeranjaneyulu, N., M. Nirupama Bhat, and A. Raghunath. "Approaches for managing and analyzing unstructured data." *International Journal on Computer Science and Engineering* 6.1 (2014): 19.